STA 437/2005: Methods for Multivariate Data

Week 11: Conditional independence

Piotr Zwiernik

University of Toronto

Basic definitions

Random vector and independence

Let (X, Y) be a vector of two random variables.

Joint distribution

Density function $f_{XY}(x, y)$ if continuous.

Probability mass function $f_{XY}(x, y) = \mathbb{P}(X = x, Y = y)$ if discrete.

Marginal distribution

continuous: $f_X(x) = \int_{\mathbb{R}} f_{XY}(x, y)\mathrm{d}y$.

discrete: $f_X(x) = \sum_y f_{XY}(x, y) = \mathbb{P}(X = x)$.

This can be generalized to random vectors.

Independence

If $f_{XY}(x, y)$ is the joint density (or PMF) of (X, Y) then X and Y are independent if and only if

$$f_{XY}(x, y) = f_X(x) f_Y(y) \qquad \text{for all } x, y.$$

We write $X \perp\!\!\!\perp Y$.

Recall:

$$\text{cov}(X, Y) = \mathbb{E}(XY) - \mathbb{E}(X)\mathbb{E}(Y) \quad \text{and} \quad \mathrm{var}(X) = \text{cov}(X, X).$$

The correlation $\rho_{X,Y}$ between X, Y is:

$$\rho_{X,Y} \;=\; \frac{\text{cov}(X,Y)}{\sqrt{\mathrm{var}(X)\mathrm{var}(Y)}} \;\in\; [-1, 1].$$

If $X \perp\!\!\!\perp Y$ then $\rho_{X,Y} = 0$.

(but in general not the other way around, see slide 13)

Conditional distribution

Conditional distribution

In the discrete case the conditional probability mass function is defined as

$$f_{X|Y}(x|y) = \mathbb{P}(X = x|Y = y) = \frac{\mathbb{P}(X=x, Y=y)}{\mathbb{P}(Y=y)}$$

for all x, y such that $\mathbb{P}(Y = y) > 0$ and so

$$f_{X|Y}(x|y) = \frac{f_{XY}(x,y)}{f_Y(y)} \text{ for all } x, y \text{ s.t. } f_Y(y) > 0.$$

In the continuous case we use the same definition.

Important reformulation of independence

$X \perp\!\!\!\perp Y$ if and only if $f_{X|Y}(x|y) = f_X(x)$.
(knowing Y brings no extra information about X)

A cautionary note

Note: $f_{X|Y}(x|y) \neq f_{Y|X}(y|x)$.

Example: A medical test for a disease D has outcomes $+$ and - with probabilities

	D	D^c
$+$	.009	.099
-	.001	.891

As needed $\mathbb{P}(+|D) = 0.9$ and $\mathbb{P}(-|D^c) = 0.9$. However, $\mathbb{P}(D|+) \approx 0.08$ (!)

Conditional independence

X, Y, Z random variables.

X is independent of Y given Z (write $X \perp\!\!\!\perp Y|Z$) if

$$f_{XY|Z}(x, y|z) \;=\; f_{X|Z}(x|z) f_{Y|Z}(y|z) \qquad \text{for every } z.$$

Important reformulation of independence

$X \perp\!\!\!\perp Y|Z$ if and only if $f_{X|Y,Z}(x|y, z) = f_{X|Z}(x|z)$.
(if we observed Z, extra information about Y brings no extra information about X)

Testing independence

Recall: A statistical test

Given a statistical hypothesis $H_0 : \theta \in \Theta_0$, $H_1 : \theta \in \Theta_1$, a statistical test consists of a test statistics $T(X^{(1)}, \ldots, X^{(n)})$ and a rejection region, typically of the form

$$R = \{T(X^{(1)}, \ldots, X^{(n)}) > t\}.$$

If the null hypothesis is true T is unlikely to take large values.

Type I error: $\mathbb{P}(T \in R | H_0)$

although H_0 is true, it is rejected

Type II error: $\mathbb{P}(T \notin R | H_1)$

although H_0 is false, it is retained

A good test should minimize probabilities of both types of errors.

Testing independence

Data: $(X_1, Y_1), \ldots, (X_n, Y_n) \overset{iid}{\sim} P_{X,Y}$.

Goal: Decide whether $X \perp\!\!\!\perp Y$.

Statistical test: $\quad H_0 : X \perp\!\!\!\perp Y, \quad H_A : X \not\perp\!\!\!\perp Y$

There are many tests of independence.

We discuss some examples.

Test for vanishing correlation

Fisher's z-transform test for Gaussian data

Let r_n is the sample correlation coefficient from an *iid* sample $(X^{(i)}, Y^{(i)})$.

Define $Z_n = \frac{1}{2} \log \left(\frac{1+r_n}{1-r_n} \right)$.

If (X, Y) is bivariate normal with correlation ρ then Z_n has asymptotically normal distribution with mean $\frac{1}{2} \log \left(\frac{1+\rho}{1-\rho} \right)$ and variance $\frac{1}{n-3}$.

Fisher's z-transform test is implemented in R as `cor.test`.

Non-gaussianity may invalidate the test and affect its power.

Basic nonparametric test

Kendall's tau test for non-Gaussian data

Suppose a bivariate sample (x_i, y_i) for $i = 1, \ldots, n$ is given.

Pair (x_i, y_i) , (x_j, y_j) is concordant if $(x_i, y_i) < (x_j, y_j)$ or $(x_i, y_i) > (x_j, y_j)$. Otherwise discordant.

Define $\tau_{XY} = \frac{(\#\text{concordant}) - (\#\text{discordant})}{\binom{n}{2}} \in [-1, 1]$.

Test based on Kendell's τ statistic is implemented in R as `cor.test`.

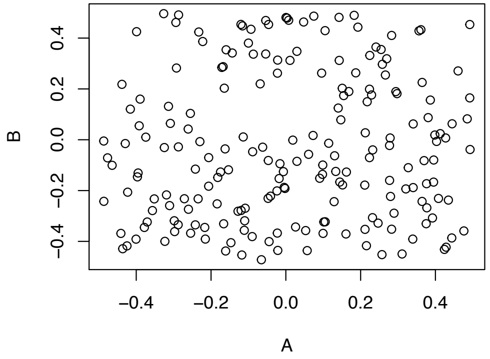
```
> set.seed(1); n <- 200; rho <- 0.2; Z <- runif(n);
> X <- runif(n)^2+sqrt(rho)*Z; Y <- runif(n)+sqrt(rho)*Z
> cor.test(X, Y, method = "pearson")$p.value
[1] 0.03417231
> cor.test(X, Y, method = "kendall")$p.value
[1] 0.01100592
```

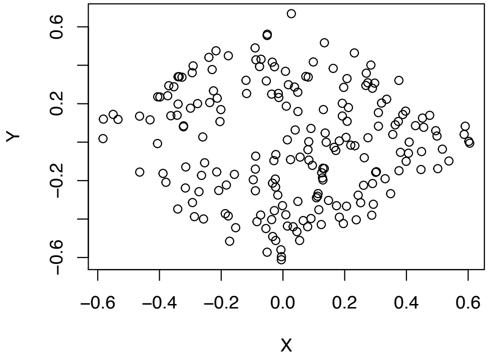
Non-Gaussianity issue

Vanishing covariance does not imply independence!

```
# generate sample from two uncorrelated but dependent random variables
> set.seed(1); n <- 200
> A <- runif(n)-1/2; B <- runif(n)-1/2
> X <- t(c(cos(pi/4),-sin(pi/4)) %*% rbind(A,B))
> Y <- t(c(sin(pi/4),cos(pi/4)) %*% rbind(A,B))
> cor.test(X,Y, method = "pearson")
# Pearson's product-moment correlation
data:  X and Y
t = -0.84711, df = 198, p-value = 0.398
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.1971897  0.0793095
sample estimates:
        cor
-0.06009275
```

X and Y are uncorrelated but dependent!





We see that X and Y are highly dependent.

Test based on distance correlation

Distance correlation $\mathcal{R}(X, Y)$ provides a test which applies when X, Y are two random **vectors** of any dimensions.

$\mathcal{R}(X, Y) = 0$ if and only if X and Y are independent.

The sample version of $\mathcal{R}(X, Y)$ gives a nonparametric test of independence.

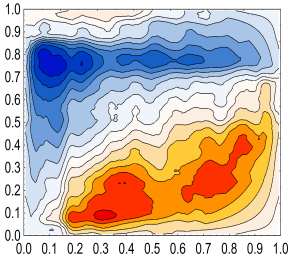
```
> library(energy); set.seed(1); n <- 200
> A <- runif(n)-1/2; B <- runif(n)-1/2
> X <- t(c(cos(pi/4),-sin(pi/4)) %*% rbind(A,B))
> Y <- t(c(sin(pi/4),cos(pi/4)) %*% rbind(A,B))
> dcor.test(X,Y,R=1000)

# dCor independence test (permutation test)
data:  index 1, replicates 1000
dCor = 0.21161, p-value = 0.004995
sample estimates:
      dCov        dCor     dVar(X)     dVar(Y)
0.03999654 0.21160982 0.17870935 0.19990601
```

Here $R = 1000$ is the number of the permutation bootstrap replications.

Another cautionary example

Bowman& Azzalini (1997) analyse aircraft wing span and speed data.



```
> library(sm); set.seed(1);
> X <- aircraft$Span
> Y <- aircraft$Speed
> cor.test(X,Y)$p.value
[1] 0.7816014
> dcor.test(X,Y,R=1000)$p.value
[1] 0.000999001
```

Tests for discrete data

χ^2-test for discrete data

```
> M <- as.table(rbind(c(762, 327, 468), c(484, 239, 477)))
> dimnames(M) <- list(gender = c("F", "M"),
+                     party = c("Democrat","Independent", "Republican"))

> (Xsq <- chisq.test(M))  # Prints test summary

Pearsons Chi-squared test

data:  M
X-squared = 30.07, df = 2, p-value = 2.954e-07

> Xsq$expected   # expected counts under the null
      party
gender Democrat Independent Republican
     F 703.6714    319.6453   533.6834
     M 542.3286    246.3547   411.3166
```

`df = 2` is the difference between 5 (saturated model) and 3 (independence)

Testing conditional independence

Testing conditional independence is hard in general.

For discrete data we have the asymptotic χ^2-test.

Some parametric tests are implemented in the library `bnlearn`.
Many non-parametric methods have been implemented in `CondIndTest`

```
> library(CondIndTests); library(bnlearn); set.seed(1); n <- 100
> Z <- rnorm(n); X <- 4 + 2 * Z + rnorm(n); Y <- 3 * X^2 + Z + rnorm(n)
> CondIndTest(X,Y,Z, method = "KCI")$pvalue
[1] 2.419926e-10
> bnlearn::ci.test(X,Y,Z)$p.value
[1] 1.15458e-25
```

See Section 3 in: C. Heinze-Deml, J. Peters, N. Meinshausen, Invariant Causal Prediction for Nonlinear Models, Journal of Causal Inference, 2018.

See: `http://www.bnlearn.com/documentation/man/conditional.independence.tests.html`

Simpson's paradox: UC Berkeley admissions example

The admission figures of the grad school at UC Berkeley in 1973: 8442 (44%) men, 4321 (35%) women admitted.

The same data conditioned on the department are:

Department	Men		Women	
	Applicants	Admitted	Applicants	Admitted
A	825	62%	108	**82%**
B	560	63%	25	**68%**
C	325	**37%**	593	34%
D	417	33%	375	**35%**
E	191	**28%**	393	24%
F	373	6%	341	**7%**

"Measuring bias is harder than is usually assumed, and the evidence is sometimes contrary to expectation."

(Bickel et al, *Sex Bias in Graduate Admissions: Data From Berkeley*, Science, 1975)

In R:

```
> library(gRim); data(UCBAdmissions)
> bnlearn::ci.test(x = "Gender" , y = "Admit", z = "Dept", test="x2",data = as
    .data.frame(UCBAdmissions))

Pearsons X^2
data:  Gender ~ Admit | Dept
x2 = 0, df = 6, p-value = 1
alternative hypothesis: true value is greater than 0

# gRim gives a slightly more refined output
> gRim::ciTest(as.data.frame(UCBAdmissions),set=~Gender+Admit+Dept)

set: [1] "Gender" "Admit"  "Dept"
Testing Gender _|_ Admit | Dept
Statistic (DEV):    0.000 df: 6 p-value: 1.0000 method: CHISQ

Slice information:
  statistic p.value df Dept
1         0       1  1    A
2         0       1  1    B
3         0       1  1    C
4         0       1  1    D
5         0       1  1    E
6         0       1  1    F
```

Conditional independence for Gaussian distributions

Recall: Marginal and conditional distributions

Split X into two blocks $X = (X_A, X_B)$. Denote

$$\mu = (\mu_A, \mu_B) \qquad \text{and} \qquad \Sigma = \begin{bmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{bmatrix}.$$

Marginal distribution

$X_A \sim N_{|A|}(\mu_A, \Sigma_{AA})$

Conditional distribution

$X_A | X_B = x_B \sim N_{|A|} \left(\mu_A + \Sigma_{AB}\Sigma_{BB}^{-1}(x_B - \mu_B), \Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA} \right)$

- Note that the conditional covariance is constant.

Conditional independence

Independence and conditional independence

$X_i \perp\!\!\!\perp X_j$ if and only if $\Sigma_{ij} = 0$.

$X_i \perp\!\!\!\perp X_j | X_C$ if and only if $\Sigma_{ij} - \Sigma_{i,C}\Sigma_{C,C}^{-1}\Sigma_{C,j} = 0$

Let $R = V \setminus \{i, j\}$. The following are equivalent:

- $X_i \perp\!\!\!\perp X_j | X_R$
- $\Sigma_{ij} - \Sigma_{i,R}\Sigma_{R,R}^{-1}\Sigma_{R,j} = 0$
- $(\Sigma^{-1})_{ij} = 0$

Useful: `https://en.wikipedia.org/wiki/Block_matrix#Block_matrix_inversion`

Part 2: Undirected graphical models

2.1 Graph factorizations

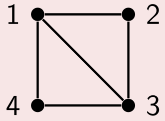
Factorization

G = an undirected graph with nodes $\{1, \ldots, m\}$ and cliques $C_1, \ldots, C_k$.
We say that density $f(\boldsymbol{x})$ **factorizes according to G** if for all $\boldsymbol{x} \in \mathcal{X}$

$$f(\boldsymbol{x}) \;=\; \phi_{C_1}(\boldsymbol{x}_{C_1}) \cdots \phi_{C_k}(\boldsymbol{x}_{C_k}),$$

where $\phi_C(\boldsymbol{x}_C) \geq 0$. (a notion of simplicity)

For example



$$f(\boldsymbol{x}) = \phi_{123}(x_1, x_2, x_3)\phi_{134}(x_1, x_3, x_4).$$

This gives an alternative characterisation of $X_2 \perp\!\!\!\perp X_4 | (X_1, X_3)$.

Hammersley-Clifford theorem

Let $f > 0$ be a dentity function for $\boldsymbol{X} = (X_1, \ldots, X_m)$. Then the following are equivalent:

(F) f factorizes according to G.

(G) $X_A \perp\!\!\!\perp X_B | X_C$ whenever C separates A and B in G.

(P) $X_i \perp\!\!\!\perp X_j | X_{V \setminus \{i,j\}}$ whenever i, j not connected by an edge in G.

The graph represents conditional independence.

Graphical model $\mathcal{M}(G)$

G a graph with m nodes representing a random vector $X = (X_1, \ldots, X_m)$.

$\mathcal{M}(G)$ is the family of all distributions of X that factorize according to G.

By the Hammersley-Clifford theorem this can be equivalently described by conditional independences. (we work with positive distributions only)

Model families that admit suitable factorizations are described in later parts:

1. log-linear models for multivariate discrete data
2. graphical Gaussian models for multivariate Gaussian data.

This drives modelling for more complicated (non-parametric) settings.

Graph analysis

In high-dimensional scenarios we often want to learn the underlying graph from the data.

Although graphs directly visualize the structure this still may be hard to see.

We could be interested in:

- central vertices
- clusters in the network (communities)
- degree distribution

Using `igraph` seems a default option.

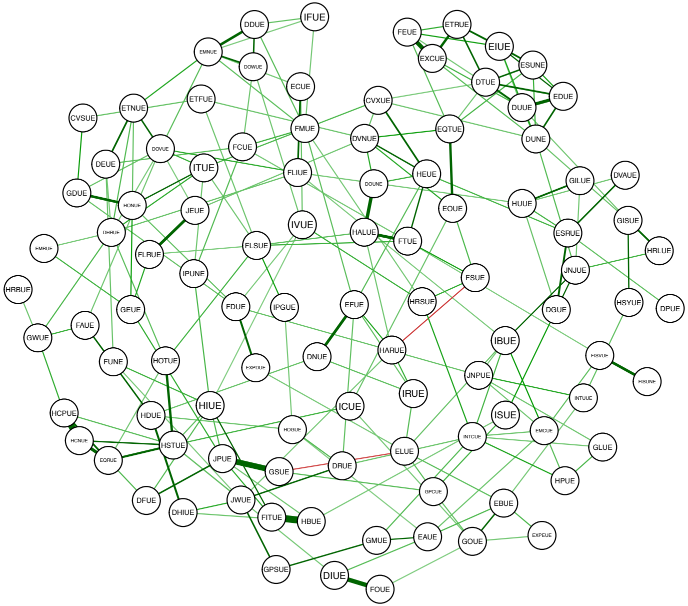
A useful resource: https://kateto.net/networks-r-igraph

Example of a graph analysis

For network analysis copy your network to `igraph`:

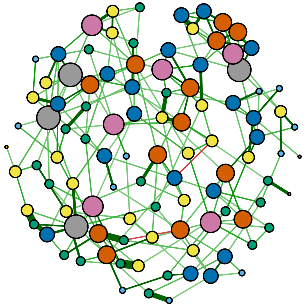
```
> library(sparseIndexTracking); library(xts)
> data(INDEX_2010)
> data <- INDEX_2010$X[,100:200]
> G <- qgraph::qgraph(cor(data),graph="glasso",sampleSize=nrow(data),layout="
   spring",threshold=.07) # takes a minute. we later discuss more scalable
   approaches.
> layout <- G$layout #will be used later
> G <- as.igraph(G)
# The proportion of present edges from all possible edges in the network.
> edge_density(G, loops=F)
[1] 0.04257426
```

qgraph reports some issues but that's irrelevant for our analysis now


IFUE
DDUE
EMNUE
DOWUE
ECUE
FEUE
ETRUE
EIUE
EXCUE
ESUNE
DTUE
EDUE
DUUE
CVXUE
EQTUE
DUNE
ETFUE
ETNUE
CVSUE
FMUE
FCUE
DOVUE
DEUE
ITUE
DVNUE
FLIUE
HEUE
DOUNE
GILUE
DVAUE
GDUE
HONUE
JEUE
IVUE
EOUE
HUUE
GISUE
DHRUE
HALUE
FTUE
ESRUE
HRLUE
EMRUE
FLRUE
FLSUE
FSUE
JNJUE
IPUNE
HRBUE
GEUE
FDUE
IPGUE
EFUE
HRSUE
DGUE
HSYUE
DPUE
FAUE
GWUE
DNUE
HARUE
IBUE
FUNE
HOTUE
EXPDUE
FISVUE
FISUNE
JNPUE
IRUE
INTUUE
HIUE
ICUE
ISUE
HCPUE
HDUE
HOGUE
INTCUE
EMCUE
HCNUE
HSTUE
GLUE
ELUE
JPUE
EQRUE
GSUE
DRUE
HPUE
GPCUE
DFUE
JWUE
DHIUE
EBUE
FITUE
HBUE
EXPEUE
GMUE
EAUE
GOUE
GPSUE
DIUE
FOUE

igraph allows for much more refined analysis

```
# the diameter is small!
> diameter(G, directed=F, weights=NA)
[1] 7
> deg <- degree(G, mode="all")
# check nodes with maximal degree
> which(deg==max(deg))
[1]  3  9 13 83
# plot the graph with node size and color depending on the degree
> plot(G, vertex.size=deg*2,vertex.color=deg,vertex.label=NA,layout=layout)
```



Centrality analysis

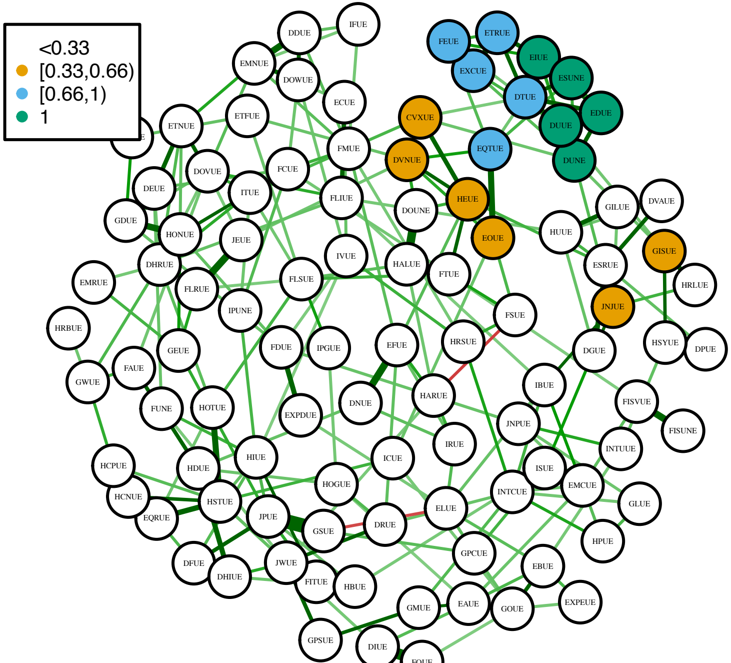
The degree is one measure of centrality of a graph.

Closeness is a centrality based on distance to others in the graph.

```
closeness(G, mode="all", weights=NA)
E(G)$weight <- abs(E(G)$weight)
closeness(G, mode="all")
centr_clo(G, mode="all", normalized=TRUE)
```

Eigenvalue centrality: vertices with high eigenvector centralities are those which are connected to many other vertices which are, in turn, connected to many others (and so on).

```
ec <- eigen_centrality(G, directed=TRUE, weights=NA)
# there are only few vertices with high eigenvalue centrality
plot(1:length(ec$vector),ec$vector,xlab="vertex",ylab="Eigenvalue␣centrality")
plot(G, vertex.color=round(3*ec$vector),vertex.label.cex=.2,layout=layout)
legend(-1.2,1,legend=c("<0.33","[0.33,0.66)","[0.66,1)","1"),col=c(0,
    categorical_pal(3)),pch=19,cex=.4)
```

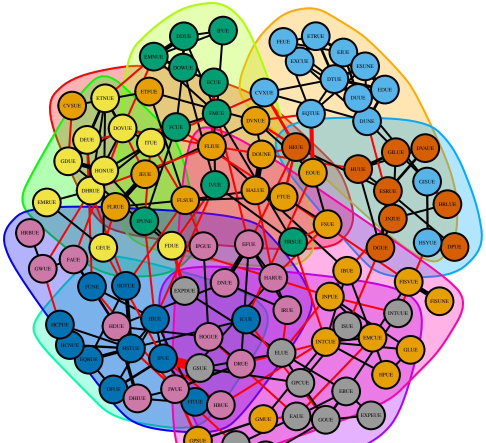


<0.33
● [0.33,0.66)
● [0.66,1)
● 1
DDUE
IFUE
FEUE
ETRUE
EIUE
EMNUE
EXCUE
ESUNE
DOWUE
ECUE
DTUE
CVXUE
EDUE
DUUE
ETNUE
ETFUE
FMUE
EQTUE
DVNUE
DUNE
DOVUE
FCUE
DEUE
ITUE
FLIUE
HEUE
GILUE
DVAUE
DOUNE
GDUE
HONUE
JEUE
EOUE
HUUE
GISUE
IVUE
HALUE
ESRUE
DHRUE
FTUE
EMRUE
FLSUE
FLRUE
HRLUE
IPUNE
FSUE
JNJUE
HRBUE
FDUE
IPGUE
EFUE
HRSUE
HSYUE
DPUE
GEUE
DGUE
FAUE
GWUE
IBUE
HARUE
DNUE
FISVUE
EXPDUE
FUNE
HOTUE
JNPUE
FISUNE
IRUE
INTUUE
HIUE
HCPUE
ICUE
ISUE
HDUE
HOGUE
EMCUE
INTCUE
HCNUE
HSTUE
GLUE
ELUE
JPUE
EQRUE
GSUE
DRUE
HPUE
GPCUE
DFUE
JWUE
EBUE
DHIUE
FITUE
HBUE
EXPEUE
GMUE
EAUE
GOUE
GPSUE
DIUE

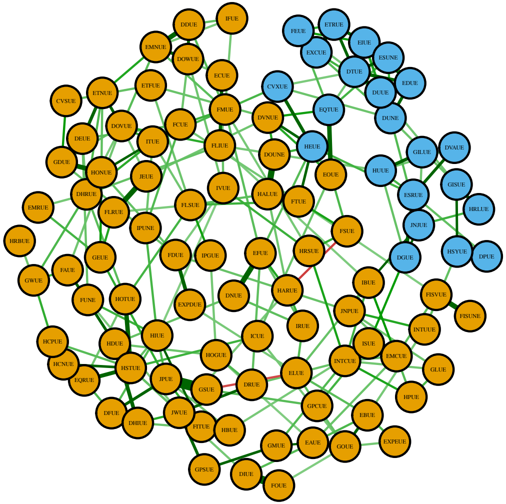
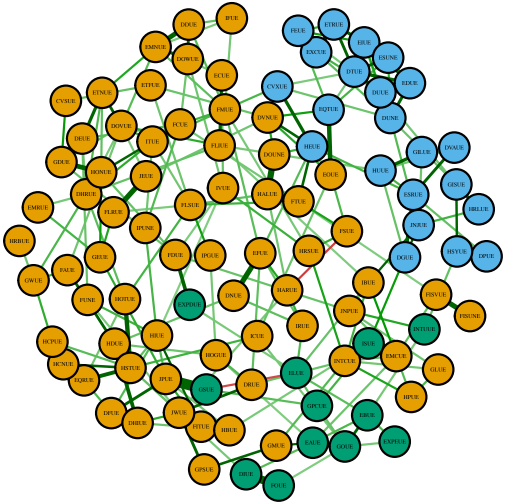
Communities

Many networks consist of modules which are densely connected themselves but sparsely connected to other modules.

```r
ceb <- cluster_edge_betweenness(G,directed=FALSE)
plot(ceb, G,vertex.label.cex=.2,layout=layout)
```



```
#coarser information possible
plot(G,vertex.color=cut_at(ceb,2),vertex.label.cex=.2,layout=layout)
```

2.2 Gaussian graphical models

The Gaussian case

For a Gaussian distribution in $\mathcal{M}(G)$:

The non-edges of G correspond to conditional independences $X_i \perp\!\!\!\perp X_j | X_{V\setminus\{i,j\}}$ or equivalently $\mathbf{K_{ij} = 0}$.

- Indeed, $\rho_{ij|V\setminus\{i,j\}} = -\frac{K_{ij}}{\sqrt{K_{ii}K_{jj}}}$.

Two main estimation problems:

Consider an *iid* sample $X^1, \dots, X^n$ from $\mathcal{M}(G)$.

The partial correlation of the sample will have no zeros.

(i) Estimate Σ for a fixed graph G.

(ii) Estimate the graph in a statistically meaningful way.

On concentration of the covariance matrix

```
> K <- matrix(c(1,0,1/2,0,1,1/2,1/2,1/2,1),3,3); Sig <- solve(K)
> set.seed(1)
> X10 <- mvrnorm(10,c(0,0,0),Sig); S10 <- cov(X10)
> X100 <- mvrnorm(100,c(0,0,0),Sig); S100 <- cov(X100)
> X1000 <- mvrnorm(1000,c(0,0,0),Sig); S1000 <- cov(X1000)
> solve(S10); solve(S100); solve(S1000)
          [,1]     [,2]      [,3]
[1,] 1.9379597 1.616762 0.8595338
[2,] 1.6167622 4.076235 2.1360457
[3,] 0.8595338 2.136046 1.6042403
           [,1]       [,2]      [,3]
[1,] 1.04792019 0.08406772 0.5781926
[2,] 0.08406772 0.93313405 0.3880432
[3,] 0.57819258 0.38804318 1.1148584
            [,1]        [,2]      [,3]
[1,]  0.88842341 -0.02029737 0.4710935
[2,] -0.02029737  0.90492134 0.4558757
[3,]  0.47109348  0.45587568 0.9628081
```

Without regularization estimating a covariance matrix is a hard problem.

Estimating the graph seems easier! (at least in this favorable case)

The Gaussian likelihood function

The sample covariance matrix of the sample $X^1, \ldots, X^n$ is

$$S = \frac{1}{n}\sum_{i=1}^{n}(X^i - \bar{X})(X^i - \bar{X})^T.$$

The log-likelihood is

$$\log L(\mu, K) \;=\; \tfrac{n}{2}\log\det K - \tfrac{n}{2}\mathrm{trace}(KS) - \tfrac{n}{2}(\bar{X}-\mu)^T K(\bar{X}-\mu).$$

For fixed K we get $\hat{\mu} = \bar{X}$ giving the profile likelihood

$$\log L(\hat{\mu}, K) \;=\; \tfrac{n}{2}\log\det K - \tfrac{n}{2}\mathrm{trace}(KS).$$

Maximizing the likelihood over $\mathcal{M}(G)$

$\mathcal{M}(G)$ consists of PD matrices K such that $K_{ij} = 0$ for $ij \notin E$.

Optimizing $\log L(\hat{\mu}, K)$ over $\mathcal{M}(G)$ is a convex optimization problem.

The MLE is the unique point $\hat{\Sigma}$ such that:

(i) $\hat{\Sigma}_{CC} = S_{CC}$ for all cliques C,

(ii) $\hat{K}_{ij} = 0$ for all $ij \notin E$.

Maximization

Typically numerically using a block coordinate-descent scheme (`ggmfit`).

The deviance

Gives the likelihood ratio statistic to test against the unconstrained model.

Using gRbase, gRim, RBGL:

```
> data(carcass)
# generate model: use plot(gen.carc) to see the graph
> glist <- list(c("Fat11","Fat12","Meat12", "Meat13"), c("Fat11","Fat12","
    Fat13","LeanMeat"),c("Meat11","Meat12","Meat13"),c("Meat11","Fat13","
    LeanMeat"))
#cmod specifies graphical Gaussian model ('c' for continuous)
> gen.carc <- cmod(glist,data=carcass,fit=TRUE)
# the deviance is large
> gen.carc$fitinfo$dev; 1-pchisq(gen.carc$fitinfo$dev,6)
[1] 19.78537
[1] 0.003023752
> qgraph::qgraph(-cov2cor(gen.carc$fitinfo$K))
# more direct way that also gives more flexibility
> carcfit <- ggmfit(S.carc,n=nrow(carcass),glist,eps = 1e-14,iter=10000)
```

Coordinate descent

- Coordinate descent is a classic optimisation technique for multidimensional functions; the idea is to optimise with respect to one variable at a time, holding the others constant, cycling through all variables in some given order repeatedly, until convergence.
- Convex functions are convex in each coordinate.
- This gives the global minimum under some additional conditions.

Old idea in statistics. . .

- Iterative Proportional Fitting for log-linear models dates back to Bartlett (1935), Csiszár (1970s).
- Gaussian graphical models: Dempster (1972), Wermuth and Scheidt (1977), Speed and Kiiveri (1982).

When coordinate descent works?

Your intuition is correct

- If f is continuously differentiable and strictly convex then coordinate descent converges to the global optimum.

Separability condition

- Suppose $f(\beta) = g(\beta) + \sum_{j=1}^{p} h_j(\beta_j)$
  - $g : \mathbb{R}^p \to \mathbb{R}$ differentiable and convex
  - $h_j : \mathbb{R} \to \mathbb{R}$ are convex
- Tseng (1988,2001): in this scenario coordinate descent converges to the global optimum

Model selection methods

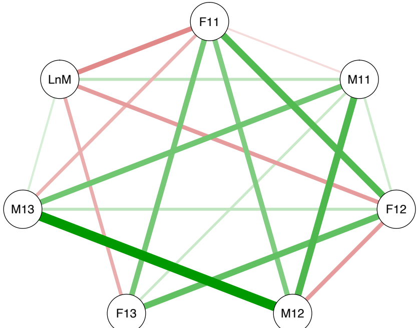
How to learn the graph:

- Stepwise methods,
- Thresholding,
- Convex optimization,
- Simultaneous p-values. (not discussed here)

Stepwise methods

The `stepwise` function in `gRim` performs stepwise model selection based on a variety of criteria (AIC, BIC, etc)

```
sat.carc <- cmod(~.^.,data=carcass); n <- nrow(carcass)
test.carc <- stepwise(sat.carc,details=1,criterion="aic",type="decomposable",k
    =log(n))
# plot(test.carc) or using the qgraph package
Sigma.hat <- solve(test.carc$fitinfo$K)
qgraph::qgraph(Sigma.hat,graph="pcor")
```



Thresholding

A simple and natural method is to threshold entries of the sample concentration matrix.

Some obvious issues:

- The sample covariance matrix S is not invertible if $n < m$.
- Thresholded S^{-1} may not be positive definite.
- The approach is not statistically sound.

Thresholding

Consider example on Slide 40

```
> data(carcass); S.carc <- cov(carcass); PC.carc <- cov2pcor(S.carc)
> threshold <- 0.1
# define the thresholded version of PC.carc
> PC.thresh <- PC.carc; PC.thresh[abs(PC.thresh)<threshold] <- 0
> qgraph::qgraph(PC.thresh) # visually similar to the stepwise case
# compare this with the stepwise selection graph
> cmod(PC.thresh,data=carcass)
Model: A cModel with 7 variables
 -2logL    :       11384.93 mdim :   23 aic :      11430.93
 ideviance :        2456.30 idf  :   16 bic :      11519.27
 deviance  :          17.48 df   :    5

 >  test.carc
Model: A cModel with 7 variables
 -2logL    :       11370.74 mdim :   25 aic :      11420.74
 ideviance :        2470.49 idf  :   18 bic :      11516.76
 deviance  :           3.29 df   :    3
```

Learning the graph in high-dimension

Graphical lasso

If the dimension m is large then the number of possible models is too high.

Following the same idea as in the lasso regression we maximize

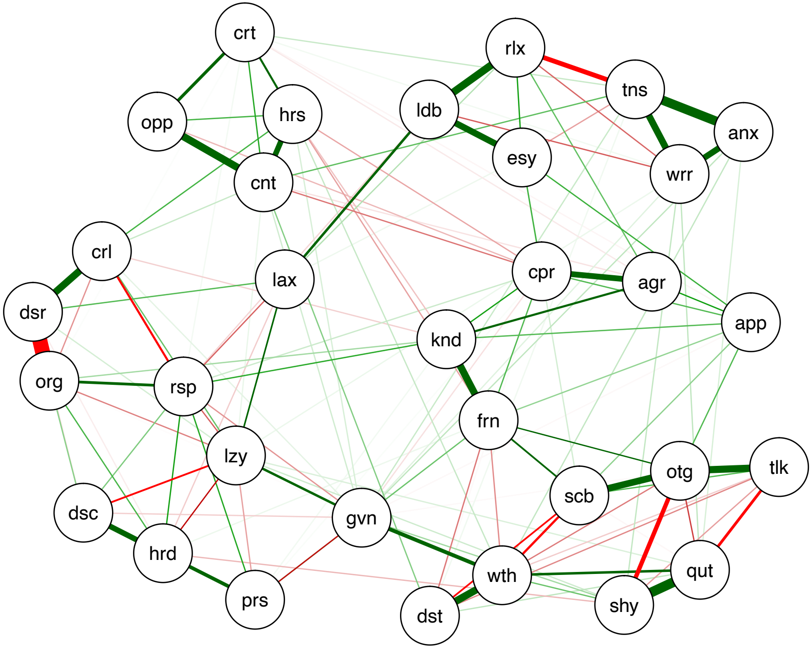
$$L_{\mathrm{pen}}(K, \hat{\mu}) \;=\; \log\det(K) - \mathrm{trace}(SK) - \lambda \|K\|_1.$$

See the package `glasso` and `EBICglasso` (finds an optimal λ).

Example: Personality traits

```
> d <- read.table("./data/personality0.txt"); S <- cor(d)
# the partial correlation graph of the data is dense
> qgraph::qgraph(S,graph="pcor")
# the glasso graph is much sparser
> qgraph::qgraph(S,graph="glasso",sampleSize=nrow(d),layout="spring")
```

The estimated graph has 137 edges compared to 496 of the full graph.


crt
rlx
tns
opp
hrs
ldb
anx
esy
wrr
cnt
crl
lax
cpr
agr
dsr
app
knd
org
rsp
frn
lzy
tlk
otg
scb
dsc
gvn
hrd
qut
wth
prs
shy
dst

EBIC graphical lasso

EBIC provides a data-driven way to choose the penalty parameter λ in graphical lasso.

1. Run the graphical lasso for a range of penalty parameters to get a preselected list of models.
2. For the models in the above list maximize the EBIC for $\gamma \in [0, 1]$:

$$\mathrm{BIC}_\gamma \; = \; -2\ell_n(\hat{K}) + |E| \log n + 4|E|\gamma \log p.$$

Consistency of EBIC has been shown under some technical assumptions (Drton, Foygel, 2010).

Simulations show that consistency holds under much weaker conditions.

The procedure is much quicker than CV.

Main papers on graphical Lasso

Basic asymptotic theory studied in:

Yuan & Lin (2007). Model selection and estimation in the Gaussian graphical model. Biometrika, 94(1), 19-35.

Alternative optimization methods + the name:

Friedman, Hastie & Tibshirani (2008). Sparse inverse covariance estimation with the graphical lasso. Biostatistics, 9(3), 432-441.

d'Aspremont, Banerjee & El Ghaoui (2008). First-order methods for sparse covariance selection. SIAM Journal on Matrix Analysis and Applications, 30(1), 56-66.

Non-asymptotic theory studied in:

Ravikumar, Wainwright, Raskutti & Yu (2011). High-dimensional covariance estimation by minimizing ℓ1-penalized log-determinant divergence. Electronic Journal of Statistics, 5, 935-980.

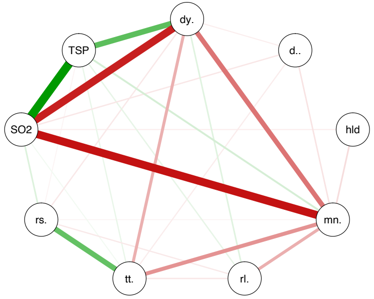
Under some extra conditions we get: $\|\hat{K} - K^*\| = \mathcal{O}\left(\sqrt{\min\{d^2, s+m\}\frac{\log m}{n}}\right)$ and model selection consistency.

We assume sparsity!

The sparsity or bounded degree assumption are not always justified. Consider the data set from Slide **??**.

```r
data(milan.mort,package="SemiPar")
qgraph::qgraph(cor(milan.mort), layout="groups", graph="glasso",sampleSize=
    nrow(milan.mort)) # outputs some warnings
# traditional model selection
sat.mort <- cmod(~.^.,data=milan.mort)
test.mort <- stepwise(sat.mort,details=1,"test")
qgraph::qgraph(-cov2cor(test.mort$fitinfo$K)) # essentially the same graph
```

day.num	number of days since 31st December, 1979
day.of.week	1=Monday,2=Tuesday,3=Wednesday,4=Thursday, 5=Friday,6=Saturday,7=Sunday.
holiday	indicator of public holiday: 1=public holiday, 0=otherwise.
mean.temp	mean daily temperature in degrees Celcius.
rel.humid	relative humidity.
tot.mort	total number of deaths.
resp.mort	total number of respiratory deaths.
SO2	measure of sulphur dioxide level in ambient air.
TSP	total suspended particles in ambient air.



(Note on TSP – dy. and TSP – mn.)

CLIME estimator

Alternative to graphical lasso:

Instead of maximizing the log-likelihood, CLIME estimator finds a sparse estimate of the precision matrix by solving

$$\hat{K} \;=\; \arg\min \|K\|_1 \quad \text{subject to } \|SK - \mathbb{I}_p\|_\infty \leq \lambda.$$

- similar consistency guarantees as graphical lasso,
- computationally attractive,
- implemented in the `R` package `clime`.

2.3 Log-linear models

Hierarchical models

State space $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_m, \quad |\mathcal{X}_i| < +\infty$.

Simplicial complex: $\mathcal{C}_0$ set of subsets of $\{1, \dots, m\}$ such that

- $\{i\} \in \mathcal{C}_0$ for all $i = 1, \dots, m$,
- $C \in \mathcal{C}_0$ and $A \subset C$ then $A \in \mathcal{C}_0$

Generators: Denote by $\mathcal{C}$ the maximal elements of $\mathcal{C}_0$.

The model: $p(x) \;=\; \prod_{C \in \mathcal{C}} \phi_C(x_C), \quad$ where $\phi_C > 0$.

Main effect model: $\mathcal{C} = \{\{1\}, \dots, \{m\}\}$.

Pairwise interaction model: $\mathcal{C} = \{$all pairs$\}$.

- more generally: edges of a graph with m nodes.

Graphical and non-graphical models

(Graphical) hierarchical models: $\mathcal{C}$ be the set of maximal cliques of G.

Hammersley-Clifford theorem (Slide 22) gives equivalent description in terms of conditional independences.

Not every hierarchical model is graphical.

- e.g. no three-way interaction models with $\mathcal{C} = \{\{1,2\},\{1,3\},\{2,3\}\}$.

Not every graphical model is a pairwise interaction model.

- e.g. $\mathcal{C} = \{\{1,2,3\}\}$.

Two remarks:

- Both pairwise interaction models and graphical models are represented by graphs but the graphs encode different information.
- Every pairwise interaction model with underlying graph G is contained in graphical model over G. (so the conditional independence information is retained)

Fitting log-linear models in R

The function `loglin()` fits general log-linear models. We focus here on the ones represented by graphs, that is, graphical or pairwise.

The gRim package has a function `dmod()` to define and fit hierarchical log-linear models for a fixed set of generators.

```
> data(lizard)
> mliz <- dmod(list(c("species","height"),c("species","diam")),data=lizard)
> mliz

Model: A dModel with 3 variables
 -2logL     :         1604.43 mdim :    5 aic :       1614.43
 ideviance  :           23.01 idf  :    2 bic :       1634.49
 deviance   :            2.03 df   :    2
```

On `data(lizard)`: Behaviour characteristics of 409 lizards were recorded: species (`anoli`, `dist`), perch diameter (<=4, >4), and perch height (>`4.75`,<=`4.75`).

Testing conditional independence

If the data is given in form of a contingency table it is convinient to use gRim's ciTest_table.

```
> ciTest_table(lizard,set=c("height","diam","species"))
Testing height _|_ diam | species
Statistic (DEV):    2.026 df: 2 p-value: 0.3632 method: CHISQ
Slice information:
  statistic p.value df species
1     0.178  0.6731  1   anoli
2     1.848  0.1741  1    dist


> ciTest_table(lizard,set=c("diam","species","height"))
Testing diam _|_ species | height
Statistic (DEV):   14.024 df: 2 p-value: 0.0009 method: CHISQ
Slice information:
  statistic   p.value df height
1     2.903 0.0884377  1  >4.75
2    11.122 0.0008533  1 <=4.75
```

Notice the df calculation, which in general may be complicated.

Model selection

In general graphical lasso approach is hard to perform.

- c.f. Ling-Loh, Wainwright, *Structure estimation for discrete graphical models: Generalized covariance matrices and their inverses*

Typical strategies:

(i) using conditional independence tests,

(ii) heuristic search to optimize some information criterion,

(iii) Bayesian methods involving MCMC.

Stepwise methods

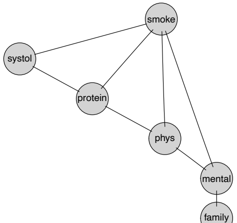
For stepwise methods of type (ii) typically the penalized likelihood criteria like BIC or AIC are used. For a set of models M_j the criterion we minimize is:

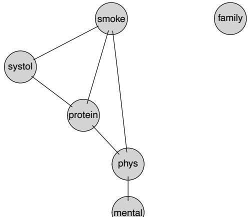
$$-2 \log L(j) + k \dim M_j,$$

where $k = 2$ for AIC and $k = \log n$ for BIC.

Stepwise methods either start with the full graph or the empty graph.

```
> data(reinis); m.init <- dmod(~.^., data=reinis)
> m.reinis <- stepwise(m.init) # AIC criterion
> m.reinis.2 <- stepwise(m.init,k=log(sum(reinis))) # BIC criterion
> plot(m.reinis); plot(m.reinis.2)
```





Modelling large data sets

With > 10 variables stepwise methods are computationally prohibitive.

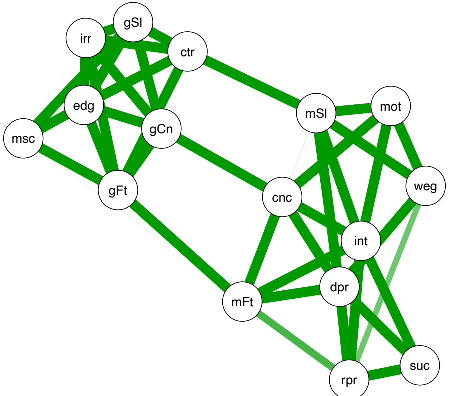
Possible strategies:

1. Use MCMC (`BDgraph` package)
2. Focus on decomposable models (`gRapHD` package).
3. Rely on simpler models (e.g. pairwise interaction models).
4. Restrict to very simple graphs (e.g. trees).

Bayesian Structure Learning using Birth-Death MCMC

bdgraph.mpl in the package BDgraph can be used for discrete data. Consider the data table X defined in Slide 59.

```
> install.packages("BDgraph"); library(BDgraph)
> X.fit <- bdgraph.mpl(X,method="dgm-binary",g.prior=.5)
> A <- X.fit$p_links+t(X.fit$p_links)
> qgraph::qgraph(A,layout="spring")
```



Binary Ising model

Consider a log-linear model with only pairwise interactions.

If $\mathcal{X} = \{-1, 1\}^m$ then

$$f(x) \;=\; \tfrac{1}{Z(h,B)} \exp\left(\sum_i h_i x_i + \sum_{i<j} \beta_{ij} x_i x_j\right),$$

where $h \in \mathbb{R}^m$ and $B = [\beta_{ij}]$ has zeros on the diagonal.

Likelihood is hard to handle because $Z(h, B)$ is intractable if m is large.

This will be our first encounter with pseudo-likelihood methods.

Pseudo-likelihood approach

Logistic regression

Denoting $\eta_k = h_k + \sum_{i \neq k} \beta_{ik} x_i$, the full conditional distributions satisfy

$$\log p(x_k | x_{-k}) = \eta_k x_k - \log(e^{-\eta_k} + e^{\eta_k})$$

and so, if $p = p(1|x_{-k})$, then

$$\log \frac{p}{1-p} \;=\; 2\eta_k \;=\; 2h_k + \sum_{i \neq k} 2\beta_{ik} x_i$$

ℓ_1-regularized logistic regression (Ravikumar et al, 2010)

Having observed $x^{(1)}, \ldots, x^{(n)}$, to learn the support of B, we can optimize for each k the logistic regression given the data. Alternatively we can maximize

$$\sum^{n} \sum \log \quad (\quad^{(i)} | \quad^{(i)}) \qquad \lambda \| B \|$$

Using IsingFit

```
> install.packages("IsingFit"); library(IsingFit)
> ncsdata=read.table(file="./data/DepressionAnxiety.txt")
> colnames(ncsdata)=c("depr", "inte", "weig", "mSle", "moto", "mFat", "repr",
    "conc", "suic", "anxi", "even", "ctrl", "edge", "gFat", "irri", "gCon", "
    musc", "gSle") #Define variable names.
# remove two variables that are perfectly correlated with each other in the
    sample
> X <- (ncsdata[,-(10:11)])
# run the high-dimensional Ising model selection problem
> Res <- IsingFit(X,gamma = 0.5, plot=FALSE)
# compare with the correlation network
> lay <-averageLayout(Res$weiadj,cor(X), layout = "spring", repulsion = 1)
> qgraph(cor(X),layout=lay,labels=colnames(X))
> qgraph(Res$weiadj, layout = lay)
# both graphs appear on the next slide
```

For a discussion of this dataset see:

Borsboom and Cramer, Network analysis: an integrative approach to the structure of psychopathology, Annual review of clinical psychology, 9 (2013).

Two psychological disorders

About the study:

National Comorbidity Survey Replication (NCS-R data)

9282 observations of 18 binary variables such as:
`depr` (Depressed mood), `inte` (Loss of interest), etc

These are symptoms related to two disorders:
major depression and generalized anxiety disorder.

Bridge variables: sleep problems, fatigue, and concentration problems.

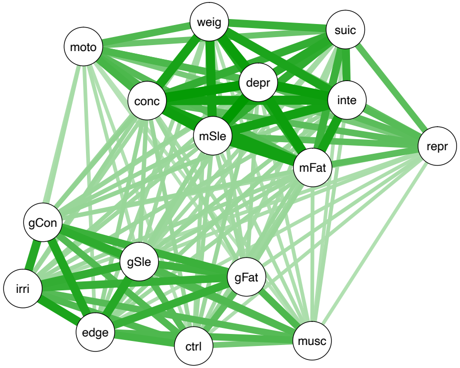
Two psychological disorders, continued

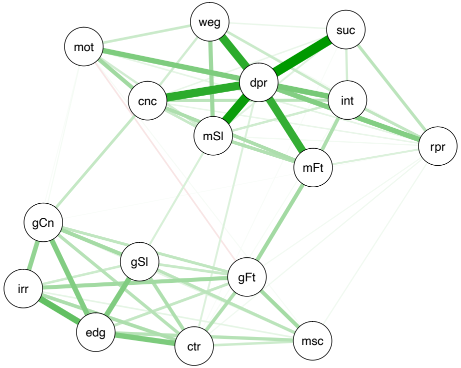
About the data:

Sparse contingency table: 872/65536 nonzero cells.

5667 out of 9282 respondents recorded no symptoms.

two variables perfectly correlated with each other and other seven variables.





Note that basically all edges on the right are green.

Part 3: Beyond the standard set-up

The "standard" set-up is not enough

The standard set-up

- Gaussian or log-linear models
- Likelihood or penalized likelihood inference.

Challenges

- Heavy-tailed or multimodal and asymmetric data.
- Other types of misspecification.
- MLE computationally intractable.
- Models for mixed data needed.

Useful `R` packages

`huge`: High-Dimensional Undirected Graph Estimation
`smart`: Sparse Multivariate Analysis via Rank Transformation

M-estimation

The MLE has some deficiencies (e.g. robustness, computational cost).

M-estimators for a sample $X^{(1)}, \dots, X^{(n)}$: A large family of alternatives.

1. $m_\theta : \mathcal{X} \to \mathbb{R}$ a known function
2. $\hat{\theta}$ maximizes $\frac{1}{n}\sum_{i=1}^{n} m_\theta(X^{(i)})$.

Often M-estimators result in consistent estimators, which are asymptocially normal but in general less efficient.

Examples we saw:

- Least Squares Estimator for Gaussian data is the the MLE solution. For non-gaussian data it is not but it provides an estimator with good statistical properties.
- Penalized log-likelihood functions lead to M-estimators.
- For the Ising model we can use conditional distributions to define a pseudo-likelihood.

3.1 Some non-parametric approaches

Pairwise interaction graphs

We first focus on models for which the density admits factorization

$$\begin{aligned} p(x) &= \frac{1}{Z(\psi)} \exp\Big(\underbrace{\sum_{ij\in E} \psi_{ij}(x_i, x_j)}_{\text{interaction terms}} + \underbrace{\sum_{i\in V} \psi_i(x_i)}_{\text{main effects}} \Big) \\ &= \prod_{ij\in E} \phi_{ij}(x_i, x_j) \prod_{i\in V} \phi_i(x_i). \end{aligned}$$

Main cases considered in the literature:

Semiparametric Exponential Family Graphical Models

- $\psi_{ij}(x_i, x_j) = \beta_{ij} x_i x_j$, inference with ℓ_1-regularized pseudo-likelihood.

Gaussian Copula Models

- $\psi_{ij}(x_i, x_j) = \beta_{ij} f_i(x_i) f_j(x_j)$ where f_i are monotone functions.

By Hammersley-Clifford, in both cases $X_i \perp\!\!\!\perp X_j | X_{V\setminus\{i,j\}}$ if and only if $\beta_{ij} = 0$.

Semiparametric Exponential Family Graphical Models

Yang, Ning, & Liu (2018)

Semiparametric approach: $\psi_{ij}(x_i, x_j) = \beta_{ij} x_i x_j$ and $\psi_i(x_i)$'s kept general.

β_{ij}'s are the parameters of interest and ψ_i are nuisance parameters.

Nuisance-free loss function

If X, X' independent copies then

$$\mathbb{P}\big(X = x, X' = x' | X_{-k} = x_{-k}, X'_{-k} = x'_{-k}, \{X_k, X'_k\} = \{x_k, x'_k\}\big)$$

does not depend on the normalizing constant nor on ψ_k.

This can be used to construct a pseudo-likelihood that depends only on B.

Adding an $\ell 1$-penalty term assure sparse solutions.

Currently there is no `R` package; stay tuned.

Nonparanormal distributions

$(X_1, \ldots, X_m)$ has nonparanormal distribution if $(f_1(X_1), \ldots, f_m(X_m))$ is Gaussian for some monotone functions $f_1, \ldots, f_m$.

The functions f_i are treated again as nuisance parameters. The correlation matrix is estimated directly using rank correlation statistics:

We can estimate the correlation matrix of $f(X)$ without knowing f!!

If $\tau_{ij} = \mathrm{cor}(\mathrm{sgn}(X_i - X_i'), \mathrm{sgn}(X_j - X_j'))$ is the Kendall's tau statistic then

$$\Sigma^0_{ij} := \mathrm{cor}(f(X_i), f(X_j)) \; = \; \sin\left(\tfrac{\pi}{2}\tau_{ij}\right).$$

- τ_{ij}'s are invariant under monotone transformations on X_i's.
- $f(X)$ is Gaussian and in this case the formula follows by standard results.
- the inverse Σ^0 holds the conditional independence structure of X.
  $(f_i(X_i) \perp\!\!\!\perp f_j(X_j) | \mathrm{rest} \quad \Leftrightarrow \quad \beta_{ij} = 0 \quad \Leftrightarrow \quad X_i \perp\!\!\!\perp X_j | \mathrm{rest})$

Estimating nonparanormal graphical models

Compute Kendall's tau coefficients from the data (c.f. Slide 10)

$$\hat{\tau}_{ij} = \frac{1}{\binom{n}{2}} \sum_{1 \le t \le t' \le n} \mathrm{sign}(X_{it} - X_{it'})\mathrm{sign}(X_{jt} - X_{jt'}).$$

Define $\hat{S}^{\tau} = \sin(\frac{\pi}{2}\hat{\tau}_{ij})$.

Concentration analysis based on U-statistics shows that

$$\max_{ij} |\hat{S}^{\tau}_{ij} - \Sigma^0_{ij}| = \mathcal{O}\left(\sqrt{\frac{\log(mn)}{n}}\right).$$

Based on $\hat{S}^{\tau}$ we can now employ any standard Gaussian procedure to learn the graph, e.g., graphical lasso.

Application using huge package

Stock market data: closing prices from all stocks in the S&P 500 for all the days that the market was open between Jan 1, 2003 and Jan 1, 2008.

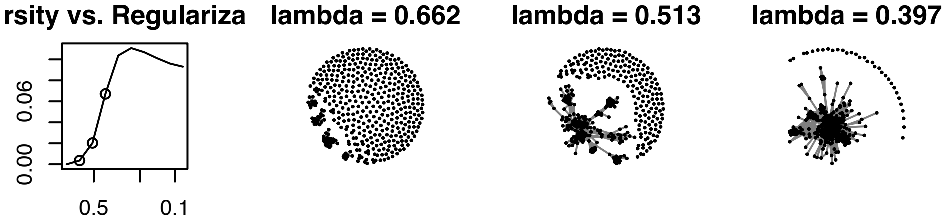
```
> library(huge); data(stockdata) # Load the data
> x = log(stockdata$data[2:1258,]/stockdata$data[1:1257,]) # Preprocessing
> colnames(x) <- stockdata$info[,1]
> x.npn = huge.npn(x, npn.func="skeptic") # Nonparanormal + Kendal's tau
> out.npn = huge(x.npn,method = "glasso")
> plot(out.npn)
# plot the graph corresponding to lambda=0.397
> Khat <- (out.npn$icov[[15]]+t(out.npn$icov[[15]]))/2; colnames(Khat) <-
    rownames(Khat) <- colnames(x)
> dev.off(); qgraph::qgraph(-cov2cor(Khat),layout="spring")
^^I
```

This is a rather large example to be handled by qgraph directly.

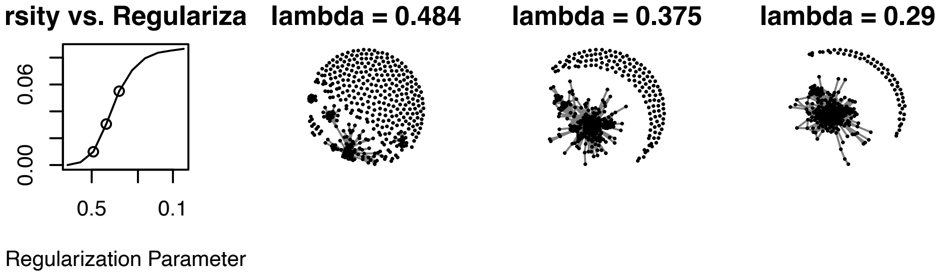
```
> qgraph(x.npn,graph="glasso",sampleSize=nrow(x),layout="spring")
^^I
```

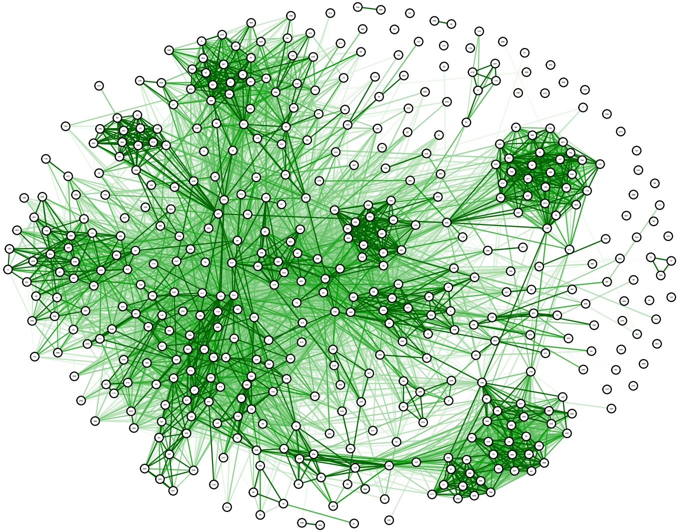
Zhao, T., Liu, H., Roeder, K., Lafferty, J., & Wasserman, L. (2012). The huge package for high-dimensional undirected graph estimation in R. Journal of Machine Learning Research.

The non-paranormal approach gives:



For comparison a direct Gaussian approach gives:





Note that all edges are green.

(for visualizing such big networks may be worthwhile to learn the `visNetwork` package)