Graphical Models with R 3rd talk: Gaussian Graphical Models

Presentation · July 2018

CITATIONS
0

READS
16

1 author:

Dhafer Malouche
University of Carthage
77 PUBLICATIONS **268** CITATIONS

SEE PROFILE

Graphical Models with R

3rd talk: Gaussian Graphical Models

Dhafer Malouche

essai.academia.edu/DhaferMalouche

Table of contents

Theory

Gaussian Graphical Models

Multivariate Gaussian Distribution

V a finite set

$X_V = (X_v,\, v \in V)' \in \mathbb{R}^{|V|}$ a r.v. with density, $\forall\, x_V \in \mathbb{R}^{|V|}$:

$$f(x_v) = \frac{|K|^{1/2}}{(2\pi)^{|V|/2}} \exp\left(-\frac{1}{2}(x_V - \mu)' K (x_V - \mu)\right)$$

where $K = (k_{uv}) \in \mathbb{R}^{|V| \times |V|}$ is the precision matrix (K^{-1} is the covariance-matrix) and μ is the mean vector.

$$\mathrm{Var}(X_V) = K^{-1} \text{ and } \mathbb{E}(X_V) = \mu$$

We denote $X_V \sim \mathcal{N}(\mu, K^{-1})$.

Partial correlation matrix

$$\rho^* = (\rho^*_{uv}) \in [-1, 1]^{|V| \times |V|}$$

where

$$\rho^*_{uv} = -\frac{k_{uv}}{\sqrt{k_{uu} k_{vv}}}$$

$$\rho^*_{uv} = 0 \iff u \perp\!\!\!\perp v \mid V \setminus \{u, v\}$$

Partial correlation matrix

$$\rho^* = (\rho^*_{uv}) \in [-1, 1]^{|V| \times |V|}$$

where

$$\rho^*_{uv} = -\frac{k_{uv}}{\sqrt{k_{uu} k_{vv}}}$$

$$\rho^*_{uv} = 0 \iff u \perp\!\!\!\perp v \mid V \setminus \{u, v\}$$

Graphical models

$\mathcal{G} = (V, E)$ is a Graphical model associated to the distribution $\mathbf{p}$ of X_V, if

$$u \not\sim v \iff u \perp\!\!\!\perp v \mid V \setminus \{u, v\}$$

$(X_V \sim \mathbf{p}, \mathcal{G})$ is called a Undirected Gaussian Graphical Models (UGGM).

$$\forall (A, B, S) \subset V, \text{ if } S \text{ separates } A \text{ and } B \Rightarrow A \perp\!\!\!\perp B \mid S$$

Model fitting

Likelihood

- Data: $(X_V^1, \ldots, X_V^n)$ is an $n-$sample with the same distribution as $X_V \sim \mathcal{N}(\mu, K^{-1})$.
- Empirical covariance matrix: $S = W/n$ where

$$W = \sum_{k=1}^{n} (X_V^k - \bar{X}_V)(X_V^k - \bar{X}_V)'$$

where

$$\bar{X}_V = \frac{1}{n} \sum_{k=1}^{n} X_V^k$$

- Log-Likelihood:

$$L(K, \mu) = \frac{n}{2} \log|K| - \frac{n}{2} \mathrm{tr}(KS) - \frac{n}{2} (\bar{X}_V - \mu)'(\bar{X}_V - \mu)$$

ML Estimation

- For fixed K, $L(K, \mu)$ is maximized for $\widehat{\mu} = \overline{X}_V$.
- MLE of K is the maximum of

$$K \longmapsto L(K, \widehat{\mu}) = \frac{n}{2} \log|K| - \frac{n}{2} \mathrm{tr}(KS)$$

- For a $(X_V \sim \mathbf{p}, \mathcal{G})$ (UGGM) and $\mathcal{C} = \{C_1, \ldots, C_Q\}$ the set of cliques of $\mathcal{G}$, $\widehat{K}$, satisfied

$$\widehat{\Sigma}_{C_l C_l} = S_{C_l C_l} \quad \forall\, l = 1, \ldots, Q$$

  where $\widehat{\Sigma} = \widehat{K}^{-1}$.

IPS algorithm, $\mathcal{G}$ and $\mathcal{C} = \{C_1, \ldots, C_Q\}$

1. $k = 0$, K^k a positive definite matrix satisfying the condition of the model
2. Let, $j = 1$
3. $C = C_j$ and $B = V \setminus C$, solve (E unknown)

$$\begin{pmatrix} S_{CC} & S_{CB} \\ S_{BC} & S_{BB} \end{pmatrix} = \begin{pmatrix} K^k_{CC} + E & K^k_{CB} \\ K^k_{BC} & K^k_{BB} \end{pmatrix}^{-1}$$

   Solution: $E = S_{CC}^{-1} - \left(K^k_{CC} - K^k_{CB}(K^k_{BB})^{-1}K^k_{BC}\right)$ where S_{CC}^{-1} and $(K^k_{BB})^{-1}$ are resp. the inverse of S_{CC} and K^k_{CC}
4. Let K^{k+1}, s.t

$$K^k_{CC} \leftarrow S_{CC}^{-1} + (K^k_{CB})(K^k_{BB})^{-1}(K^k_{BC})$$

5. $j \leftarrow j + 1$, if $j = Q$, then $k \leftarrow k + 1$, else go to 3.
6. Stop until convergence

Example, Generating data

```
> ## Generate a sample of Gaussian distribution
> set.seed(123)
>  A=t(matrix(c(1,2,0,0,1,-1,0,0,1),3,3))
> dt=rnormDag(n=100,A,rep(1,3))
> S=cov.wt(x = dt,method = "ML")$cov
> S
          [,1]       [,2]       [,3]
[1,]  8.939863 -3.9033137 -1.9529918
[2,] -3.903314  1.8745730  0.9210539
[3,] -1.952992  0.9210539  0.8932473
> K0=matrix(c(2,1,1,1,2,0,1,0,2),3,3)
> det(K0)
[1] 4
```

Example, Iteration 1

```
> ## iter 1
> K1=K0
> ## Update 12
> K1[1:2,1:2]=solve(S[1:2,1:2])+
+   K0[1:2,3]%*%solve(K0[3,3])%*%K0[3,1:2]
> K1
         [,1]     [,2] [,3]
[1,] 1.731191 2.563638    1
[2,] 2.563638 5.871567    0
[3,] 1.000000 0.000000    2
> ## 13
> K1[c(1,3),c(1,3)]=solve(S[c(1,3),c(1,3)])+
+   K1[c(1,3),2]%*%solve(K1[2,2])%*%K1[2,c(1,3)]
> K1
          [,1]     [,2]      [,3]
[1,] 1.3334723 2.563638 0.4681937
[2,] 2.5636375 5.871567 0.0000000
[3,] 0.4681937 0.000000 2.1431673
> det(K1)
[1] 1.4076
> L=(100/2)*log(det(K1))-(100/2)*sum(diag(K1%*%S))
> L
[1] -132.9057
```

Example, Iteration 2

```
> ## iter 2
> K0=K1
> K1=K0
> ## Update 12
> K1[1:2,1:2]=solve(S[1:2,1:2])+
+   K0[1:2,3]%*%solve(K0[3,3])%*%K0[3,1:2]
> K1
          [,1]     [,2]      [,3]
[1,] 1.3334723 2.563638 0.4681937
[2,] 2.5636375 5.871567 0.0000000
[3,] 0.4681937 0.000000 2.1431673
> ## 13
> K1[c(1,3),c(1,3)]=solve(S[c(1,3),c(1,3)])+
+   K1[c(1,3),2]%*%solve(K1[2,2])%*%K1[2,c(1,3)]
> K1
          [,1]     [,2]      [,3]
[1,] 1.3334723 2.563638 0.4681937
[2,] 2.5636375 5.871567 0.0000000
[3,] 0.4681937 0.000000 2.1431673
> det(K1)
[1] 1.4076
> L=(100/2)*log(det(K1))-(100/2)*sum(diag(K1%*%S))
> L
[1] -132.9057
```

Example, Iteration 3

```
> ## iter 3
> K0=K1
> K1=K0
> ## Update 12
> K1[1:2,1:2]=solve(S[1:2,1:2])+
+   K0[1:2,3]%*%solve(K0[3,3])%*%K0[3,1:2]
> K1
          [,1]     [,2]      [,3]
[1,] 1.3334723 2.563638 0.4681937
[2,] 2.5636375 5.871567 0.0000000
[3,] 0.4681937 0.000000 2.1431673
> ## 13
> K1[c(1,3),c(1,3)]=solve(S[c(1,3),c(1,3)])+
+   K1[c(1,3),2]%*%solve(K1[2,2])%*%K1[2,c(1,3)]
> K1
          [,1]     [,2]      [,3]
[1,] 1.3334723 2.563638 0.4681937
[2,] 2.5636375 5.871567 0.0000000
[3,] 0.4681937 0.000000 2.1431673
>
> L=(100/2)*log(det(K1))-(100/2)*sum(diag(K1%*%S))
> L
[1] -132.9057
```

Example, with ggmfit

```
> g1=ug(~a:b+a:c)
> colnames(S)=c("a","b","c")
> rownames(S)=c("a","b","c")
> fit1 <-ggmfit(S,n=100,edgeList(as(g1,"graphNEL")))
> fit1

$detK
[1] 1.4076

$S
          a          b          c
a  8.939863 -3.9033137 -1.9529918
b -3.903314  1.8745730  0.9210539
c -1.952992  0.9210539  0.8932473

$K
          a        b         c
a 1.3334723 2.563638 0.4681937
b 2.5636375 5.871567 0.0000000
c 0.4681937 0.000000 2.1431673

$iter
[1] 2
```

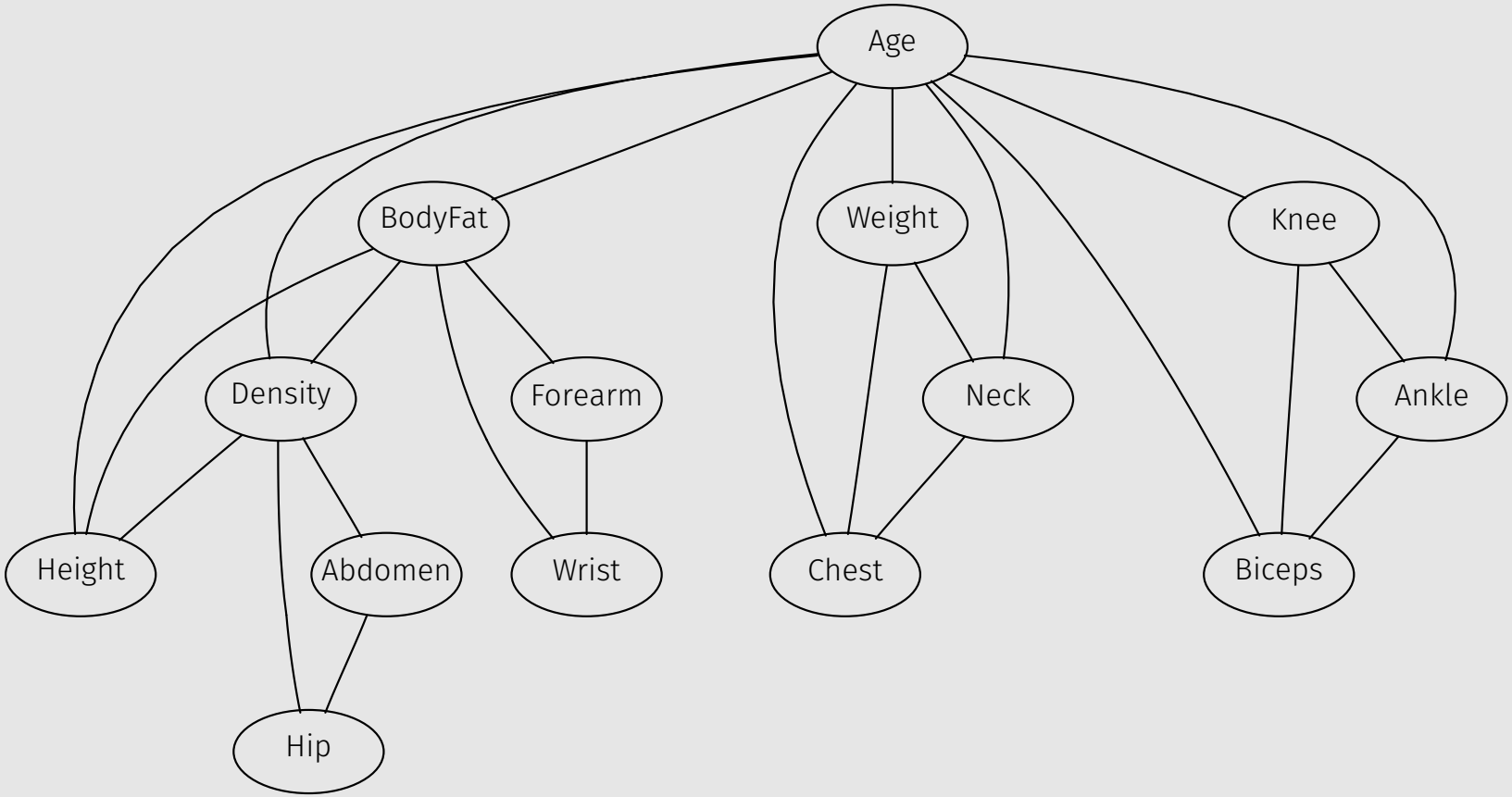
IPS with R, gRim package

```
> library(gRim)
> data(BodyFat)
> head(BodyFat)
  Density BodyFat Age Weight Height Neck Chest Abdomen   Hip Thigh Knee
1  1.0708    12.3  23 154.25  67.75 36.2  93.1    85.2  94.5  59.0 37.3
2  1.0853     6.1  22 173.25  72.25 38.5  93.6    83.0  98.7  58.7 37.3
3  1.0414    25.3  22 154.00  66.25 34.0  95.8    87.9  99.2  59.6 38.9
4  1.0751    10.4  26 184.75  72.25 37.4 101.8    86.4 101.2  60.1 37.3
5  1.0340    28.7  24 184.25  71.25 34.4  97.3   100.0 101.9  63.2 42.2
6  1.0502    20.9  24 210.25  74.75 39.0 104.5    94.4 107.8  66.0 42.0
  Ankle Biceps Forearm Wrist
1  21.9   32.0    27.4  17.1
2  23.4   30.5    28.9  18.2
3  24.0   28.8    25.2  16.6
4  22.8   32.4    29.4  18.2
5  24.0   32.2    27.7  17.7
6  25.6   35.7    30.6  18.8
> colnames(BodyFat)
 [1] "Density" "BodyFat" "Age"     "Weight"  "Height"  "Neck"    "Chest"
 [8] "Abdomen" "Hip"     "Thigh"   "Knee"    "Ankle"   "Biceps"  "Forearm"
[15] "Wrist"
```

IPS with R, gRim package

```
> g1<-cmod(~Density*BodyFat*Age*Height+
+            Age*Weight*Neck*Chest+
+            Abdomen*Hip*Density+
+            Knee*Ankle*Biceps*Age+
+            Forearm*Wrist*BodyFat,data=BodyFat)
> g1
Model: A cModel with 14 variables
 graphical :  TRUE  decomposable :  TRUE
 -2logL    :       15196.01 mdim :   38 aic :     15272.01
 ideviance :        2811.12 idf  :   24 bic :     15406.13
 deviance  :        1788.13 df   :   67
```

IPS with R, gRim package



Degree of freedom

```
> library(igraph)
> g1$fitinfo$dimension
mod.dim sat.dim   i.dim      df     idf
     38     105      14      67      24
> V(as(g1,"igraph"))
+ 14/14 vertices, named:
 [1] Age     BodyFat Density Height  Weight  Neck    Chest   Knee
 [9] Ankle   Biceps  Abdomen Hip     Forearm Wrist
> E(as(g1,"igraph"))
+ 24/24 edges (vertex names):
 [1] Age    --BodyFat Age    --Density Age    --Height  Age    --Weight
 [5] Age    --Neck    Age    --Chest   Age    --Knee    Age    --Ankle
 [9] Age    --Biceps  BodyFat--Density BodyFat--Height  BodyFat--Forearm
[13] BodyFat--Wrist   Density--Height  Density--Hip     Density--Abdomen
[17] Weight --Neck    Weight --Chest   Neck   --Chest   Knee   --Ankle
[21] Knee   --Biceps  Ankle  --Biceps  Abdomen--Hip     Forearm--Wrist
> p=length(V(as(g1,"igraph")))
> p*(p+1)/2 ## Number of parameters in sat model
[1] 105
> length(E(as(g1,"igraph")))+p ## Number of parameters in g1
[1] 38
> 105-38 ## df of deviance sat-g1
[1] 67
> 38-14  ## df of deviance g1- indep mod
[1] 24
```

IPS with R, gRim package

Compute S and we use `ggmfit` to fit the UGGM.

```
> S <- cov.wt(BodyFat[,nodes(as(g1,"graphNEL"))], method="ML")$cov
> S[1:3,1:3]
                  Age    BodyFat       Density
Age     158.18120118 30.6161628 -0.0663228553
BodyFat  30.61616276 69.7578962 -0.1566989475
Density  -0.06632286 -0.1566989  0.0003607582
> fit1 <-
+ ggmfit(S,n=nrow(BodyFat),edgeList(as(g1,"graphNEL")))
> names(fit1)
[1] "dev"   "df"    "detK"  "nvar"  "S"     "n.obs" "logL"  "K"     "iter"
```

IPS with R, gRim package

```
> round(fit1$K[1:8,1:8],3)
           Age BodyFat    Density Height Weight   Neck  Chest   Knee
Age      0.009  -0.008     -2.270  0.004  0.003 -0.012 -0.010 -0.006
BodyFat -0.008   0.603    259.703 -0.014  0.000  0.000  0.000  0.000
Density -2.270 259.703 120965.224 -6.870  0.000  0.000  0.000  0.000
Height   0.004  -0.014     -6.870  0.078  0.000  0.000  0.000  0.000
Weight   0.003   0.000      0.000  0.000  0.009 -0.035 -0.021  0.000
Neck    -0.012   0.000      0.000  0.000 -0.035  0.581 -0.019  0.000
Chest   -0.010   0.000      0.000  0.000 -0.021 -0.019  0.086  0.000
Knee    -0.006   0.000      0.000  0.000  0.000  0.000  0.000  0.402
> fit1$iter
[1] 902
> fit1$logL
[1] -7598.006
```

Hypothesis testing

Deviance

- $\mathcal{G}_s = (V, V \times V)$ the complete graph and $\mathcal{G}_\emptyset = (V, \emptyset)$ is the independent graph.
- $X_V^1, \ldots, X_V^n$ is an $n-$sample from $(X_V \sim \mathcal{N}(\mu, \Sigma), \mathcal{G} = (V, E))$ an UGGM, $\widehat{\Sigma}$ is the MLE of Σ, $\widehat{K} = \widehat{\Sigma}^{-1}$ such that

$$u \not\sim v \iff \widetilde{k}_{uv} = 0$$

- S is the MLE of Σ in the saturated UGGM. S and $\widehat{\Sigma}$ differ on those entries for which $\widetilde{k}_{uv} = 0$
  $\Rightarrow$ $\text{tr}(\widehat{K}S) = d$ and

$$\widehat{l} = \log L(\widehat{K}) = n \log \det(\widehat{K})/2 - nd/2.$$

- Deviance $D = \texttt{dev} = 2(\widehat{l}_s - \widehat{l}) = -n \log \det(S\widehat{K})$
- iDeviance $iD = \texttt{idev} = 2(\widehat{l} - \widehat{l}_i) = n \left(\log \det(\widehat{K}) + \sum_{k=i}^{d} \log(s_{ii}) \right)$

Nested models

- $\mathcal{M}_1 = (X_V, \mathcal{G}_1)$, $\mathcal{M}_0 = (X_V, \mathcal{G}_0)$, s.t. $\mathcal{G}_0 \subseteq \mathcal{G}_1$.We perform

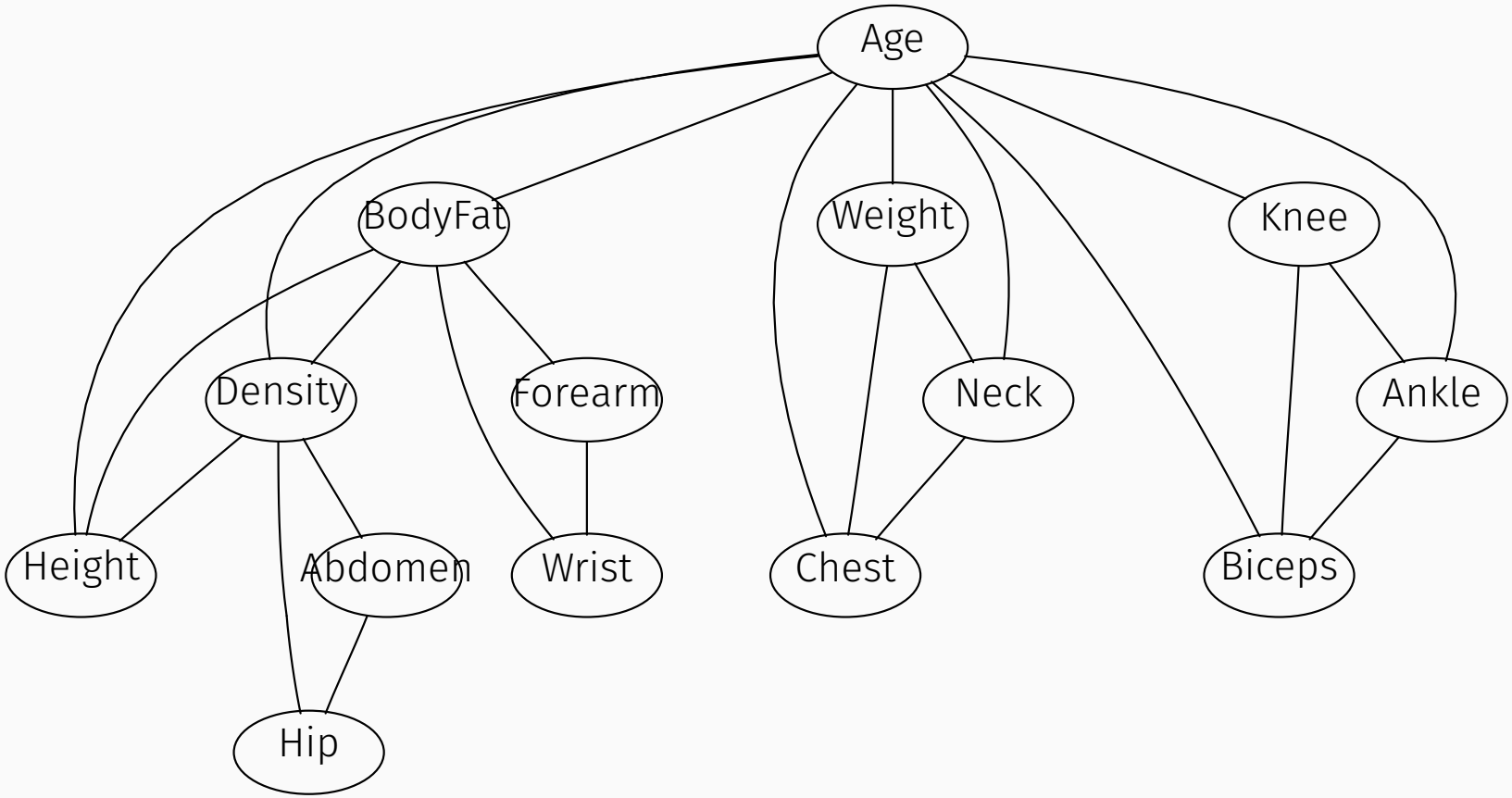
$$H_0 \ : \ \mathcal{M}_0 \text{ true vs } \ H_1 \ : \ \mathcal{M}_1 \text{ true}$$

- Log-ratio test statistic:

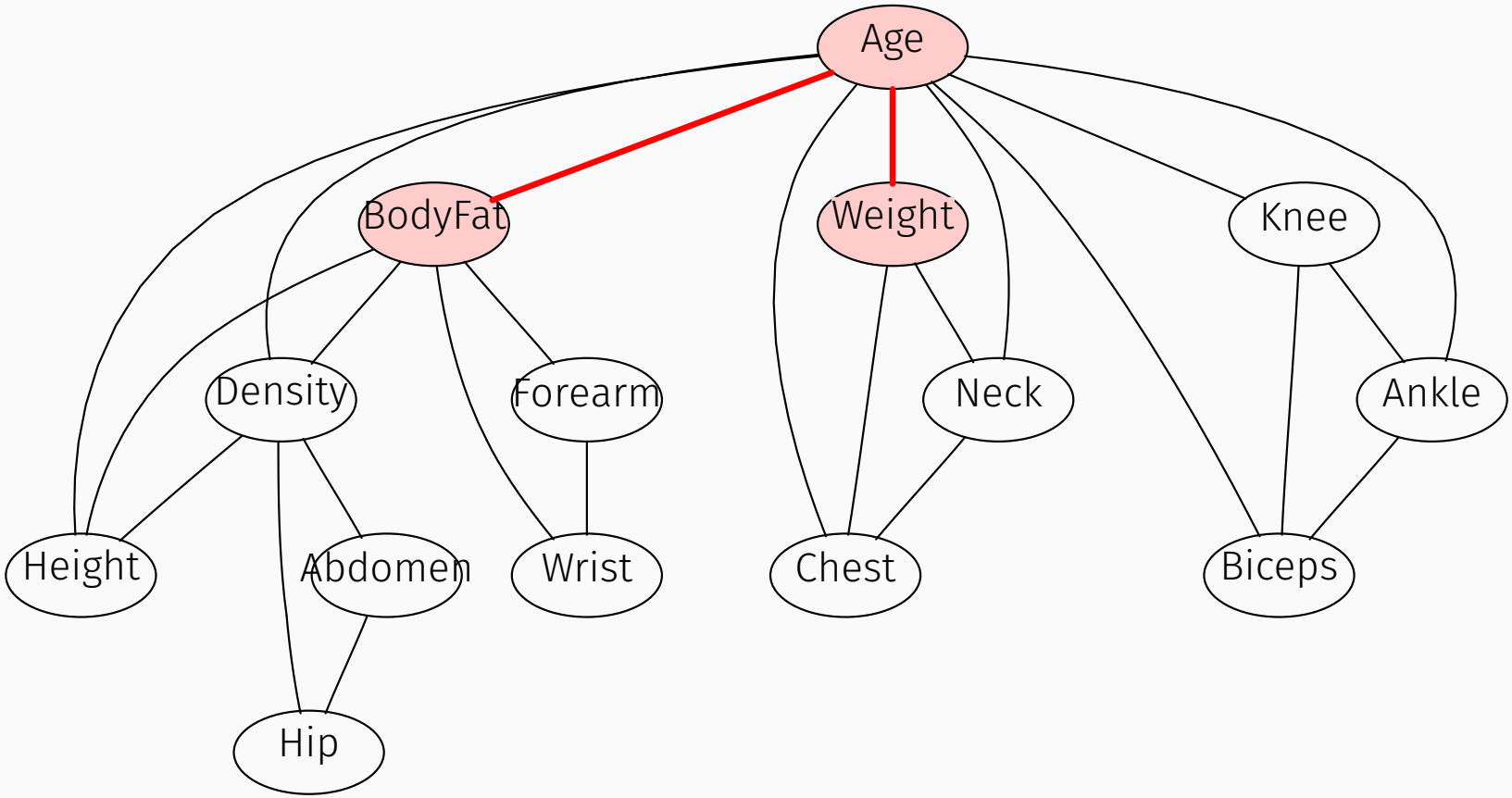
$$\texttt{lrt} = 2(\widehat{l}_1 - \widehat{l}_0) = n \log \frac{\det(\widehat{K}_1)}{\det(\widehat{K}_0)} \ \sim \ (n \to +\infty) \ \chi^2(k)$$

$k = \Delta(\text{of parameters}) \text{ btw } \mathcal{M}_1 \text{ and } \mathcal{M}_0$

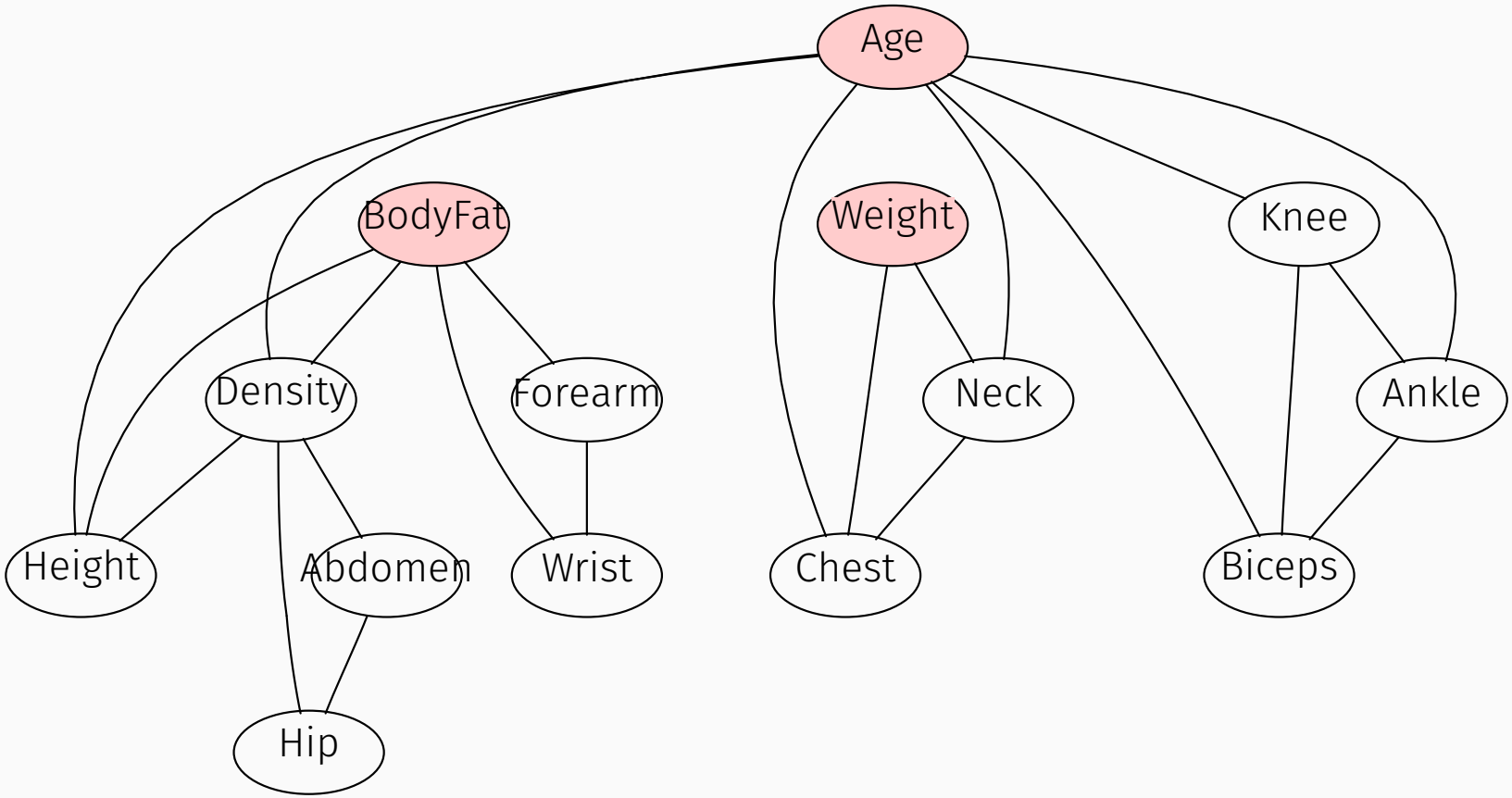
Nested models with R



Nested models with R



Nested models with R



Nested models with R

```
> g0=removeEdge(graph = as(g1,"graphNEL"),
+               from = c("Age","Age"),to=c("BodyFat","Weight"))
> g0
A graphNEL graph with undirected edges
Number of Nodes = 14
Number of Edges = 22
> g0<-cmod(g0,data=BodyFat)
> g0
Model: A cModel with 14 variables
 graphical :  TRUE  decomposable :  TRUE
 -2logL    :       15247.74 mdim :   36 aic :     15319.74
 ideviance :        2759.39 idf  :   22 bic :     15446.80
 deviance  :        1839.86 df   :   69
> g1
Model: A cModel with 14 variables
 graphical :  TRUE  decomposable :  TRUE
 -2logL    :       15196.01 mdim :   38 aic :     15272.01
 ideviance :        2811.12 idf  :   24 bic :     15406.13
 deviance  :        1788.13 df   :   67
```

Nested models with R

```
> lrt <- 2*(g1$fitinfo$logL - g0$fitinfo$logL)
> df <- g1$fitinfo$dimension[1]-g0$fitinfo$dimension[1]
> lrt
[1] 51.73143
> df
mod.dim
      2
> pchisq(lrt,df,lower.tail = F)
[1] 5.843373e-12
```

Testing $u \perp\!\!\!\perp v \mid V \setminus \{u, v\}$ with R

- Deviance test

```
> ciTest_mvn(list(cov=S, n.obs=nrow(BodyFat)),statistic = "DEV",
+             set =  colnames(S) )
Testing Age _|_ BodyFat | Density Height Weight Neck Chest Knee
+ Ankle Biceps Abdomen Hip Forearm Wrist
Statistic (DEV):    2.594 df: 1 p-value: 0.1073 method: CHISQ
```

- *F* statistic test (for small samples)

```
> ciTest_mvn(list(cov=S, n.obs=nrow(BodyFat)),statistic = "F",
+             set =  colnames(S) )
Testing Age _|_ BodyFat | Density Height Weight Neck Chest Knee
+ Ankle Biceps Abdomen Hip Forearm Wrist
Statistic (F):    2.462 df: 1 p-value: 0.1179 method: F
```

- Asymptotic normality of Fisher's z transform of the partial correlation

```
> library(pcalg)
> cS<-cov2cor(S)
> gaussCItest(1,2,3:14,list(C=cS,n=nrow(BodyFat)))
[1] 0.1180065
```

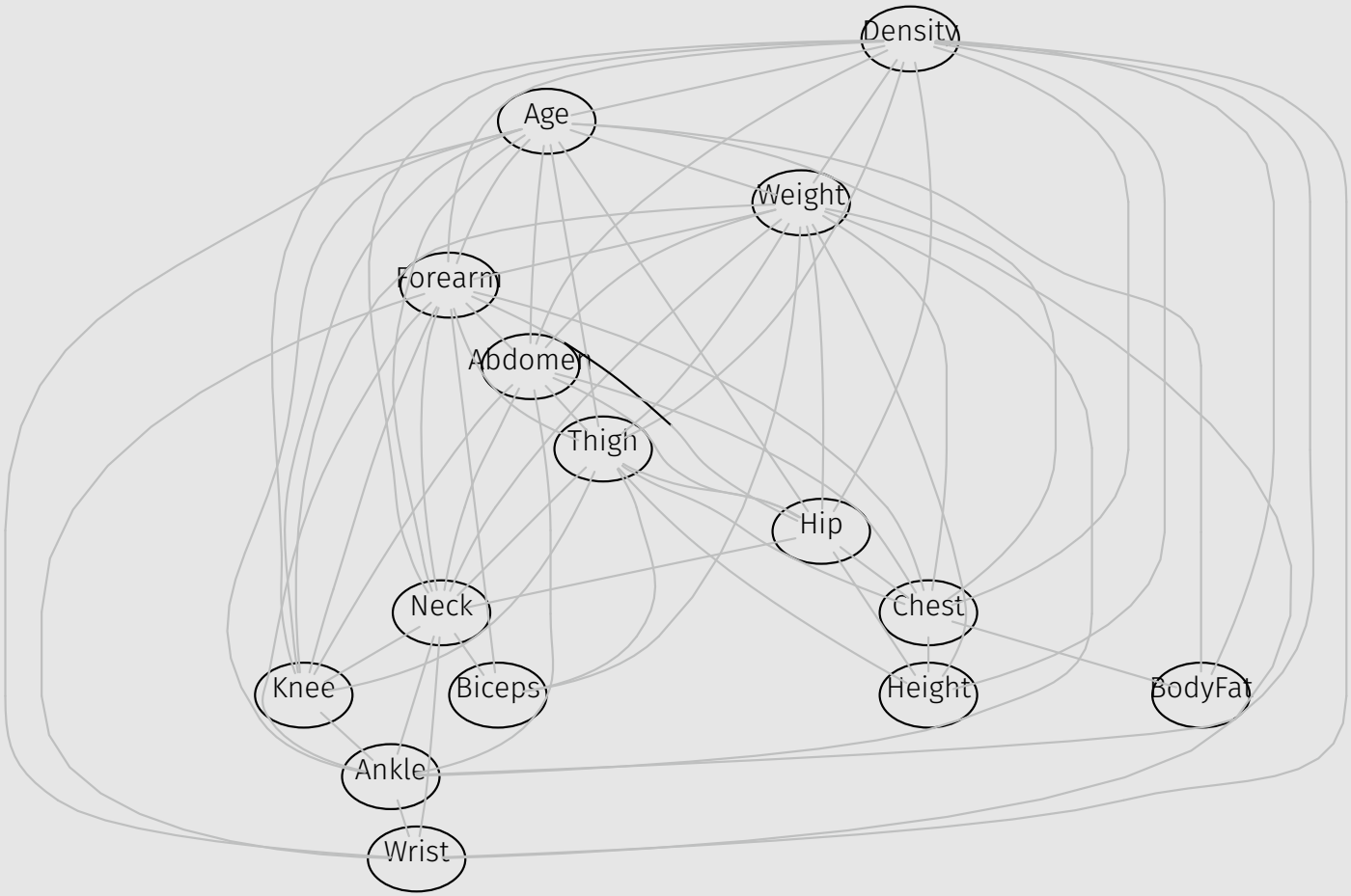
Model Selection

Stepwise procedure

Stepwise selection

```
> g.sat<-cmod(~.^.,data=BodyFat)
> g<-stepwise(g.sat,criterion = "test",details = 1,
+             direction ="backward")
STEPWISE:
 criterion: test
 direction: backward
 type     : decomposable
 search   : all
 steps    : 1000
. BACKWARD: type=decomposable search=all, criterion=test, alpha=0.05
. Initial model: is graphical=TRUE is decomposable=TRUE
  p.value    0.9735 Edge deleted: Hip Wrist
  p.value    0.9765 Edge deleted: Abdomen Wrist
  p.value    0.9622 Edge deleted: BodyFat Wrist
  p.value    0.9494 Edge deleted: Knee BodyFat
.
.
.
  p.value    0.3151 Edge deleted: Abdomen Height
  p.value    0.0541 Edge deleted: Age Height
```

Stepwise selection



Lasso procedure

Lasso algorithm

- K is maximized using the log-likelihood penalized by L_1-norm of K, i.e; $|K| = \sum |k_{uv}|$
- L_1-penalized log-likelihood

$$L_{\text{pen}}(K) = \log \det(K) - \text{tr}(KS) - \rho|K|$$

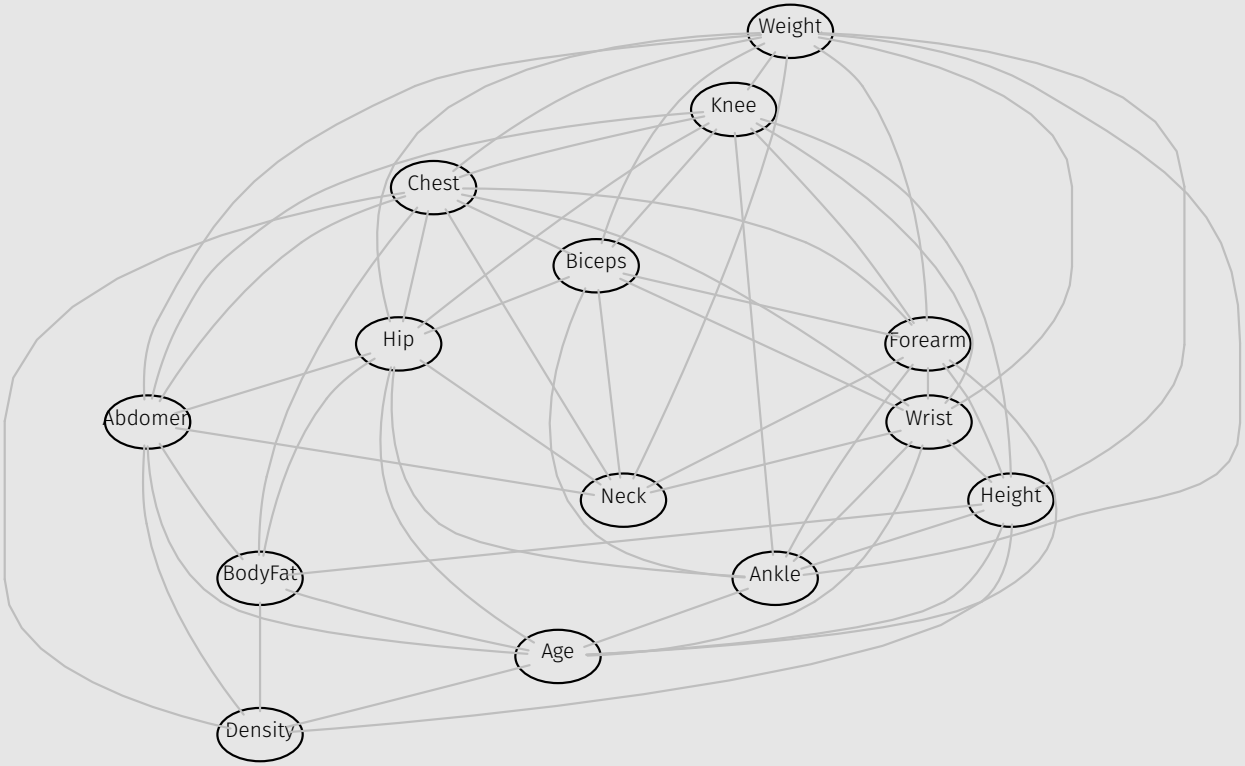
  where $\rho > 0$

- $K \mapsto L_{\text{pen}}(K)$ is convex $\Rightarrow$ it can be optimized by convex programming methods.
- $\rho \to 0$, more dense will be the estimated graph.

Lasso algorithm, with R

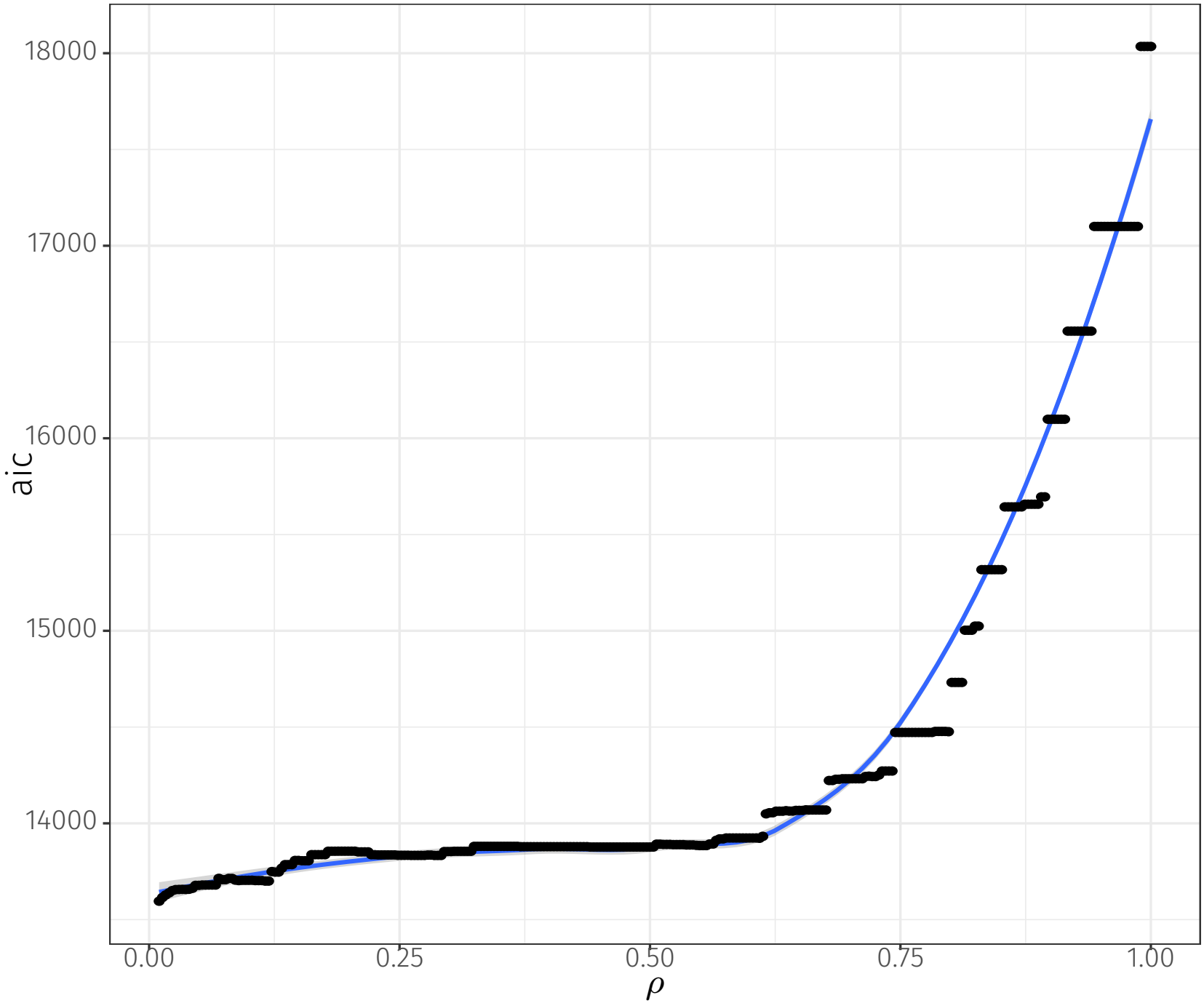
```
> C.body<-cov2cor(S)
> library(glasso)
> res.lasso<-glasso(C.body,rho=0.1)
> AM <- res.lasso$wi != 0
> diag(AM) <- F
> g.lasso <- as(AM, "graphNEL")
> nodes(g.lasso)<-colnames(S)
> glasso.body<-cmod(g.lasso,data=BodyFat)
> glasso.body
Model: A cModel with 14 variables
 graphical :  TRUE  decomposable : FALSE
 -2logL    :       13562.05 mdim :   71 aic :     13704.05
 ideviance :        4445.08 idf  :   57 bic :     13954.64
 deviance  :         154.16 df   :   34
```

Lasso algorithm, with R



Estimating ρ

```
> rho=seq(0.01,1,length=300)
> LassoGGM=function(rho){
+   res.lasso<-glasso(C.body,rho)
+   AM <- res.lasso$wi != 0
+   diag(AM) <- F
+   g.lasso <- as(AM, "graphNEL")
+   nodes(g.lasso)<-colnames(S)
+   glasso.body<-cmod(g.lasso,data=BodyFat)
+   glasso.body
+ }
> all.glasso=all.aic=vector("list",length(rho))
> for(j in 1:length(rho)) all.glasso[[j]]=LassoGGM(rho[j])
> for(j in 1:length(rho)) all.aic[[j]]=all.glasso[[j]]$fitinfo$aic
> dt=data.frame(rho,unlist(all.aic))
> colnames(dt)=c("rho","aic")
> library(ggplot2)
> p<-ggplot(dt,aes(x=rho,y=aic))+geom_smooth()+geom_point()+theme_bw()
> p
```


18000
17000
16000
15000
14000
aic
0.00
0.25
0.50
0.75
1.00
ρ

SINful procedure

Simultanuous Testing

- The set of hypotheses

$$H = \{H_0^{uv} : u \perp\!\!\!\perp v \mid V \setminus \{u, v\}\}_{u<v}$$

  and $\mathcal{P} = \{p_{uv},\}_{u<v}$ is the set of $p-$values

- Multiple hypothesis tests

	Null hypothesis is true (H_0^{uv})	Alternative hypothesis is true (H_A^{uv})	Total
Test is declared significant	V	S	R
Test is declared non-significant	U	T	$m - R$
Total	m_0	$m - m_0$	m

Simultanuous Testing

- The set of hypotheses

$$H = \{H_0^{uv} : u \perp\!\!\!\perp v \mid V \setminus \{u, v\}\}_{u<v}$$

  and $\mathcal{P} = \{p_{uv},\}_{u<v}$ is the set of $p-$values

- Multiple hypothesis tests

	Null hypothesis is true (H_0^{uv})	Alternative hypothesis is true (H_A^{uv})	Total
Test is declared significant	V	S	R
Test is declared non-significant	U	T	$m - R$
Total	m_0	$m - m_0$	m

$m = p(p-1)/2$ is the total number hypotheses tested

Simultanuous Testing

- The set of hypotheses

$$H = \{H_0^{uv} : u \perp\!\!\!\perp v \mid V \setminus \{u, v\}\}_{u<v}$$

  and $\mathcal{P} = \{p_{uv},\}_{u<v}$ is the set of $p-$values

- Multiple hypothesis tests

	Null hypothesis is true (H_0^{uv})	Alternative hypothesis is true (H_A^{uv})	Total
Test is declared significant	V	S	R
Test is declared non-significant	U	T	$m - R$
Total	$\mathbf{m_0}$	$m - m_0$	m

m_0 is the number of true null hypotheses, an unknown parameter

Simultanuous Testing

- The set of hypotheses

$$H = \{H_0^{uv} : u \perp\!\!\!\perp v \mid V \setminus \{u, v\}\}_{u<v}$$

  and $\mathcal{P} = \{p_{uv},\}_{u<v}$ is the set of $p-$values

- Multiple hypothesis tests

	Null hypothesis is true (H_0^{uv})	Alternative hypothesis is true (H_A^{uv})	Total
Test is declared significant	V	S	R
Test is declared non-significant	U	T	$m - R$
Total	m_0	$\mathbf{m - m_0}$	m

$m - m_0$ is the number of true alternative hypotheses

Simultanuous Testing

- The set of hypotheses

$$H = \{H_0^{uv} : u \perp\!\!\!\perp v \mid V \setminus \{u, v\}\}_{u<v}$$

  and $\mathcal{P} = \{p_{uv},\}_{u<v}$ is the set of $p-$values

- Multiple hypothesis tests

	Null hypothesis is true (H_0^{uv})	Alternative hypothesis is true (H_A^{uv})	Total
Test is declared significant	V	S	R
Test is declared non-significant	U	T	$m - R$
Total	m_0	$m - m_0$	m

V is the number of false positives (Type I error) (also called "false discoveries")

Simultanuous Testing

- The set of hypotheses

$$H = \{H_0^{uv} : u \perp\!\!\!\perp v \mid V \setminus \{u, v\}\}_{u<v}$$

and $\mathcal{P} = \{p_{uv},\}_{u<v}$ is the set of $p-$values

- Multiple hypothesis tests

	Null hypothesis is true (H_0^{uv})	Alternative hypothesis is true (H_A^{uv})	Total
Test is declared significant	V	S	R
Test is declared non-significant	U	T	$m - R$
Total	m_0	$m - m_0$	m

S is the number of true positives (also called "true discoveries")

Simultanuous Testing

- The set of hypotheses

$$H = \{H_0^{uv} : u \perp\!\!\!\perp v \mid V \setminus \{u, v\}\}_{u<v}$$

  and $\mathcal{P} = \{p_{uv},\}_{u<v}$ is the set of $p-$values

- Multiple hypothesis tests

	Null hypothesis is true (H_0^{uv})	Alternative hypothesis is true (H_A^{uv})	Total
Test is declared significant	V	S	R
Test is declared non-significant	U	T	$m - R$
Total	m_0	$m - m_0$	m

T is the number of false negatives (Type II error)

Simultanuous Testing

- The set of hypotheses

$$H = \{H_0^{uv} : u \perp\!\!\!\perp v \mid V \setminus \{u, v\}\}_{u<v}$$

  and $\mathcal{P} = \{p_{uv},\}_{u<v}$ is the set of $p-$values

- Multiple hypothesis tests

	Null hypothesis is true (H_0^{uv})	Alternative hypothesis is true (H_A^{uv})	Total
Test is declared significant	V	S	R
Test is declared non-significant	U	T	$m - R$
Total	m_0	$m - m_0$	m

U is the number of true negatives

Simultanuous Testing

- The set of hypotheses

$$H = \{H_0^{uv} : u \perp\!\!\!\perp v \mid V \setminus \{u, v\}\}_{u<v}$$

  and $\mathcal{P} = \{p_{uv},\}_{u<v}$ is the set of $p-$values

- Multiple hypothesis tests

	Null hypothesis is true (H_0^{uv})	Alternative hypothesis is true (H_A^{uv})	Total
Test is declared significant	V	S	R
Test is declared non-significant	U	T	$m - R$
Total	m_0	$m - m_0$	m

Family wise error rate :

$$\text{FWER} = \mathbb{P}(V \geq 1)$$

SINful procedure

- The partial correlation matrix $\rho = (\rho_{uv})$:

$$\rho_{uv} = -\frac{k_{uv}}{\sqrt{k_{uu}k_{vv}}}$$

- The sample partial correlation, $r = (r_{uv})$

$$r_{uv} = -\frac{w_{uv}}{\sqrt{w_{uu}w_{vv}}}$$

  where $W = S^{-1} = (w_{uv})$

- $z-$transform of r_{uv} and ρ_{uv}:

$$z_{uv} = \frac{1}{2}\log\left(\frac{1-r_{uv}}{1+r_{uv}}\right) \text{ and } \xi_{uv} = \frac{1}{2}\log\left(\frac{1-\rho_{uv}}{1+\rho_{uv}}\right)$$

- $n \to +\infty$

$$\sqrt{n-|V|-1}(z_{uv}-\xi_{uv}) \sim \mathcal{N}(0,1)$$

SINful procedure, simultaneous $p-$values

- $\forall\, uv$, the conservative simultaneous $p-$values are

$$p_{uv} = 1 - \left[2\Phi\left(\sqrt{n_V}(z_{uv})\right) - 1\right]^{\frac{|V|(|V|-1)}{2}}$$

  where $n_V = n - |V| - 1$ and Φ is the cdf of $\mathcal{N}(0,1)$.

- If $p_{uv} \geq \alpha$, then H_0^{uv} ($\iff$ $\rho_{uv} = 0$) is accepted.

Proposition

Let $\mathcal{G} = (V, E)$ be the UG underlying the data-generating normal distribution. Then

$$\liminf_{n \to +\infty} \mathbb{P}\left\{\, p_{uv} \geq \alpha,\ \forall uv \notin E \,\right\} \geq 1 - \alpha \tag{1}$$

- $\overline{\left\{\, p_{uv} \geq \alpha,\ \forall uv \notin E \,\right\}} = \exists uv \notin E \text{ s.t. } p_{uv} < \alpha \iff V \geq 1.$
- (1) $\Rightarrow$ $\mathbb{P}(V \geq 1) \leq \alpha$ (FWER)

SINful procedure, Holm's procedure

- *Holm's procedure*,

  1. $m = \frac{|V|(|V|-1)}{2}$ and order the $p-$values $p_{(1)} \leq \ldots \leq p_{(m)}$
  2. let k be the minimal index such that $p_{(k)} > \dfrac{\alpha}{m+1-k}$
  3. $\begin{cases} H_0^{(1)}, \ldots, H_0^{(k-1)} & \text{are rejected} \\ H_0^{(k)}, \ldots, H_0^{(m)} & \text{are accepted} \end{cases}$
  4. If $k = 1$ then accept all and if no such k exist then reject all.

- If $\widehat{\mathcal{G}}^H(\alpha)$ is the estimated UG, then

$$\liminf_{n \to +\infty} \mathbb{P}\left(\widehat{\mathcal{G}}^H(\alpha) \subseteq \mathcal{G}\right) \geq 1 - \alpha$$

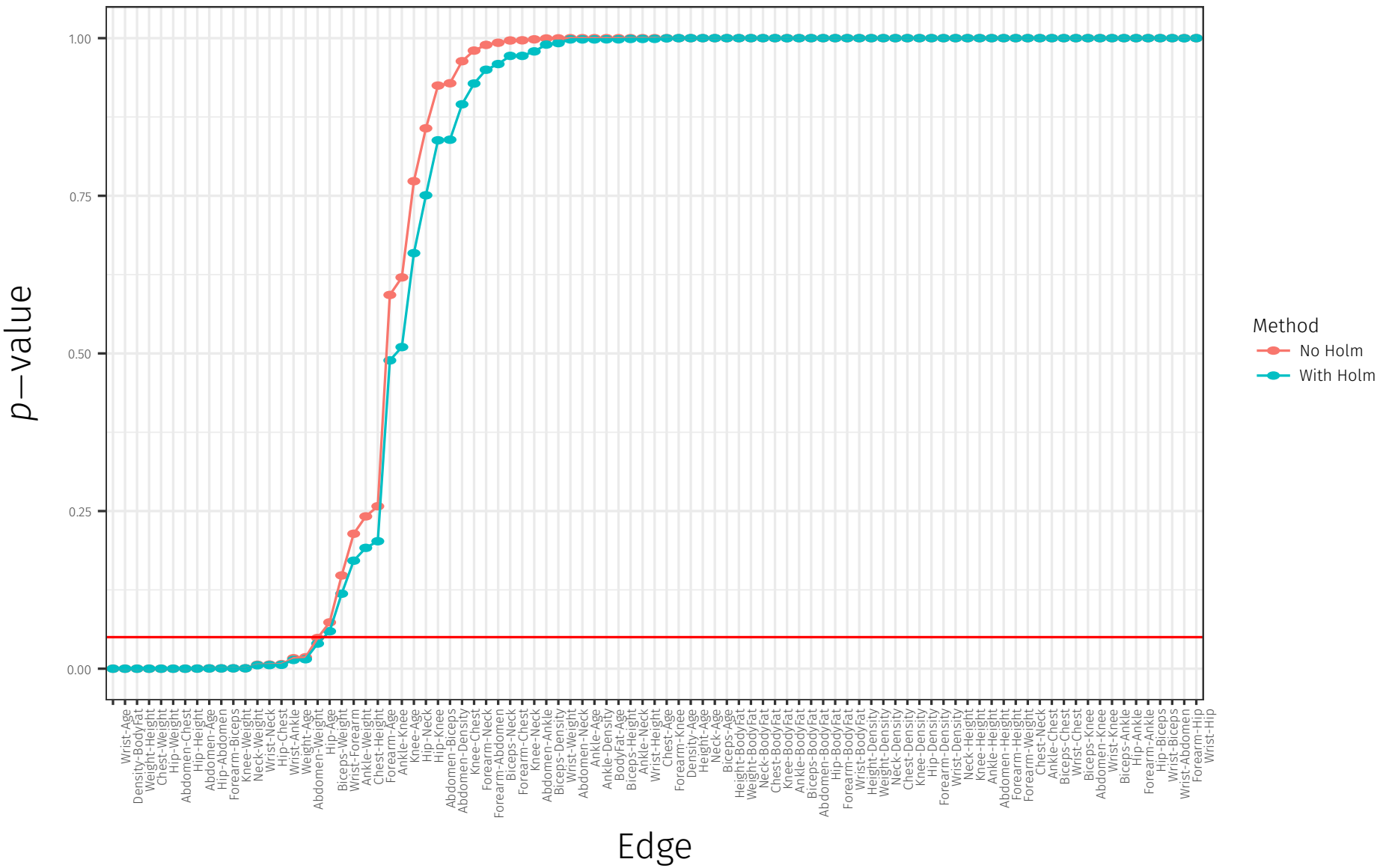
SINfull procedure, with R,

```
> library(SIN)
> psin.NH<-sinUG(S,n=nrow(BodyFat),holm = F)
> psin.WH<-sinUG(S,n=nrow(BodyFat),holm = T)
> psin.NH[upper.tri(psin.NH)]=-1
> diag(psin.NH)=-1
> psin.WH[upper.tri(psin.WH)]=-1
> diag(psin.WH)=-1
> library(reshape)
> xNH=melt(psin.NH)
> xWH=melt(psin.WH)
> xNH=xNH[-which(xNH$value==-1),]
> xWH=xWH[-which(xWH$value==-1),]
> method=c(rep("No Holm",nrow(xNH)),rep("with Holm",nrow(xWH)))
> dt=rbind.data.frame(xNH,xWH)
> dt$method=method
> dt$Edge=paste(dt$X1,dt$X2,sep="-")
```

SINfull procedure, with R,

```
> library(ggplot2)
> i=order(xNH$value)
> dt$Edge=factor(dt$Edge,levels=dt$Edge[i])
> p<-ggplot(dt,aes(x=Edge,y=value,col=method,group=method))+geom_point()+ge
> p<-p+theme_bw()+theme(axis.text.x = element_text(angle = 90, hjust = 1))
> p<-p+geom_hline(aes(yintercept =0.05),col="red")
> p
```

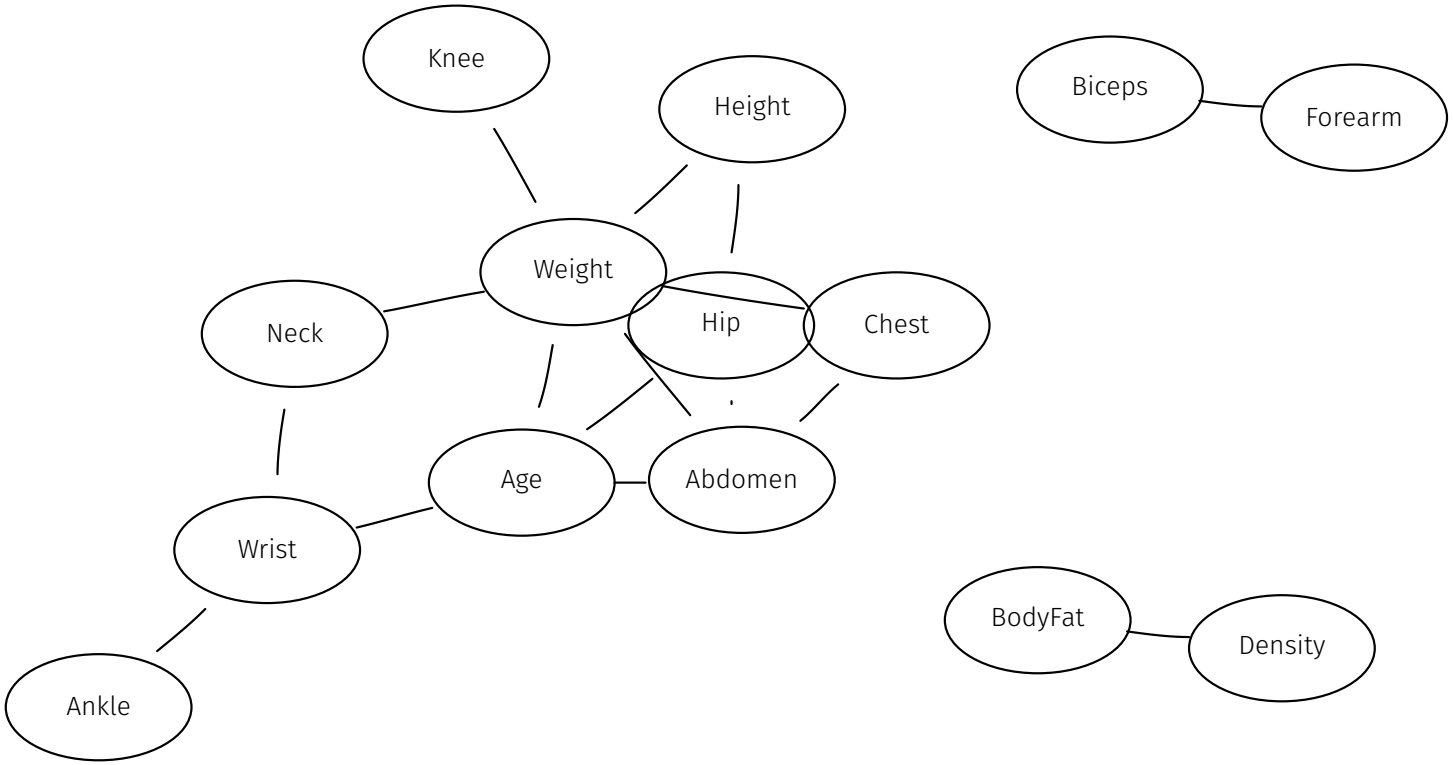
SINfull procedure, with R,



SINfull procedure, with R,

```
> psin.NH<-sinUG(S,n=nrow(BodyFat),holm = F)
> psin.WH<-sinUG(S,n=nrow(BodyFat),holm = T)
> gsin.NH <- as(getgraph(psin.NH, 0.05), "graphNEL")
> gsin.NH
A graphNEL graph with undirected edges
Number of Nodes = 14
Number of Edges = 18
> gsin.WH <- as(getgraph(psin.WH, 0.05), "graphNEL")
> gsin.WH
A graphNEL graph with undirected edges
Number of Nodes = 14
Number of Edges = 18
```

SINfull procedure, with R,



Directed Gaussian Graphical Models

Markov properties

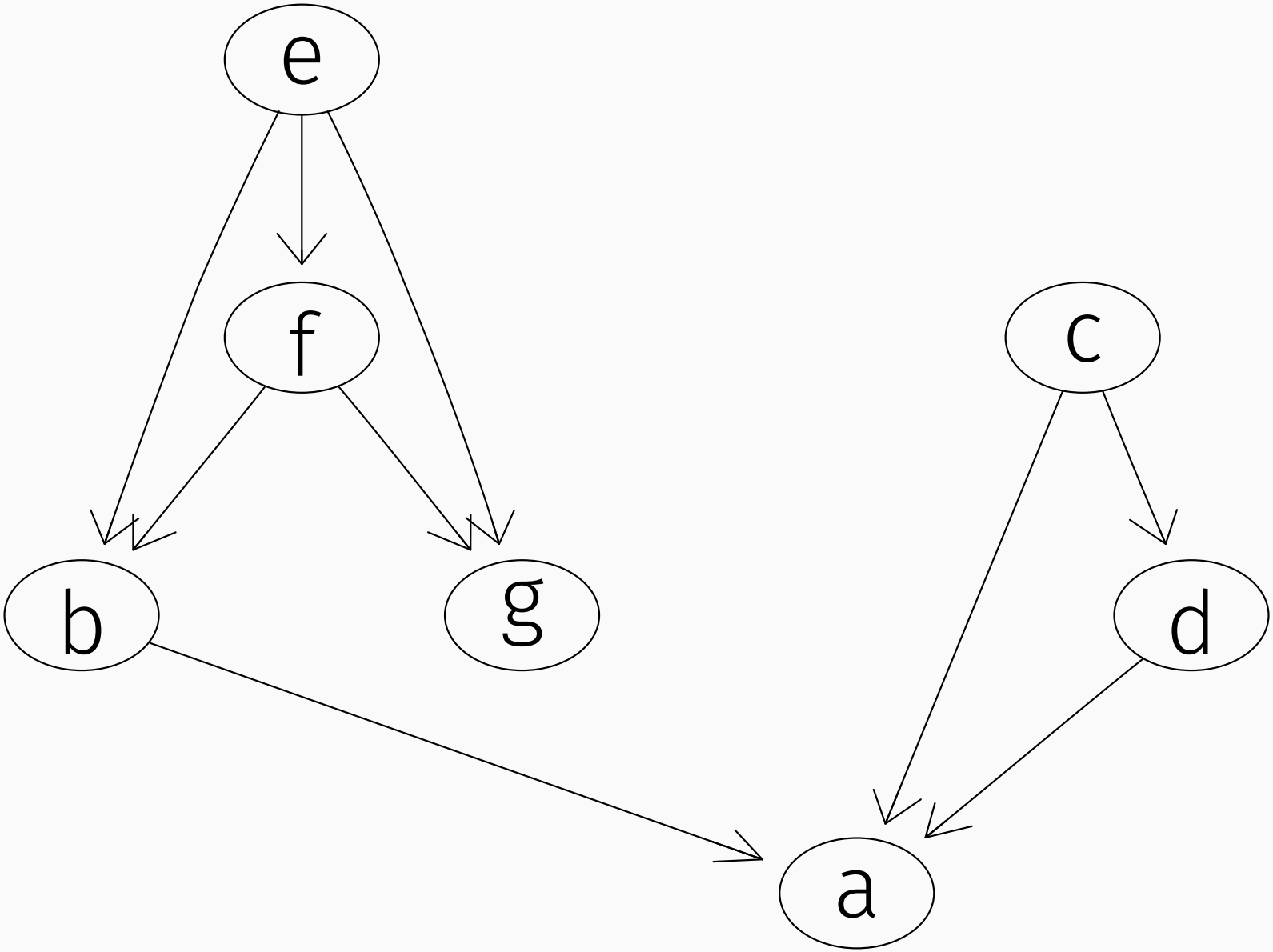
- $\vec{\mathcal{G}} = (V, E)$ is a DAG, $X_V = (X_v,\ v \in E) \sim \mathbf{p}$, p is with density f.
  f factorizes w.r.t $\vec{\mathcal{G}}$ iff
$$f(x_V) = \prod_{v \in V} g(x_v \mid x_{\mathrm{pa}(v)})$$

- A skeleton of DAG is an undirected graph formed by replacing all arrows with (undirected) lines.

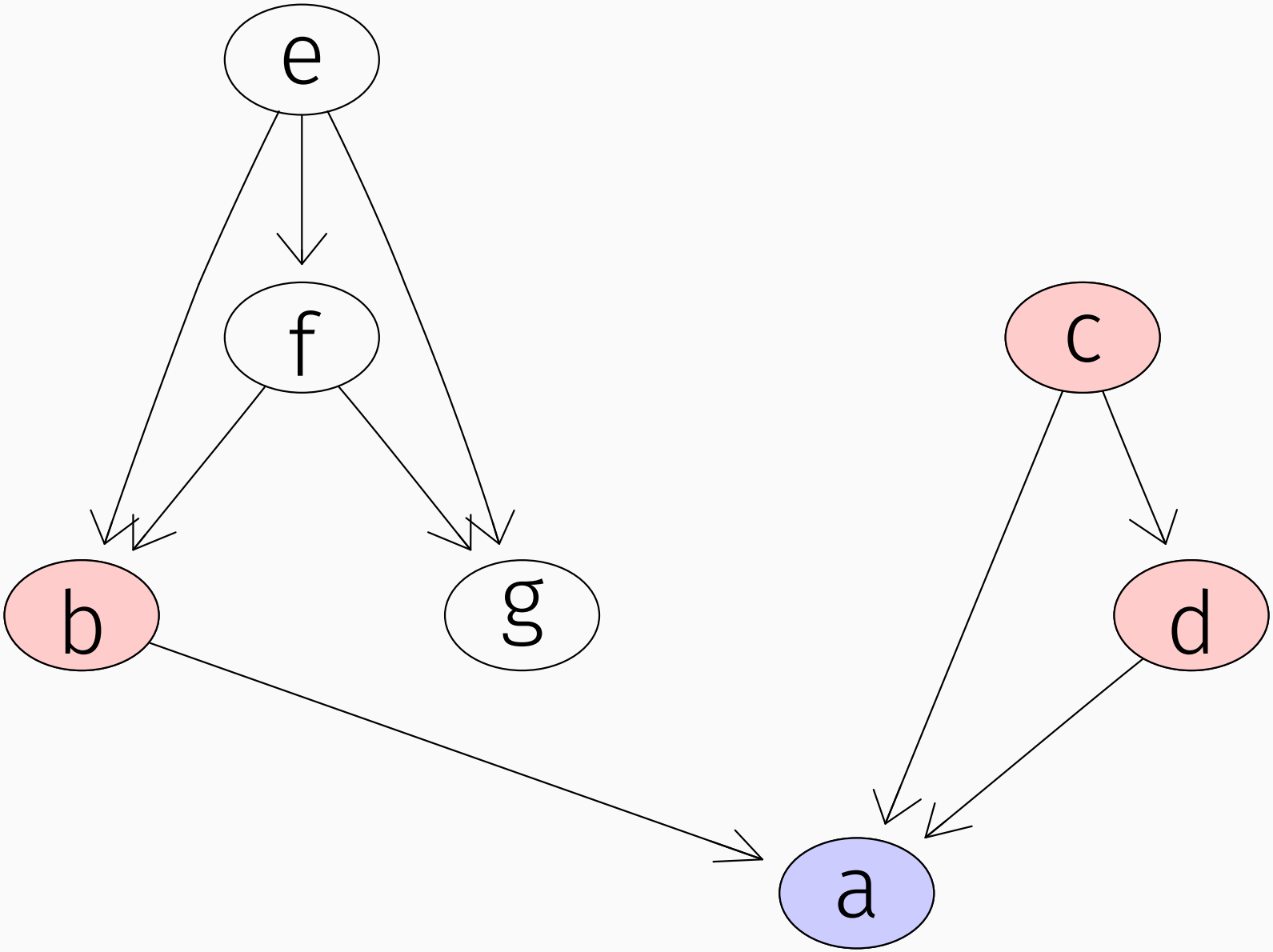
- An immorality is a node where it meets two directed edges from non-adjacent nodes meet: $a \to b \leftarrow c$. It's called also a $v-$structure:
$$a \to b \leftarrow c \iff b \perp\!\!\!\perp c \text{ and } b \not\perp\!\!\!\perp c \mid a$$

- Two DAGs are Markov equivalent if and only if they have the same skeleton and the same immoralities (iff they have the same moral graph, see talk 1).

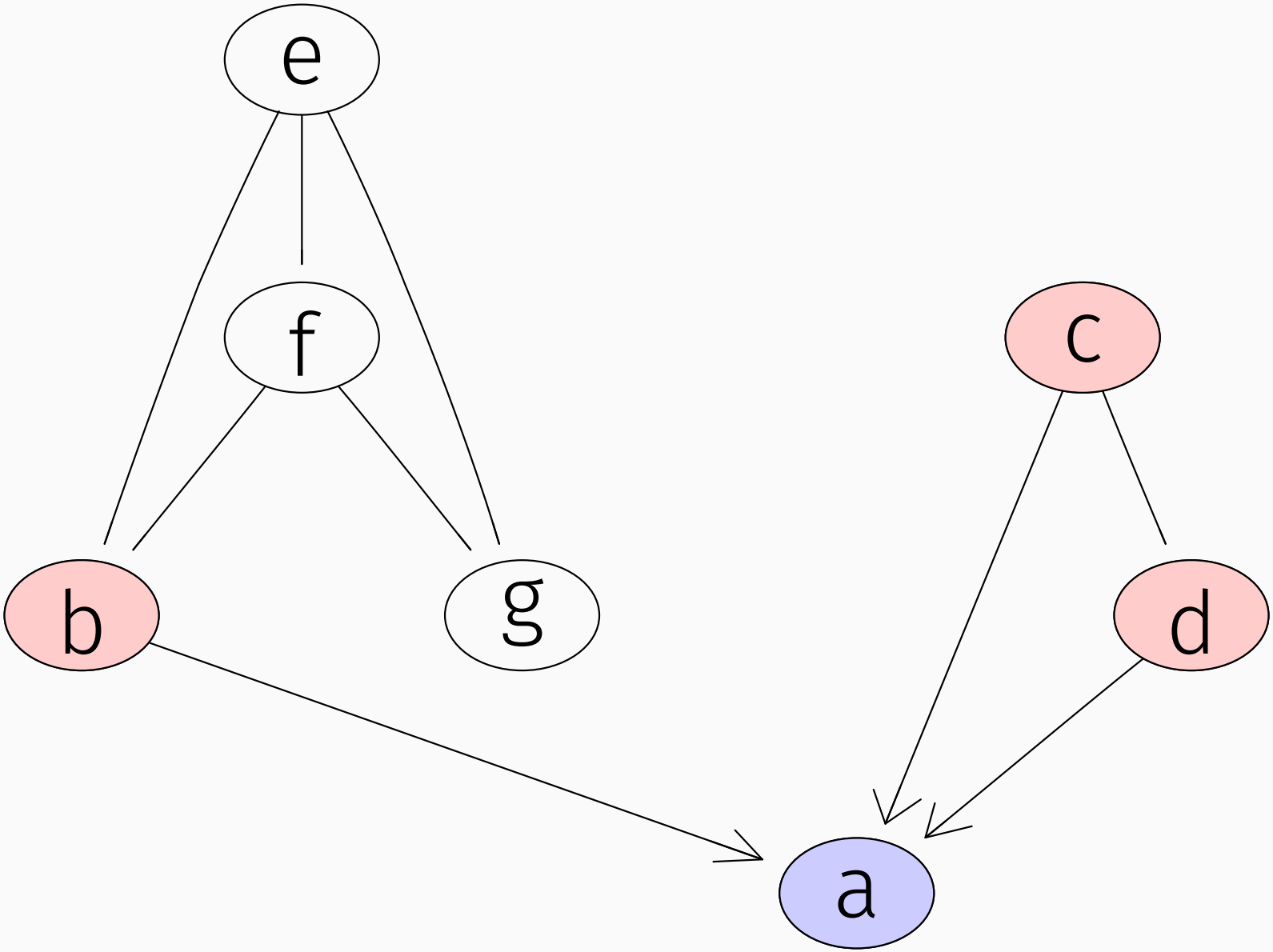
Partially directed acyclic graphs



Partially directed acyclic graphs



Partially directed acyclic graphs



Ordering variables

- Either orderings is known in advance: for example, temporal orderings $\Rightarrow$ Model selection is a series of univariate model selection tasks, regressing each variable on the prior variables.
- Or variable ordering is not known in advance, and must be inferred from the data $\Rightarrow$ Model selection will provide markov equivalence class of DAGs.
- To relate essential graphs to undirected graphs, consider a DAG generated by ordering the vertices in a triangulated graph using a perfect ordering $\Rightarrow$ the DAG has no immoralities, and all DAGs generated in this way are Markov equivalent.

PC algorithm

PC algorithm (Spirtes and Glymour 1991; Spirtes et al. 1993)

- R package: `pcalg`
- Step 1: estimating the skeleton $\mathcal{G}$ of $\vec{\mathcal{G}}$

$$u \not\sim_{\mathcal{G}} v \iff \exists S \subseteq V \text{ s.t. } u \perp\!\!\!\perp v \mid S$$

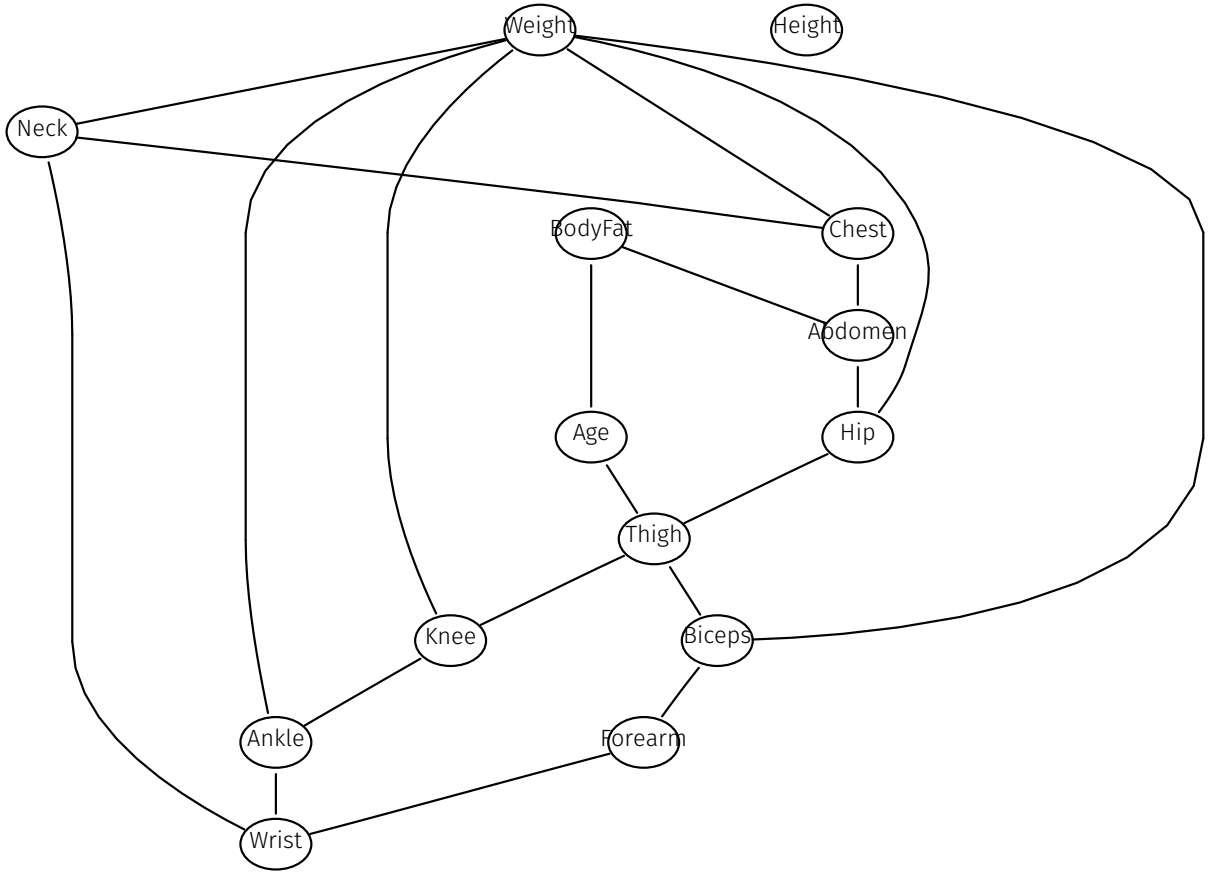
  - Start by marginal independence and S, s.t. $|S| = 1, 2, \ldots$ where $S \subseteq \text{bd}(u) \cup \text{bd}(v)$
  - Result: $\mathcal{S} = \{(u, v, S) \text{ s.t. } u \perp\!\!\!\perp v \mid S\} = \{(u, v, S), S\, d-\text{sep } u \text{ and } v\}$

- Step 2: Assigns directions to the edges of the skeleton using the set $\mathcal{S}$ computed in Step 1"
  1. We first determine the $v-$structures: $b \perp\!\!\!\perp c$ and $b \not\perp\!\!\!\perp c \mid a$
  2. Orient the edges in a way that we don't obtain more immoralities (or $v-$structures).

Example with R, Estimating skeleton

```
> data(BodyFat)
> colnames(BodyFat)
 [1] "Density" "BodyFat" "Age"     "Weight"  "Height"  "Neck"    "Chest"
 [8] "Abdomen" "Hip"     "Thigh"   "Knee"    "Ankle"   "Biceps"  "Forearm"
[15] "Wrist"
> ## Compute the sample covariance and correlation matrix
> S.body <- cov.wt(BodyFat[,-1], method="ML")$cov ## 1st variable measure the bo
> C.body=cov2cor(S.body)
> ## Step1: Estimate the skeleton
> library(pcalg)
> suffStat<-list(C=C.body,n=nrow(BodyFat))
> skeleton.body<- skeleton(suffStat = suffStat,indepTest = gaussCItest,labels=na
> plot(skeleton.body@graph)
```

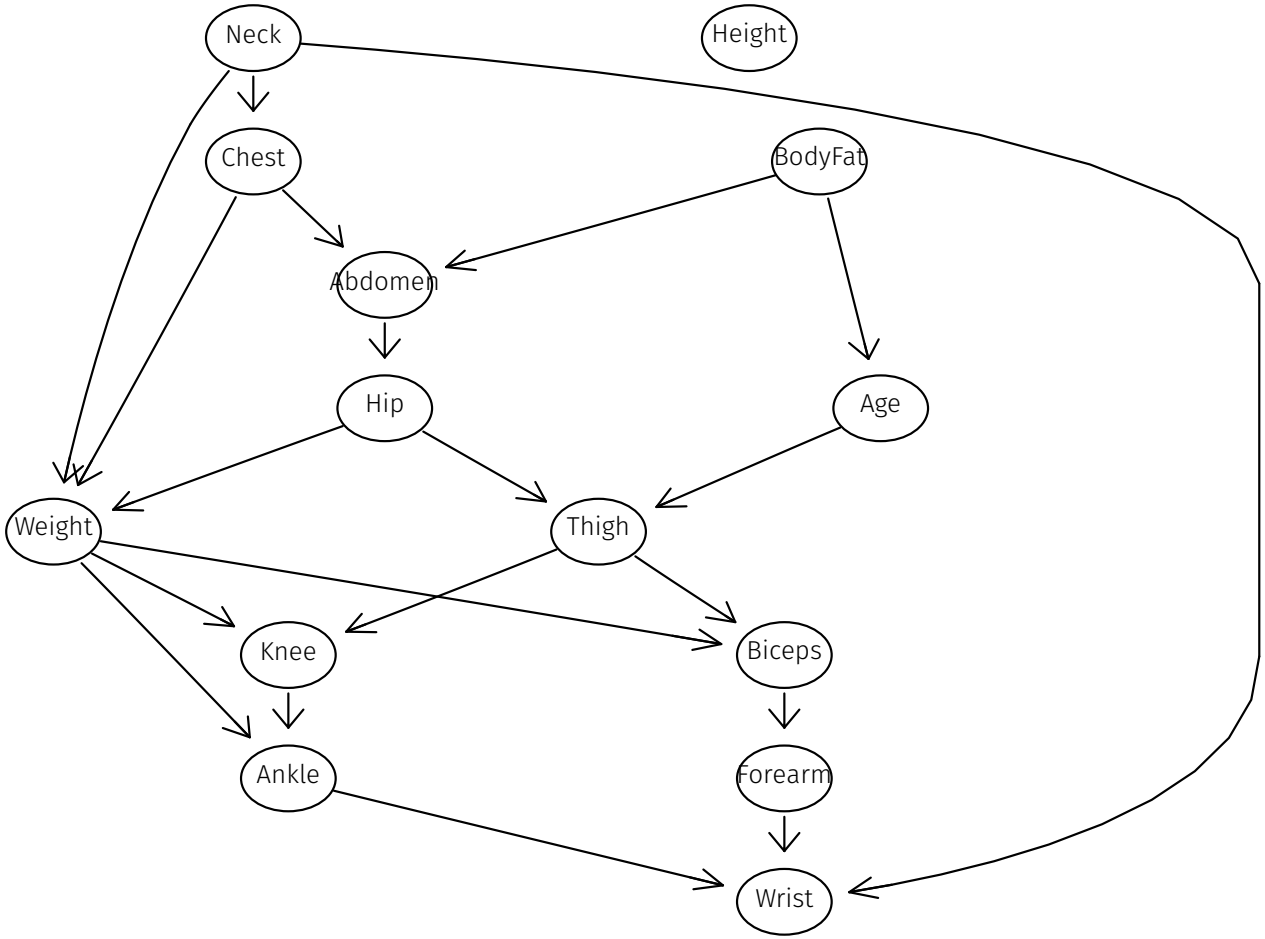
Example with R, Estimating skeleton



Example with R, Assigning directions

```
> pdag.body <- udag2pdag(skeleton.body,verbose = 0)
> nodes(pdag.body@graph)=colnames(S.body)
> plot(pdag.body@graph)
```

Example with R, Assigning directions



Does the model fit the data?

```
> m<-fitDag(amat = as(pdag.body@graph,"matrix"), S = S.body, n = nrow(BodyFat))
> names(m)
[1] "Shat" "Ahat" "Dhat" "dev"  "df"
> m[c("dev","df")]
$dev
[1] 662.5623

$df
[1] 71
> ## minus the coefficient of the regression Weight ~ Neck+Chest+Hip
> ##  (its parents)
> colnames(m$Ahat)
 [1] "BodyFat" "Age"     "Weight"  "Height"  "Neck"    "Chest"   "Abdomen"
 [8] "Hip"     "Thigh"   "Knee"    "Ankle"   "Biceps"  "Forearm" "Wrist"
> m$Ahat[3,]
   BodyFat        Age     Weight     Height       Neck      Chest
 0.0000000  0.0000000  1.0000000  0.0000000 -2.5187334 -0.8634354
   Abdomen        Hip      Thigh       Knee      Ankle     Biceps
 0.0000000 -2.3889098  0.0000000  0.0000000  0.0000000  0.0000000
   Forearm      Wrist
 0.0000000  0.0000000
> ##
> pchisq(q =  662.5623,df = 71,lower.tail = F)
[1] 7.588494e-97
```

Hill-climbing algorithm

Greedy Search, hill-climbing algorithm

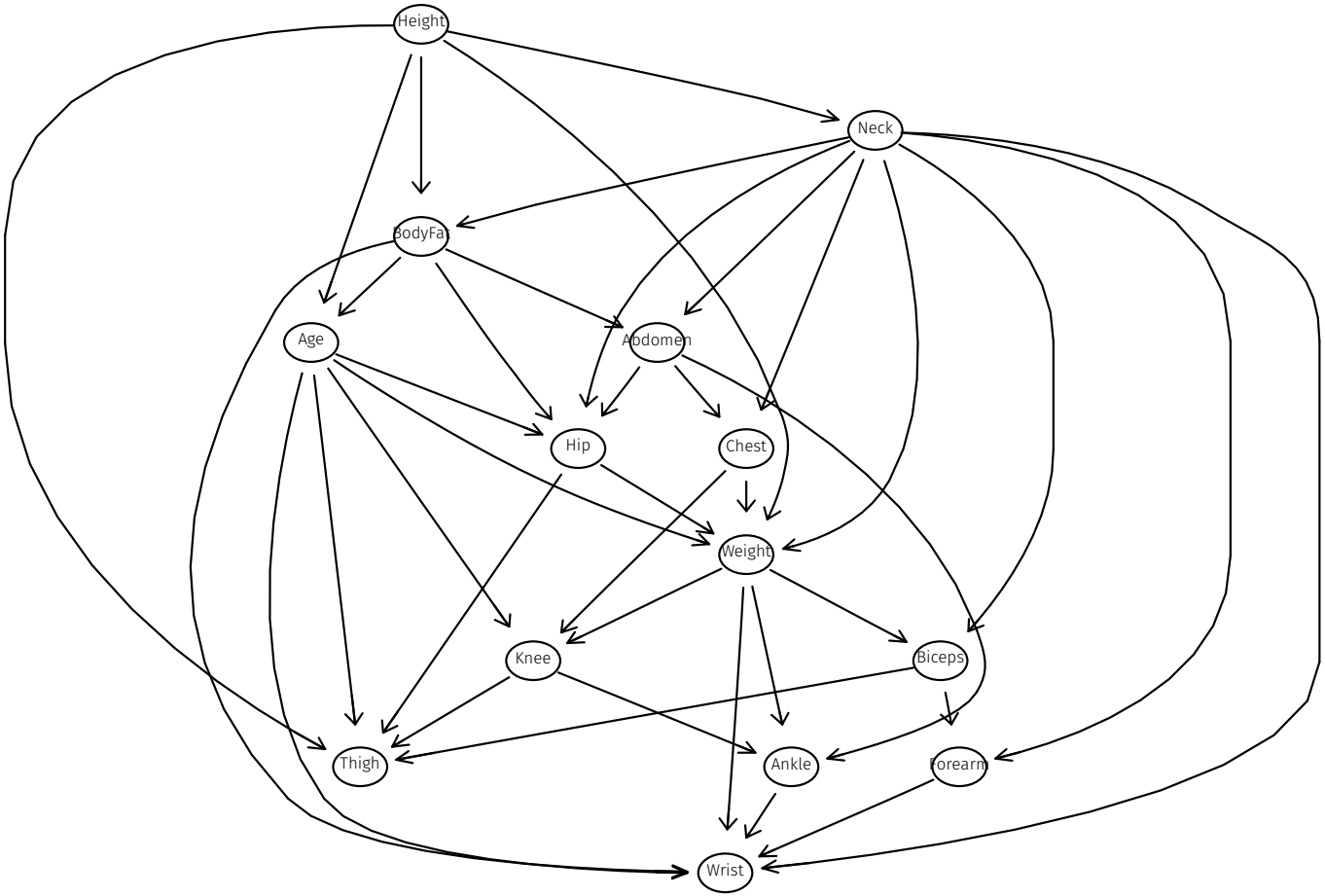
- Implemented in the `hc` function in the `bnlearn` package
- Optimizes a score, for example the BIC.
- The current DAG is compared to all DAGs obtained by adding an edge, removing an edge, or reversing the direction of an edge.
- The process repeats until no score improvement can be made
- If the algorithm get trapped in an equivalence class the score will not change,
  Solution: (a *random restarts*) a pre-specified number of edges are perturbed (added/removed/reversed), and if the resulting model has improved score, then the process restarts at this new model

Example with R

```
> library(bnlearn)
> bn.init = empty.graph(nodes = names(BodyFat)[-1])
> amat(bn.init) = as(pdag.body@graph, "matrix")
> BodyFat$Age=as.double(BodyFat$Age)
> bn.hc <- hc(BodyFat[,-1], bn.init, restart=10, perturb=4)
> m<-fitDag(amat(bn.hc), S = S.body, n = nrow(BodyFat))[c("dev","df")]
> m
$dev
[1] 77.9691

$df
[1] 52
> pchisq(q =  m$dev,df = m$df,lower.tail = F)
[1] 0.01136606
> plot(as(amat(bn.hc),"graphNEL"))
```

Example with R

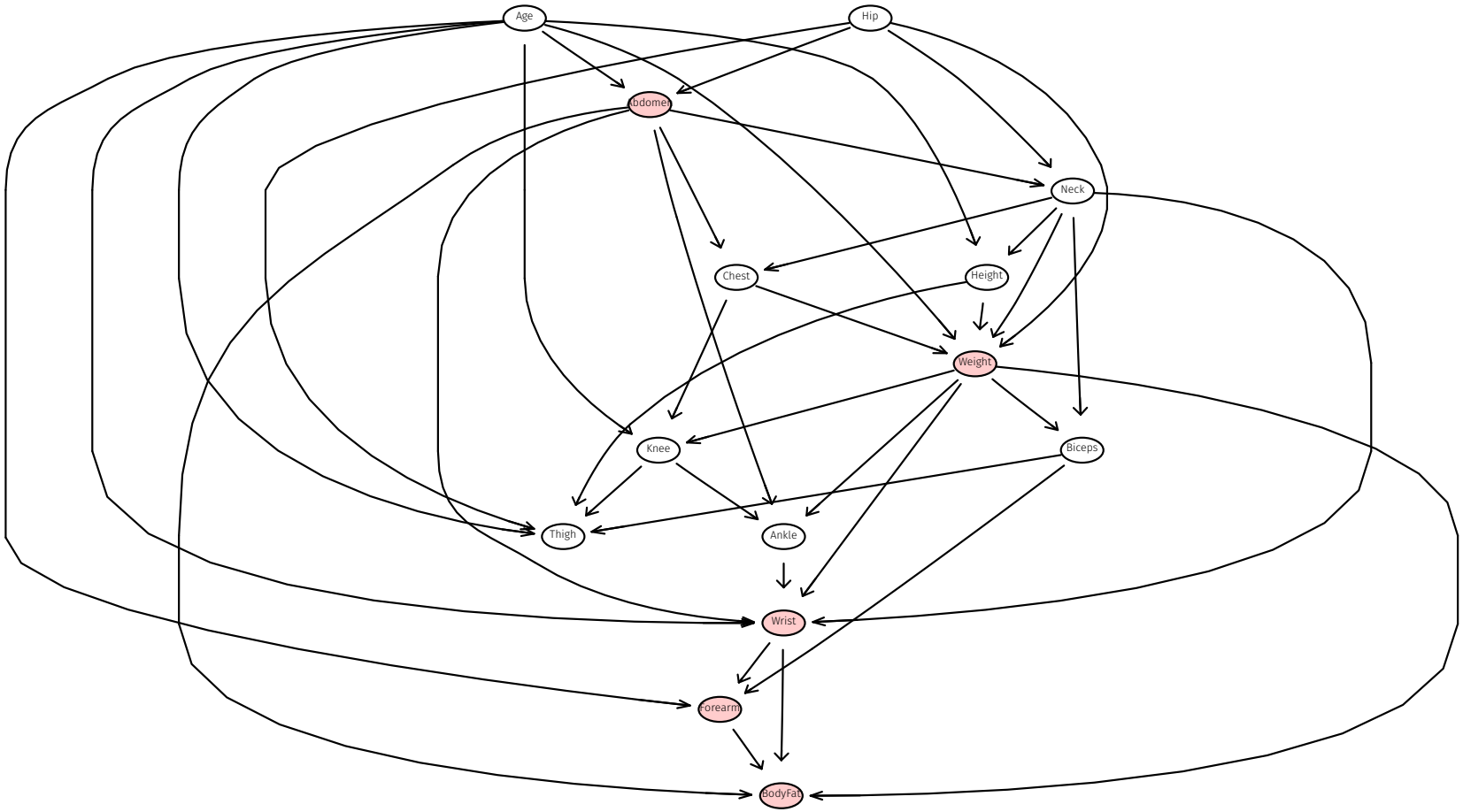


Example with R, with a black list of Edges

```
> bl=data.frame(from=c(colnames(BodyFat)[rep(2,13)],
+                      colnames(BodyFat)[-c(1,2,3)]),
+               to=c(colnames(BodyFat)[-c(1,2)],
+                    colnames(BodyFat)[rep(3,12)]))
> bl[1:2,]
     from     to
1 BodyFat    Age
2 BodyFat Weight
> bn.hcbl <- hc(BodyFat[,-1], bn.init, restart=30,
+               perturb=6, blacklist=bl)
> m<-fitDag(amat(bn.hcbl),S =  S.body,
+           n = nrow(BodyFat))[c("dev","df")]
> m
$dev
[1] 78.42593

$df
[1] 53
> pchisq(q =  m$dev,df = m$df,lower.tail = F)
[1] 0.01318886
> plot(as(amat(bn.hcbl),"graphNEL"))
```

Example with R, with a black list of Edges



Hybrid Algorithm

The max– min hill-climbing algorithm of Tsamardinos et al. (2003),

```
> bn.hcmm <- mmhc(BodyFat[,-1], restart=30,
+                 perturb=6, blacklist=bl)
> bn.hcbl2 <- hc(BodyFat[,-1], bn.hcmm, restart=30,
+                 perturb=6, blacklist=bl)
> m<-fitDag(amat(bn.hcbl2),S =  S.body,
+           n = nrow(BodyFat))[c("dev","df")]
> m
$dev
[1] 66.27016

$df
[1] 53
> pchisq(q =  m$dev,df = m$df,lower.tail = F)
[1] 0.1041402
```

Hybrid Algorithm