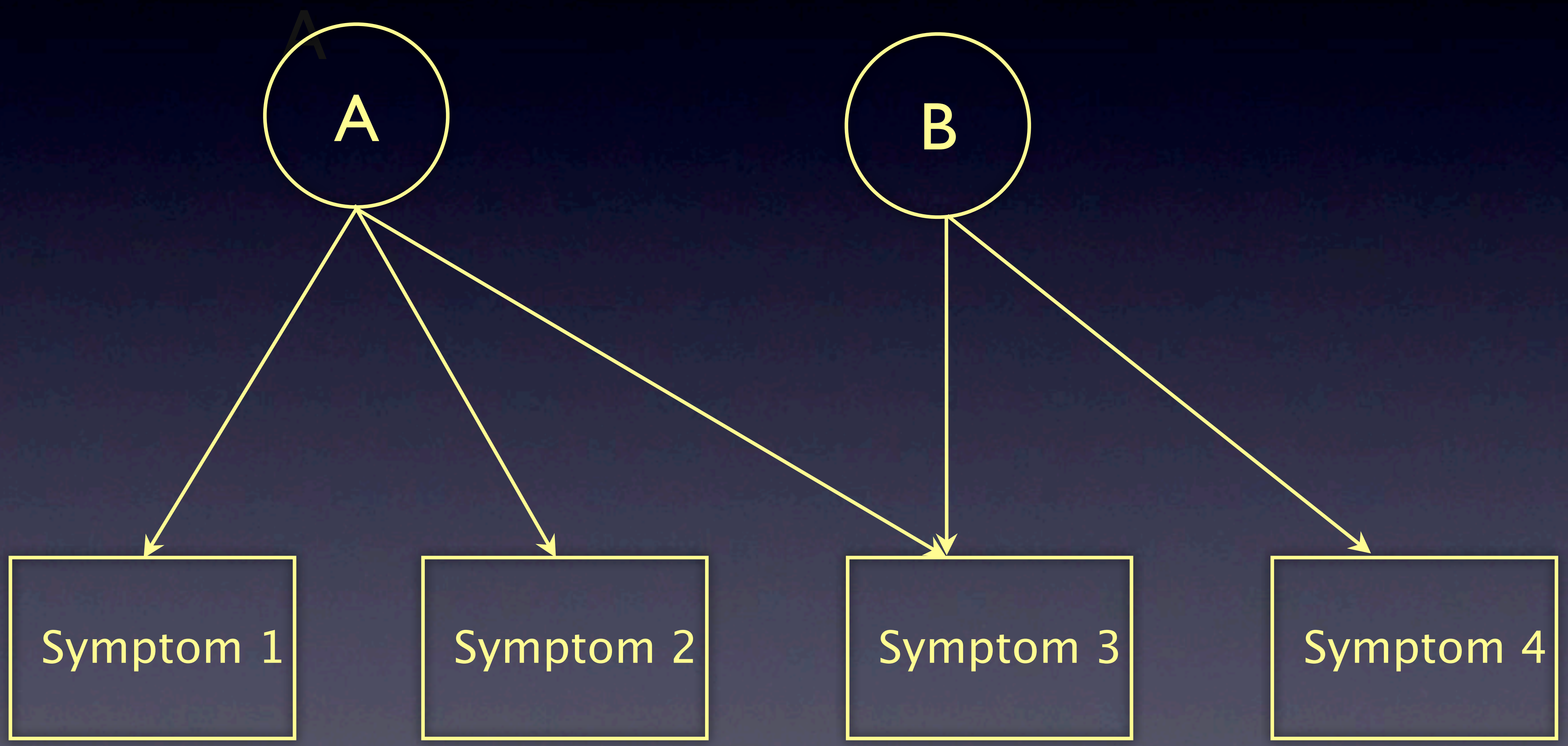
Comorbidity Models:
Analysis of Contingency Tables

Michael C. Neale
NIDA Advanced Genetic Epidemiology Workshop
October 22 2012
Richmond Virginia USA

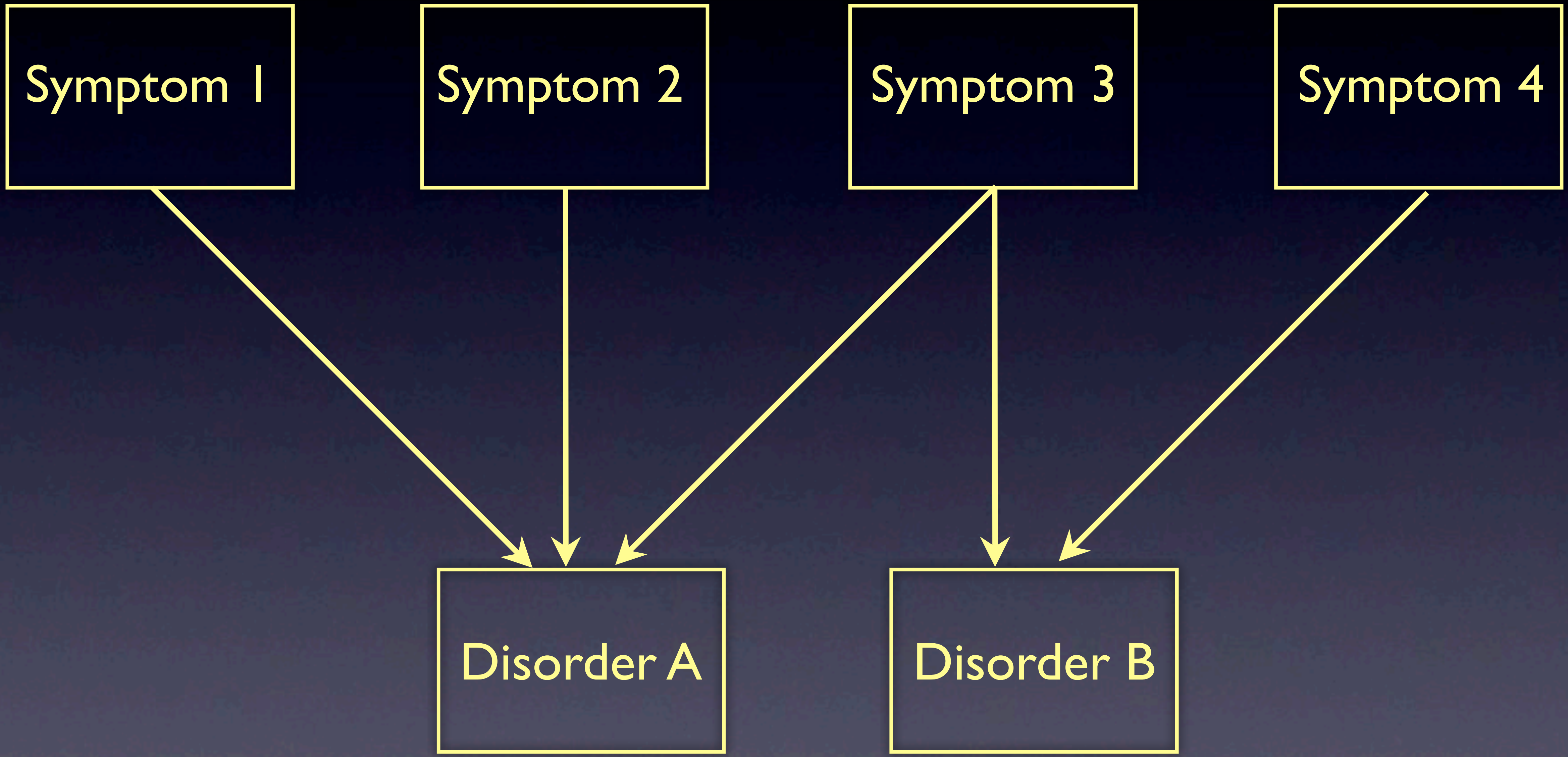
Overview

- Substance Use & Disorder Diagnoses:
  - Binary phenotypes
    - Highly comorbid
    - Psychiatric disorders
- ACE model is but one of many
- Two twins, two binary variables
  - —16 outcome combinations
- Fit models by maximum likelihood
  - —(alternatives exist)

Pure forms of two disorders A & B generate some of the same symptoms

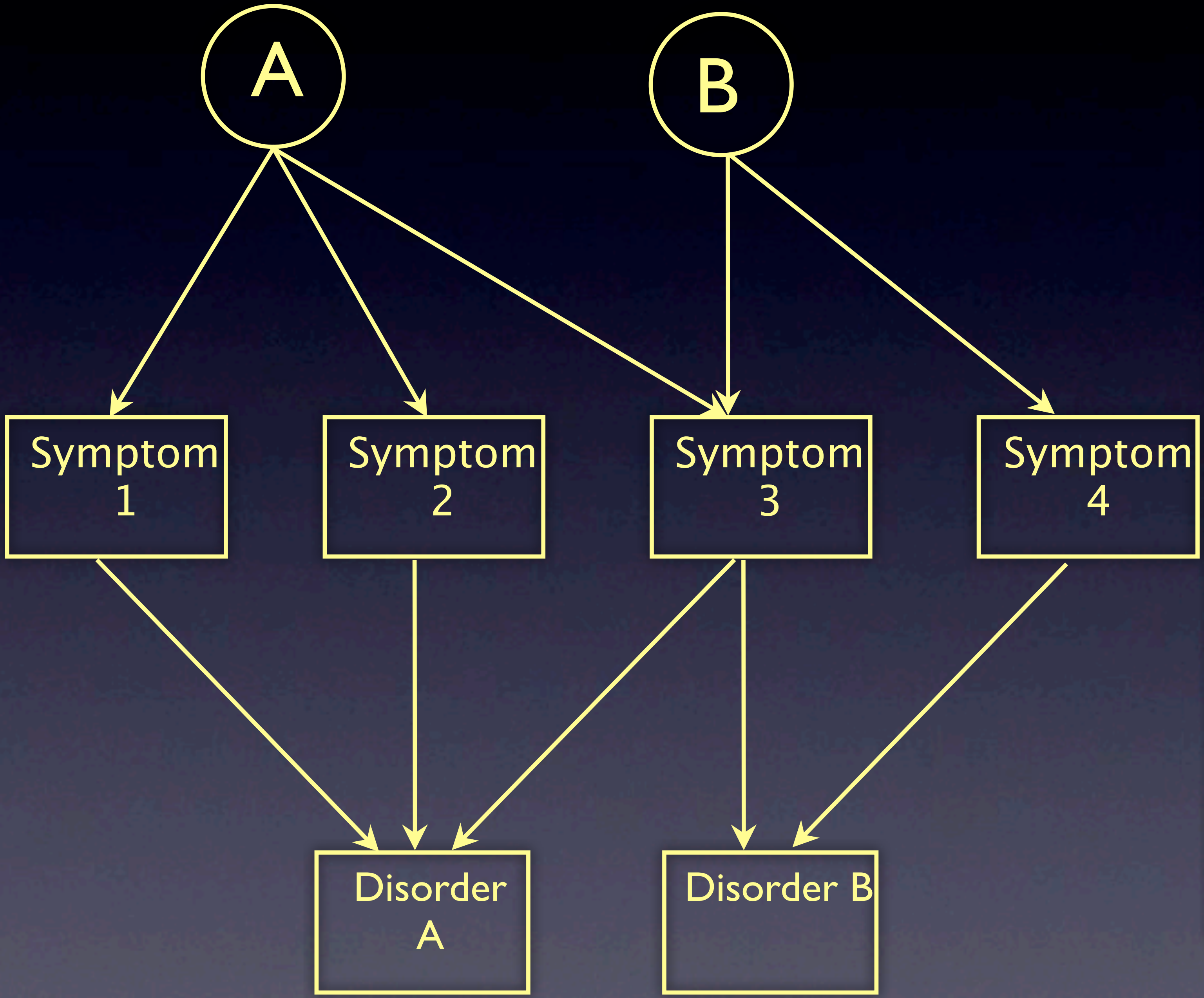


Assessments of disorders A & B share symptoms



Cramer, Waldrop, Van der Maas, Borsboom (2010) Comorbidity: A network perspective. Brain Behavior Sciences 33:137-

Comorbidity due to symptom sharing



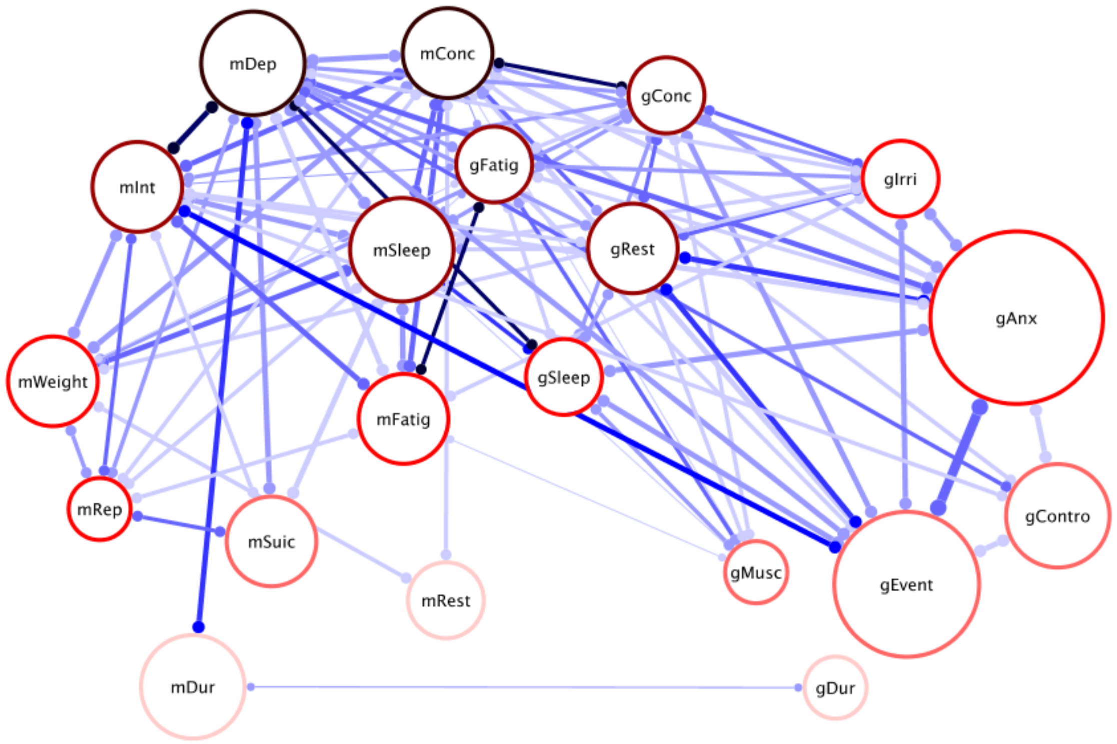


Figure 4. A comorbidity network for MDD and GAD. Larger nodes represent more frequent symptoms; darker circumference, higher centrality; thicker edges, higher frequency of co-occurrence; darker edges, stronger associations. Only edges with a log odds ratio higher than (-) 0.60 are represented. Centrally positioned nodes (*mConc*, *gConc*, *mSleep*, *gSleep*, *mFatig*, *gFatig, mRest* and *gRest*) represent overlapping symptoms. Non-overlapping MDD symptoms are displayed on the left the figure, non-overlapping GAD symptoms on the right.

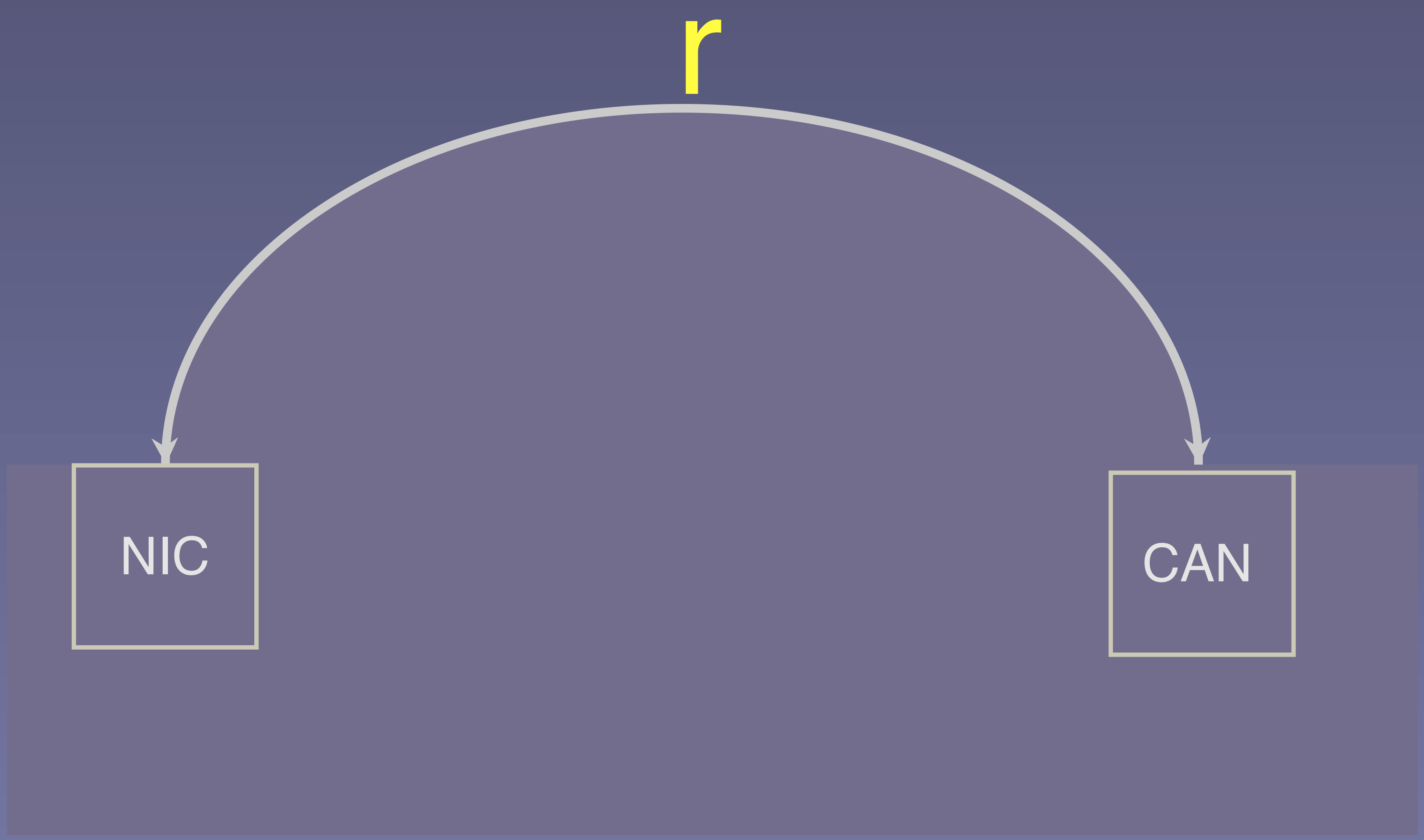
Not today!

Comorbidity

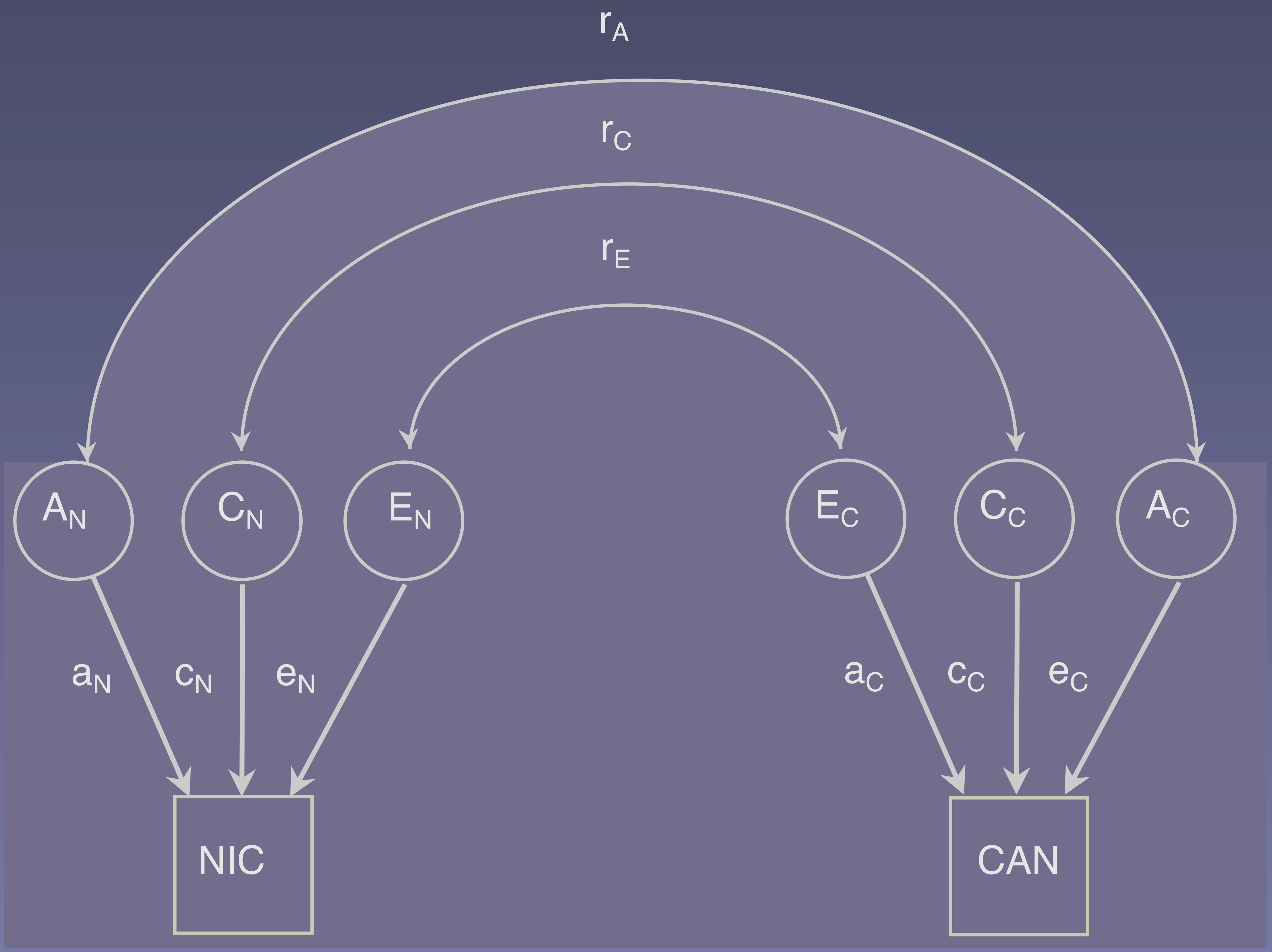
A correlation between (binary) traits

Neale & Kendler (1995) 13 Models

Based on Klein & Riso (1994)

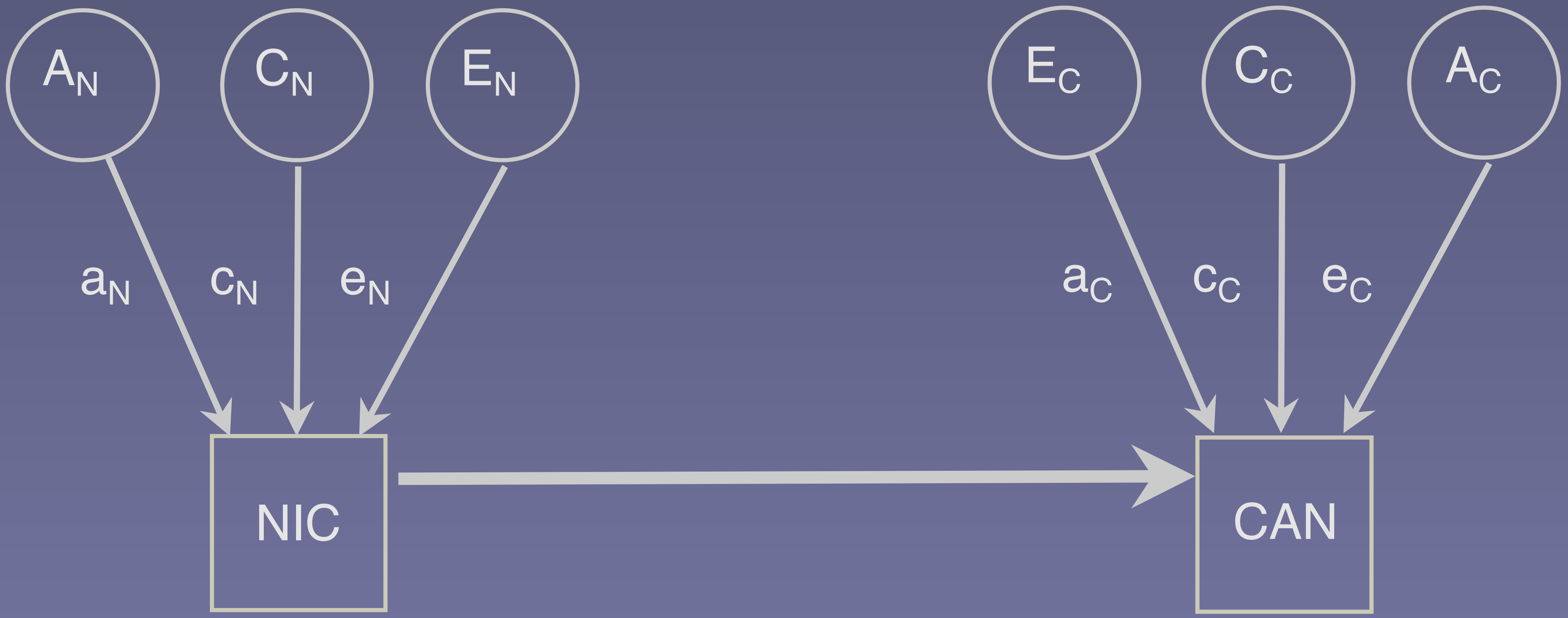


Partitioning Comorbidity



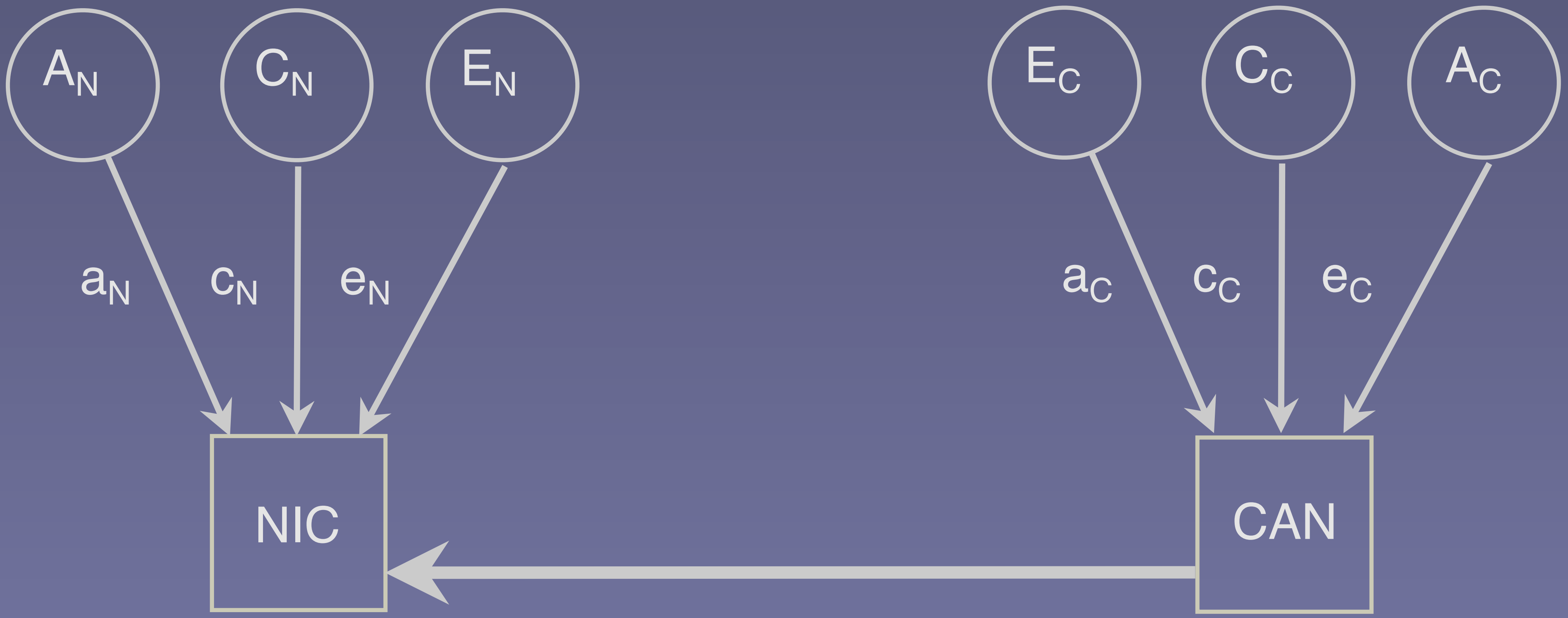
Modeling Comorbidity

Nicotine Use Causes Cannabis Use



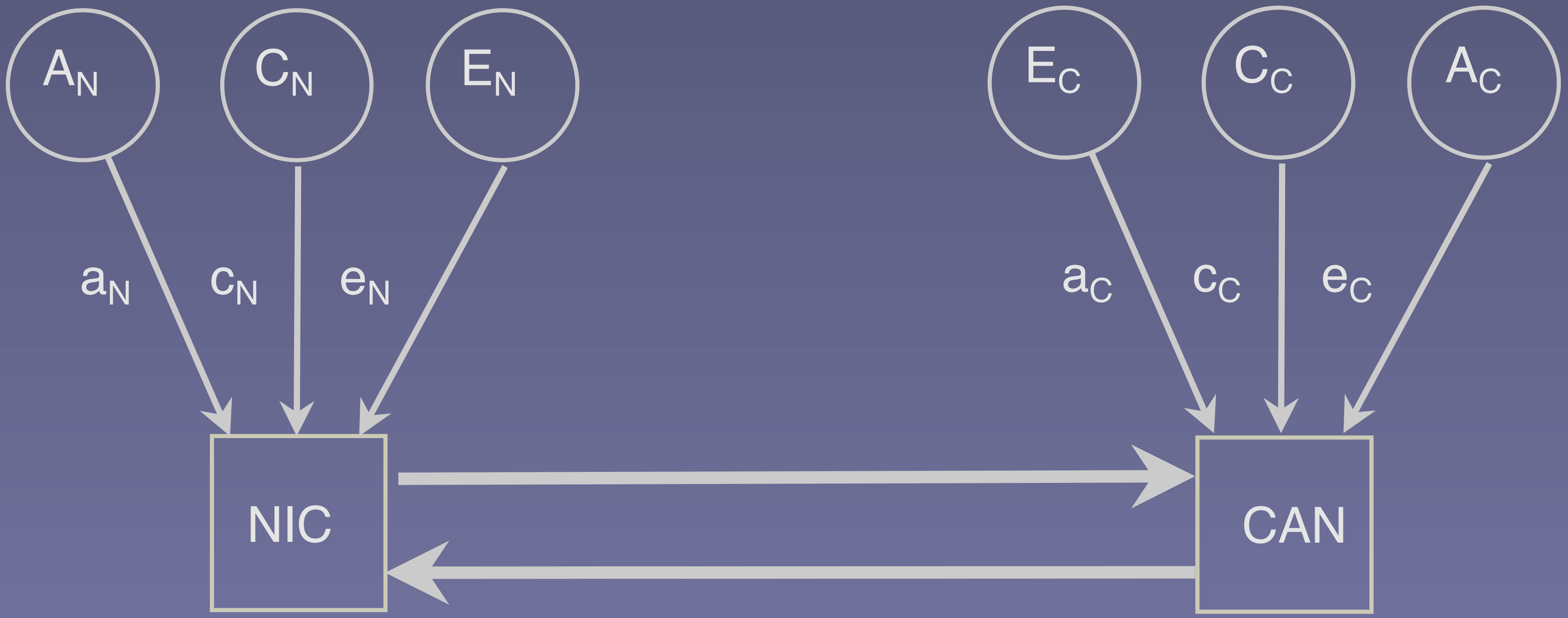
Modeling Comorbidity

Cannabis Use Causes Nicotine Use



Modeling Comorbidity

Reciprocal Causation



Multinomial Likelihood

If data x_i, i=1...n fall into k different classes with probability p_j, j=1...k, and n_j is the number falling into class i,

$$\text{Likelihood} = \prod_{j=1}^{k} p_j^{\,nj}$$

$$\text{Log-likelihood} = \sum_{j=1}^{k} n_j \log(p_j)$$

MLE of proportion is observed proportion

Directly obtain MLE of p_j j=1...n as n_j/n

Data Tabulation

Twin 2		No		Yes	
Twin 1		No	Yes	No	Yes
No	No	----	---+	--+-	--++
	Yes	-+--	-+-+	-++-	-+++
Yes	No	+---	+--+	+-+-	+-++
	Yes	++--	++-+	+++-	++++

Symmetric

Twin 2 Twin 1		No		Yes	
		No	Yes	No	Yes
No	No	----	---+	--+-	--++
	Yes	-+--	-+-+	-++-	-+++
Yes	No	+---	+--+	+-+-	+-++
	Yes	++--	++-+	+++-	++++

10 Flavors: Sum across Diagonal

Twin 2 Twin 1		No		Yes	
		No	Yes	No	Yes
No	No	----	---+	--+-	--++
	Yes		-+-+	-++-	-+++
Yes	No			+-+-	+-++
	Yes				++++

----

---+

--+-

--++

-+-+

-++-

-+++

+-+-

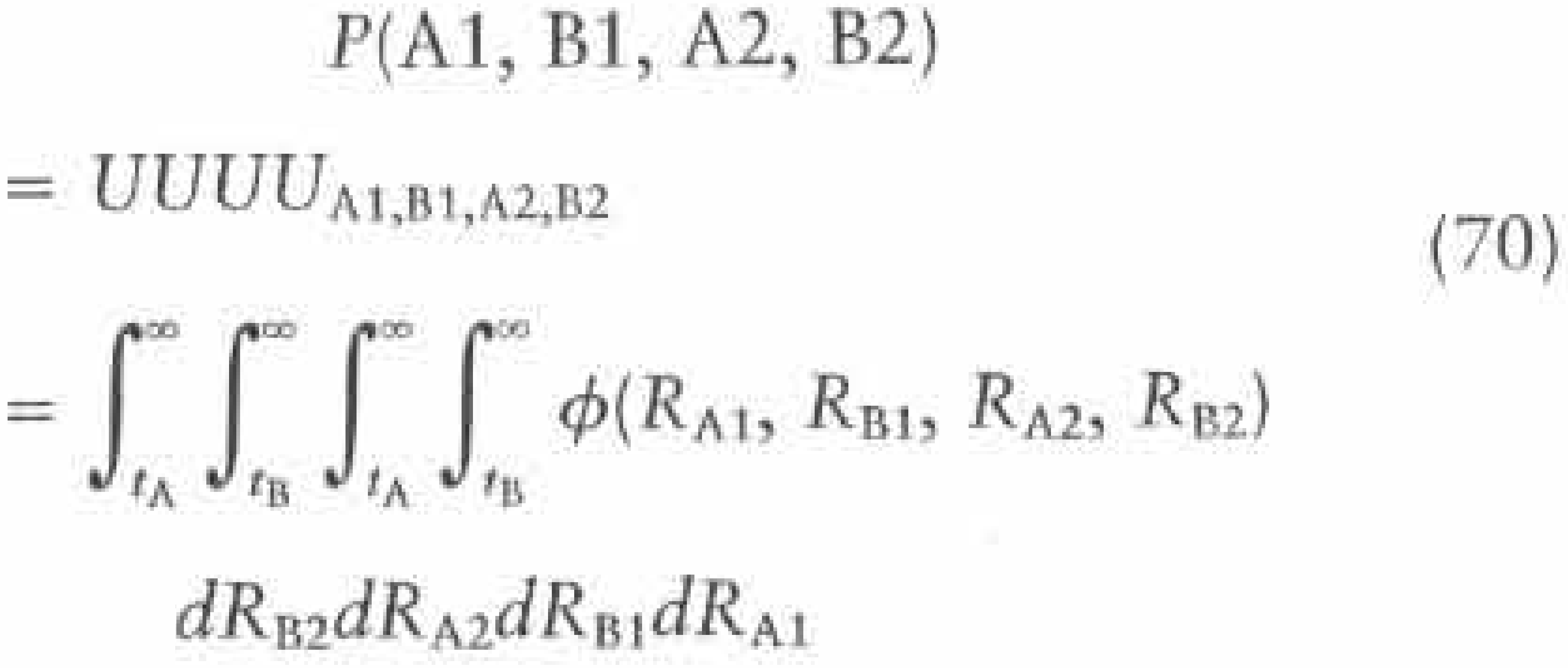
+-++

++++

Multinomial Likelihood R function

```r
mzFreqs <- c(141, 35, 32, 25, 15,  7, 33, 39, 47, 124)
dzFreqs <- c(58, 18, 27, 44, 7, 6, 33, 15, 38, 81)
satLogLike <- function(x) {
    proportions <- x/sum(x)
    logliks <- (log(proportions)*x)
    neg2ll <- -2*sum(logliks)
    return(list(neg2lnL=neg2ll))
}
satlnLmz <-  satLogLike(mzFreqs)
satlnLdz <-  satLogLike(dzFreqs)
satlnL <- satlnLmz$neg2lnL + satlnLdz$neg2lnL
```

Correlated Liabilities



$$
\begin{aligned}
&P(\mathrm{A1}, \mathrm{B1}, \mathrm{A2}, \mathrm{B2}) \\
&= UUUU_{\mathrm{A1,B1,A2,B2}} \\
&= \int_{t_\mathrm{A}}^{\infty}\int_{t_\mathrm{B}}^{\infty}\int_{t_\mathrm{A}}^{\infty}\int_{t_\mathrm{B}}^{\infty} \phi(R_{\mathrm{A1}}, R_{\mathrm{B1}}, R_{\mathrm{A2}}, R_{\mathrm{B2}}) \\
&\quad dR_{\mathrm{B2}}dR_{\mathrm{A2}}dR_{\mathrm{B1}}dR_{\mathrm{A1}}
\end{aligned}
\qquad (70)
$$

Inherent in OpenMx Ordinal Data Analysis

We can do it by hand as well

Bivariate Threshold Model

Contingency Table with 4 observed cells:

cell a: pairs concordant for unaffected

cell d: pairs concordant for affected

cell b/c: pairs discordant for the disorder

Twin1 Twin2	0	1
0	a	b
1	c	d

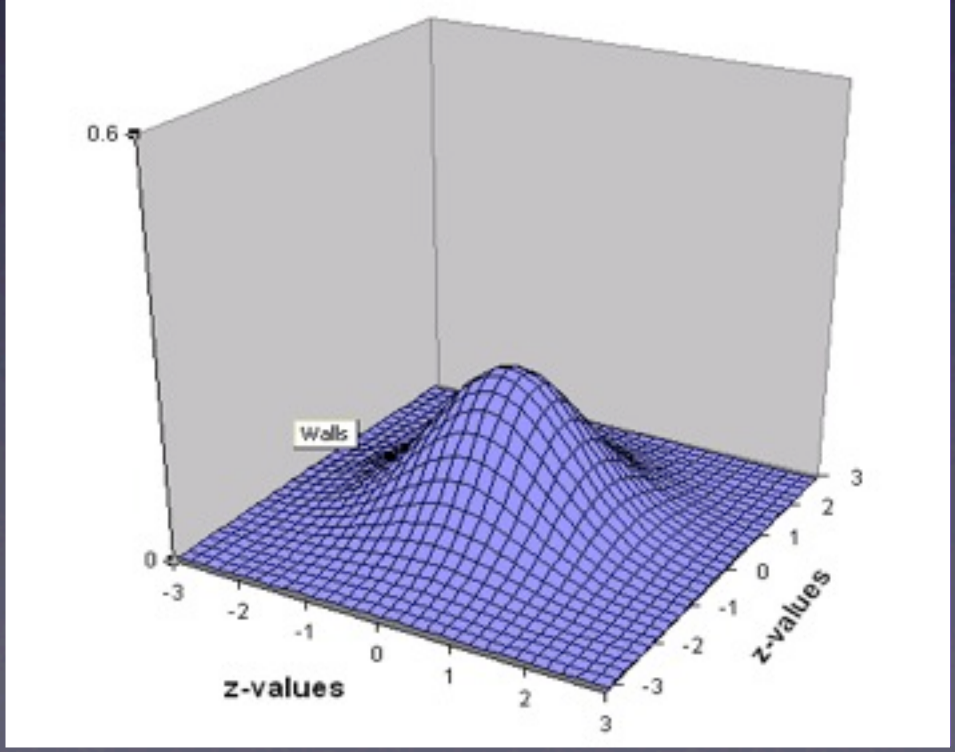
0 = unaffected
1 = affected

Joint Liability Model for Twin Pairs

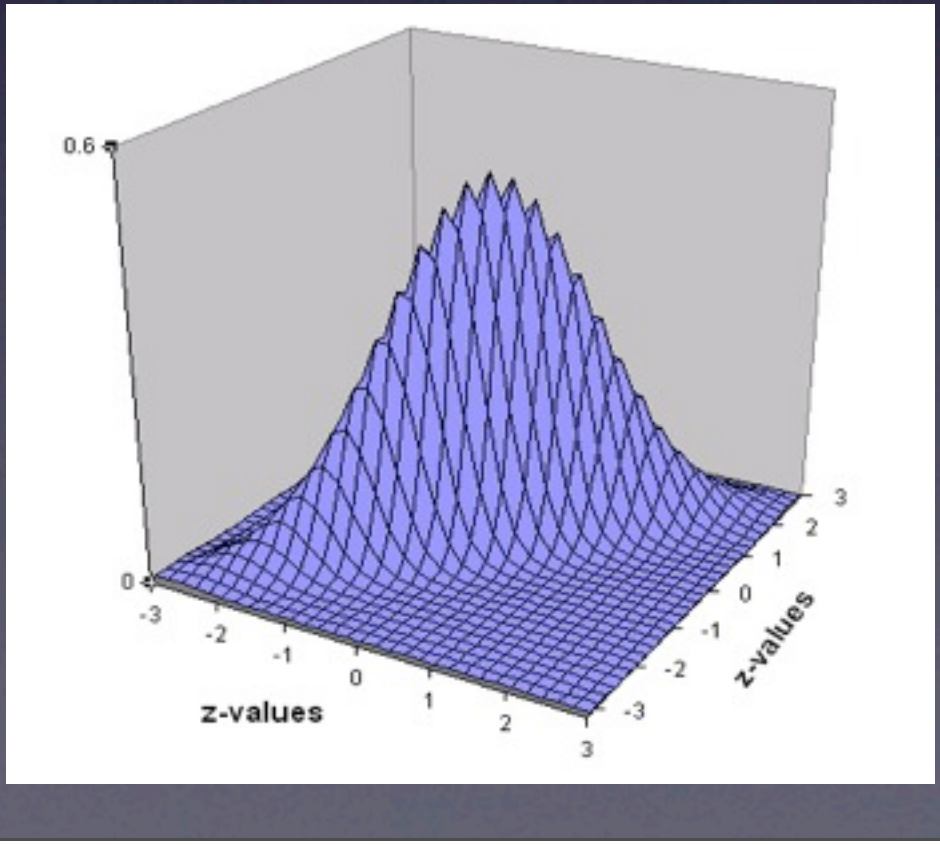
- Assumed to follow a **bivariate normal** distribution, where both traits have a mean of 0 and standard deviation of 1, but the **correlation** between them is variable.

- The **shape** of a bivariate normal distribution is determined by the **correlation** between the traits

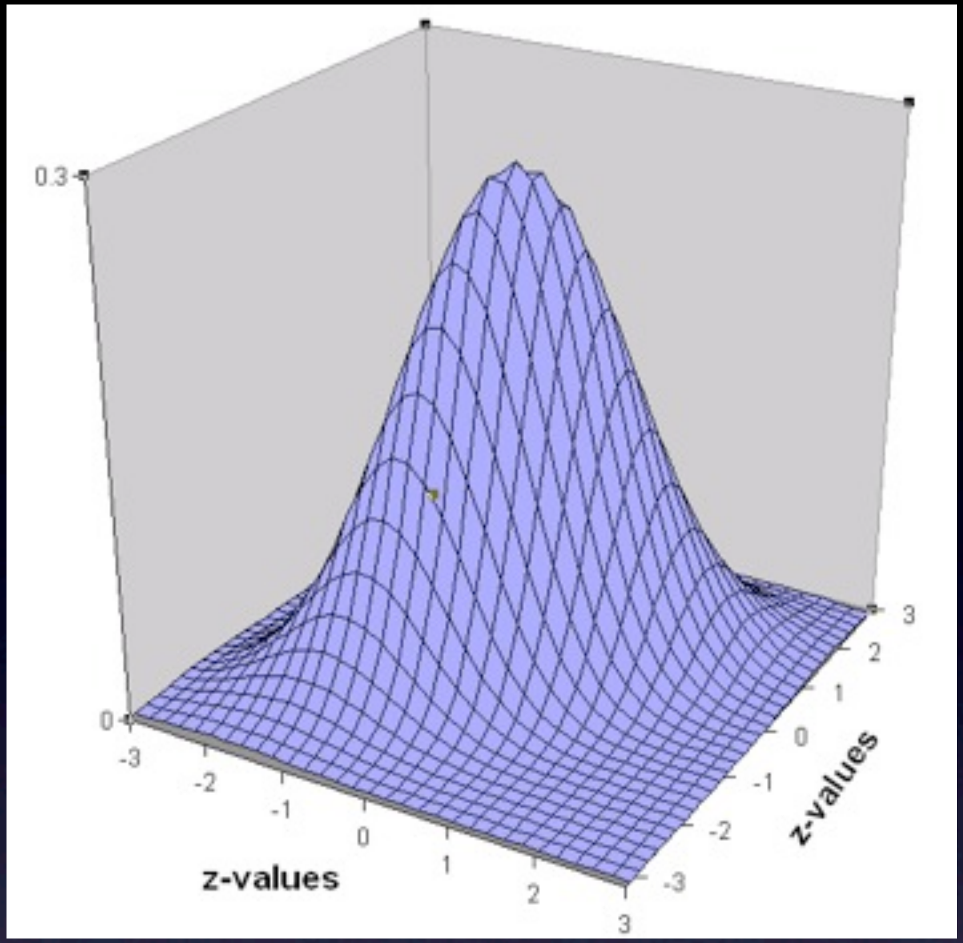
***r* =.00**



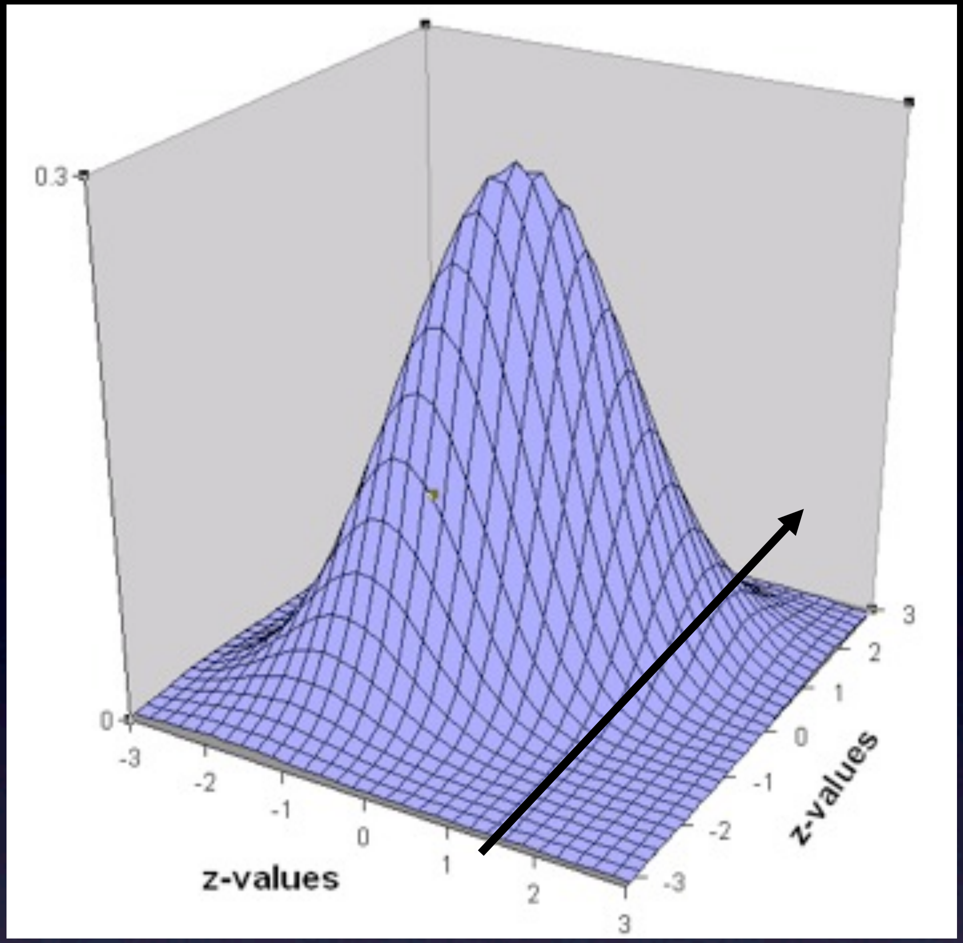
***r* =.90**



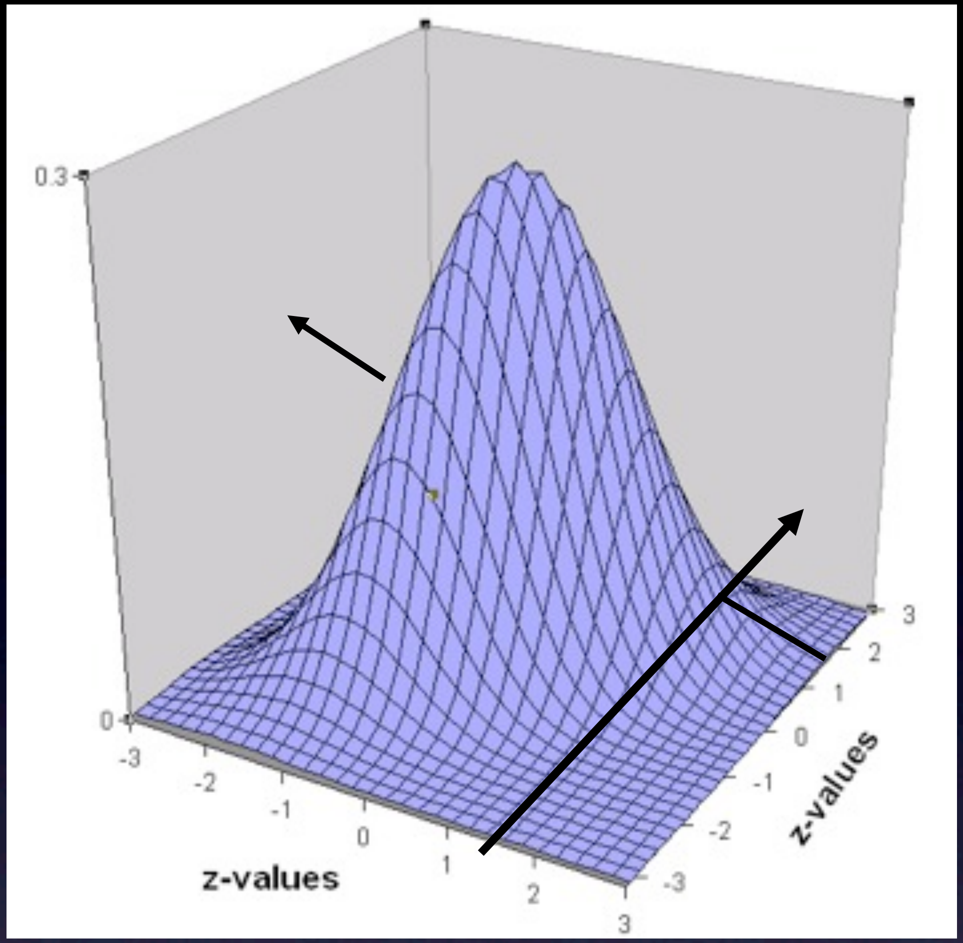
Bivariate Normal (R=0.6) partitioned at threshold 1.4 (z-value) on both liabilities



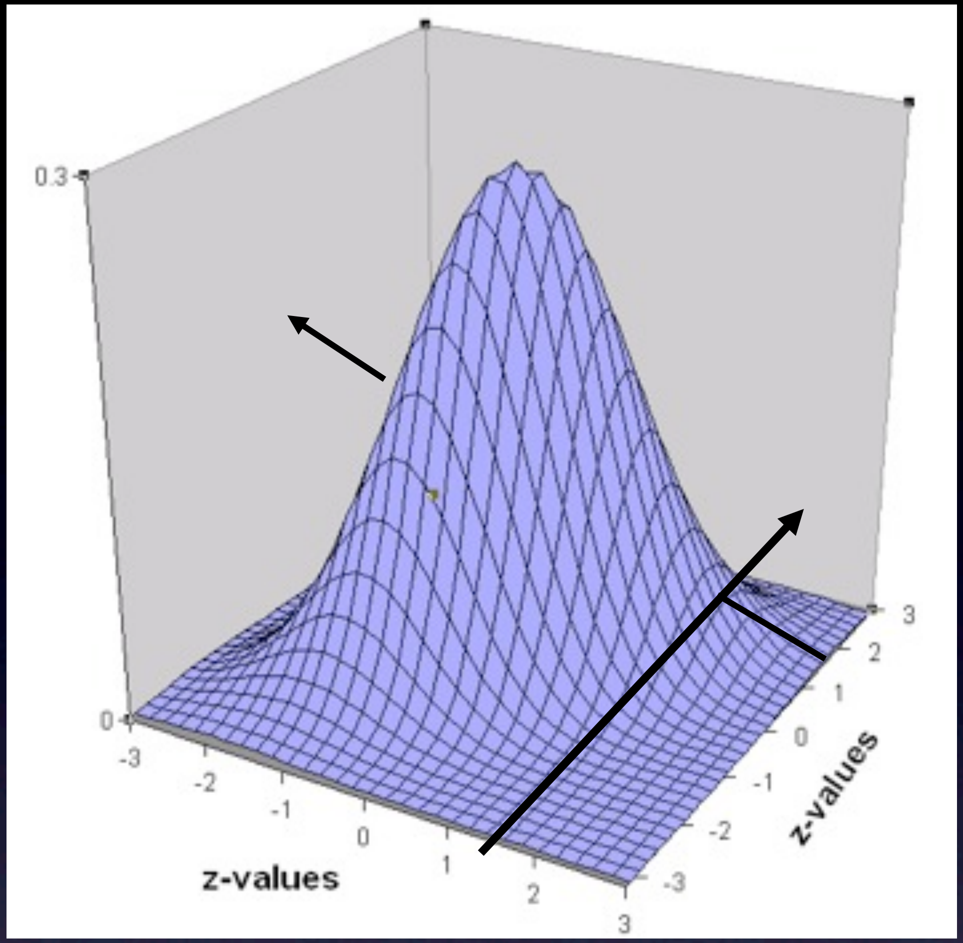
Bivariate Normal (R=0.6) partitioned at threshold 1.4 (z-value) on both liabilities

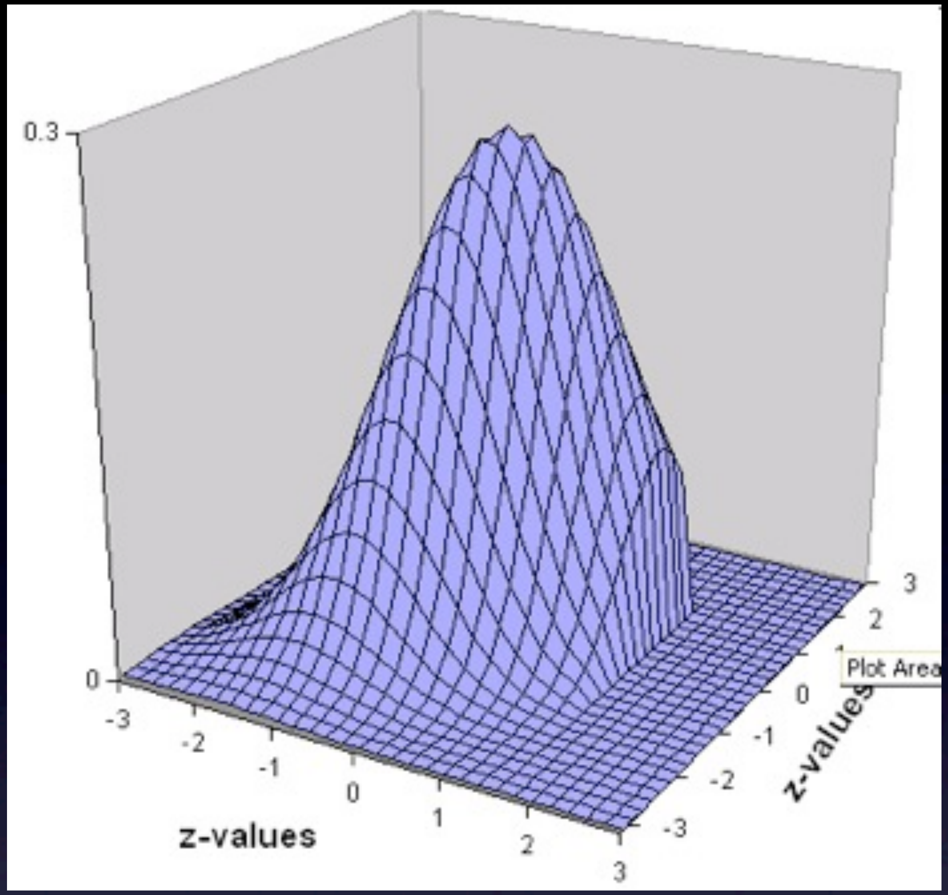


Bivariate Normal (R=0.6) partitioned at threshold 1.4 (z-value) on both liabilities

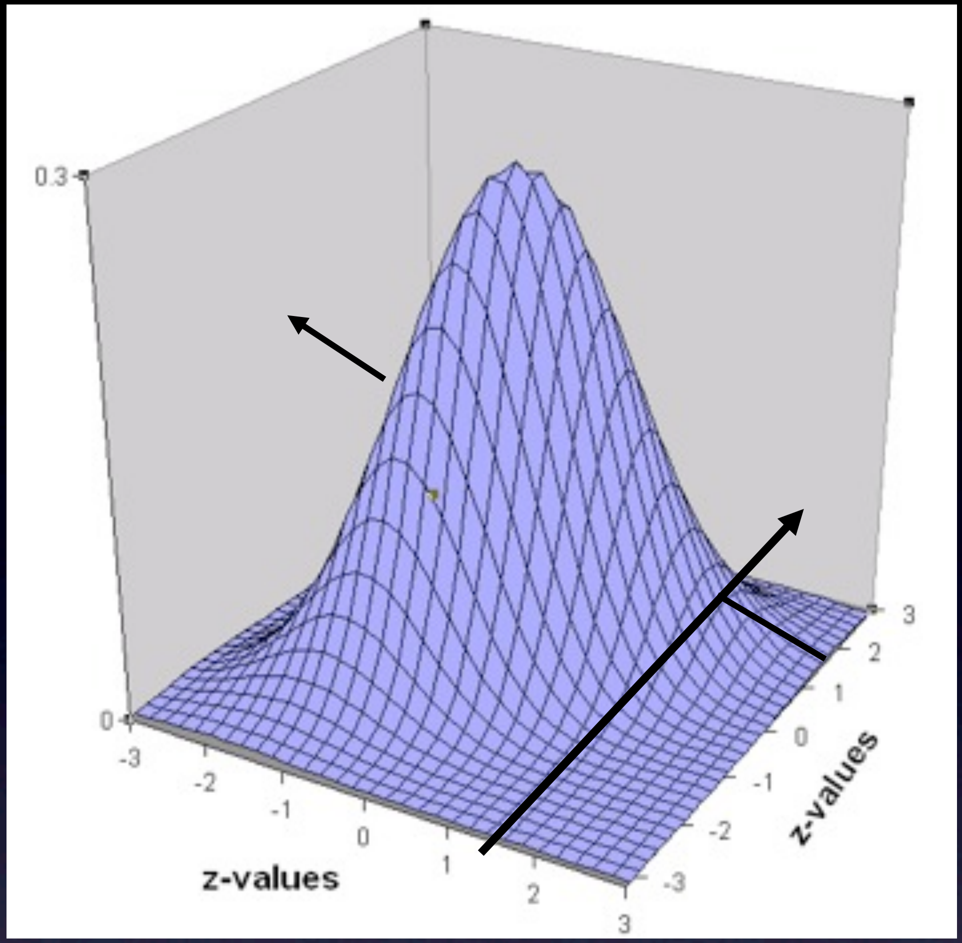


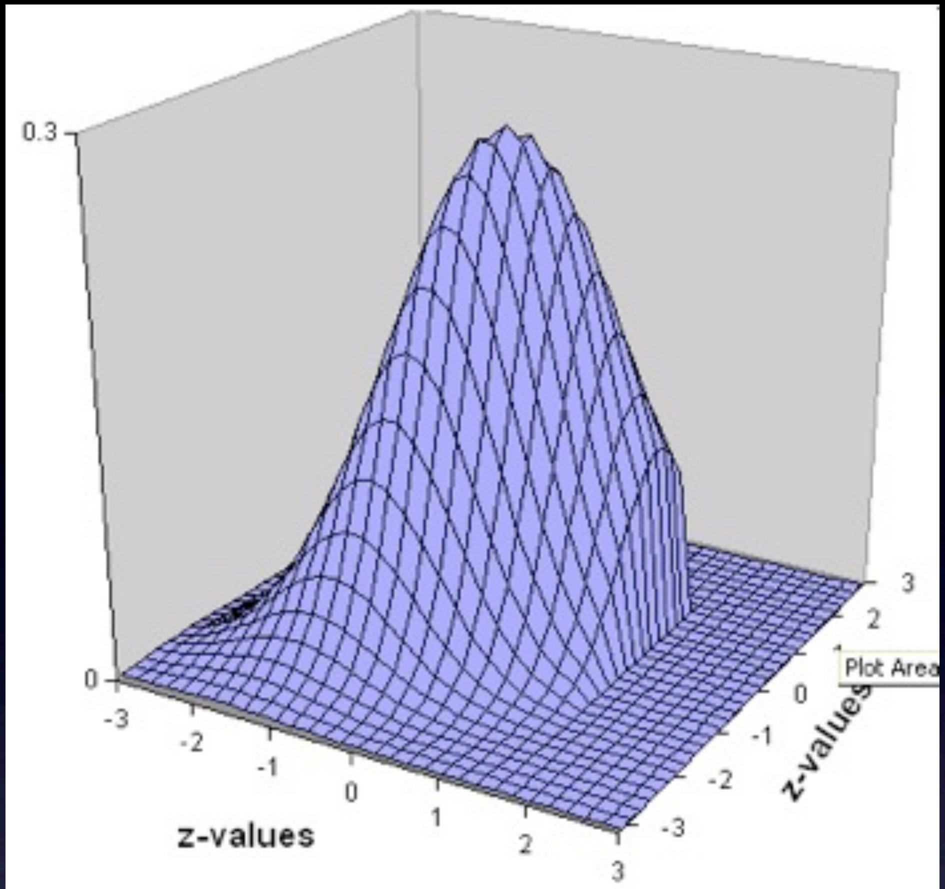
Bivariate Normal (R=0.6) partitioned at threshold 1.4 (z-value) on both liabilities

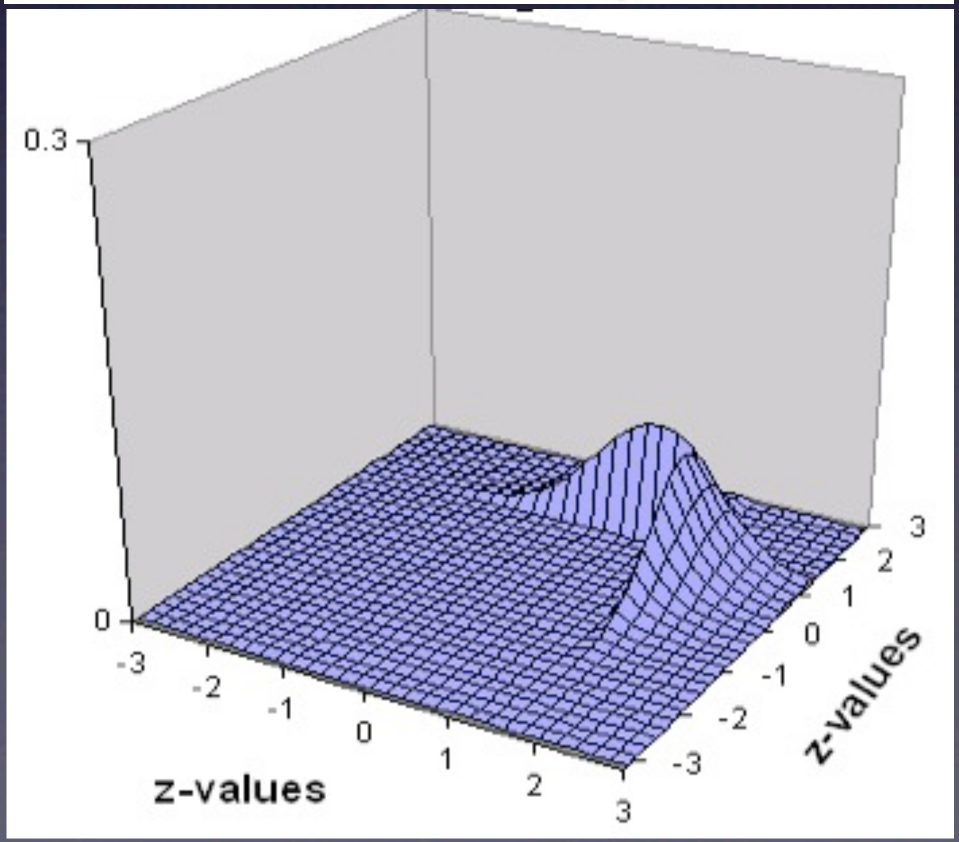




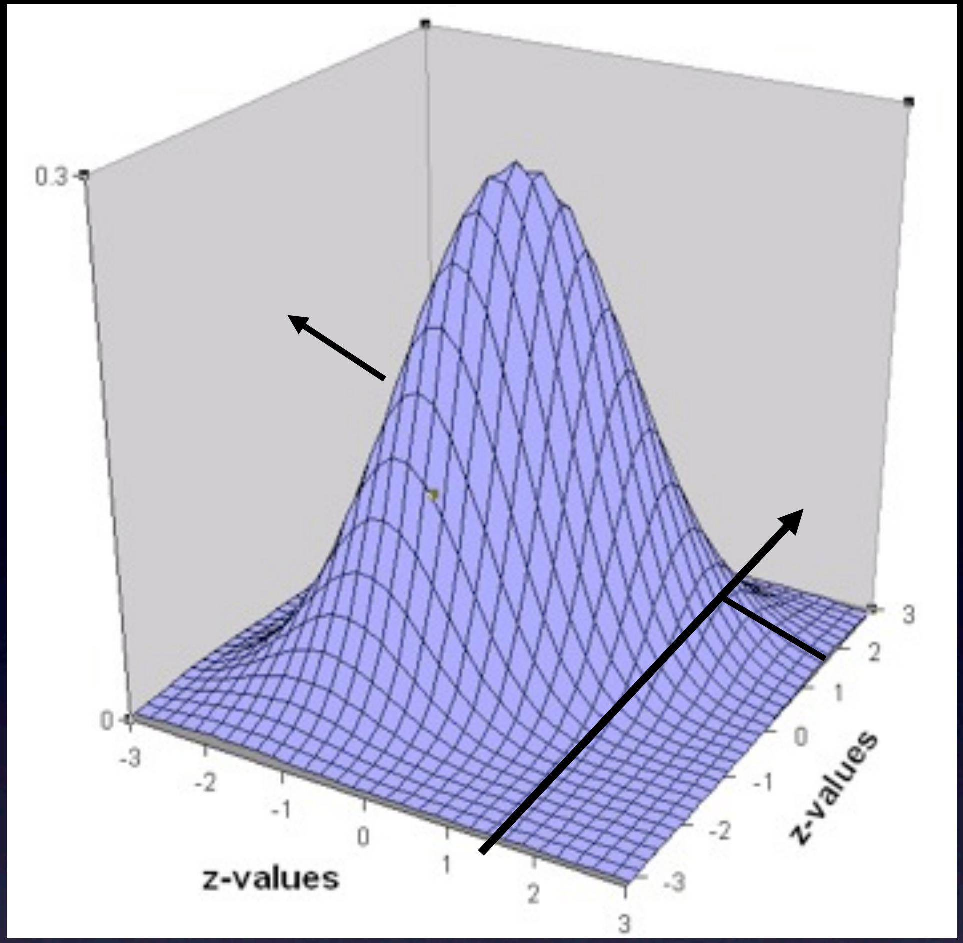
Bivariate Normal (R=0.6) partitioned at threshold 1.4 (z-value) on both liabilities

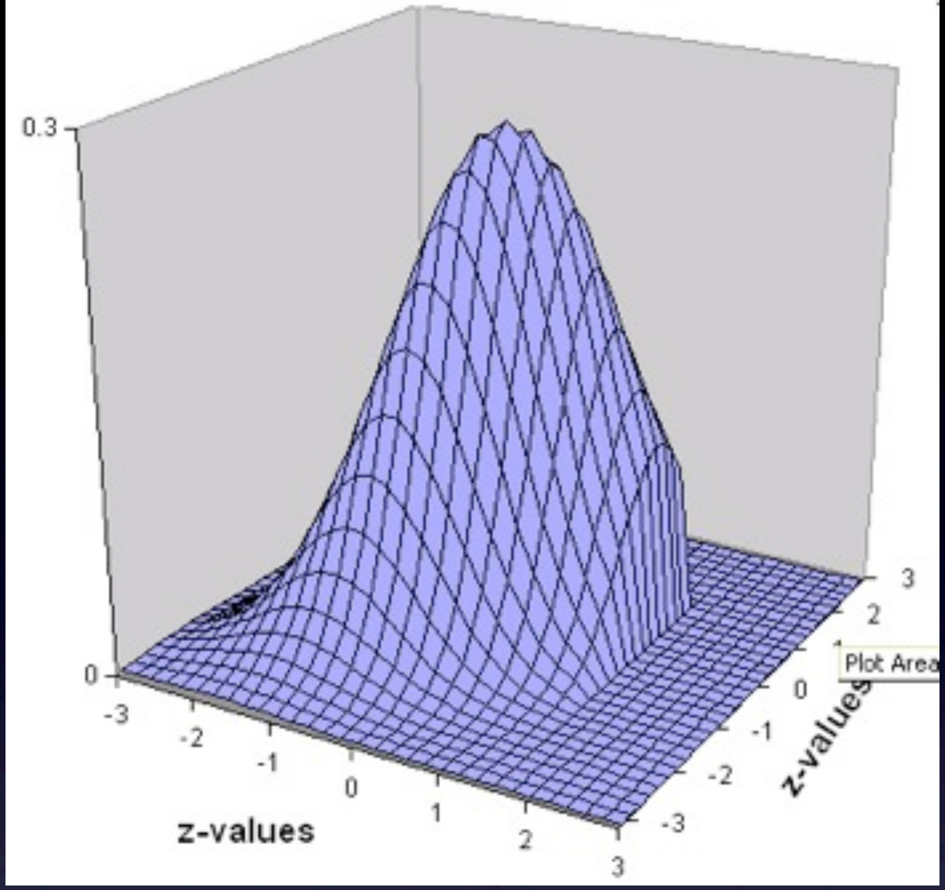


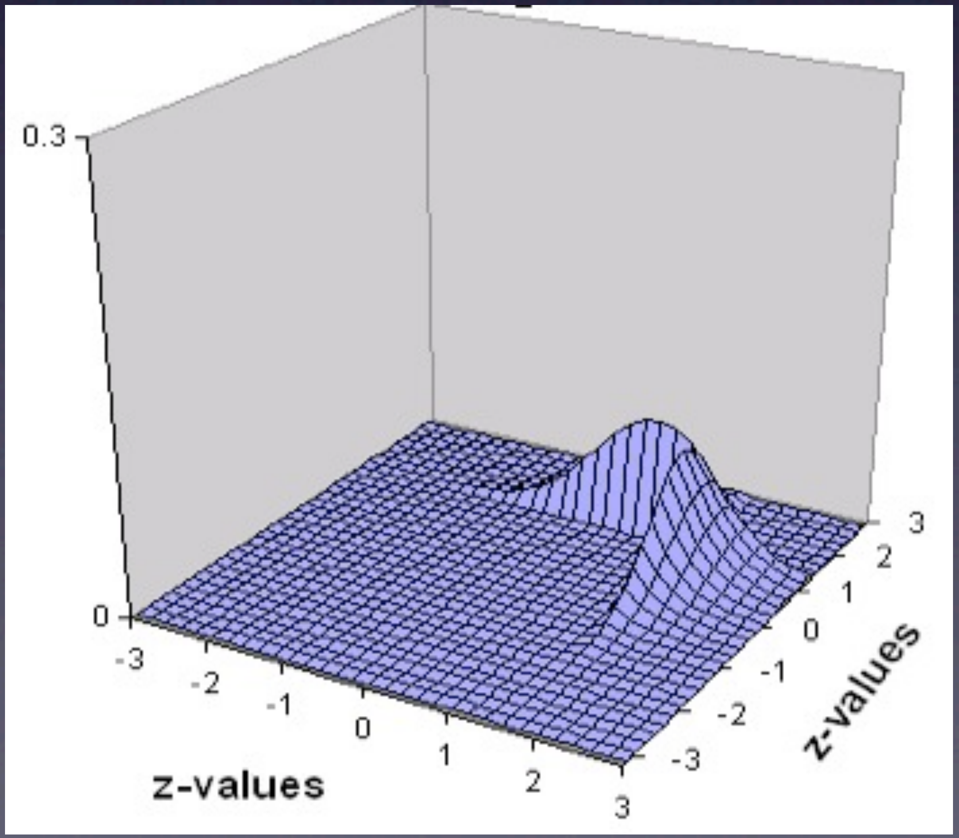


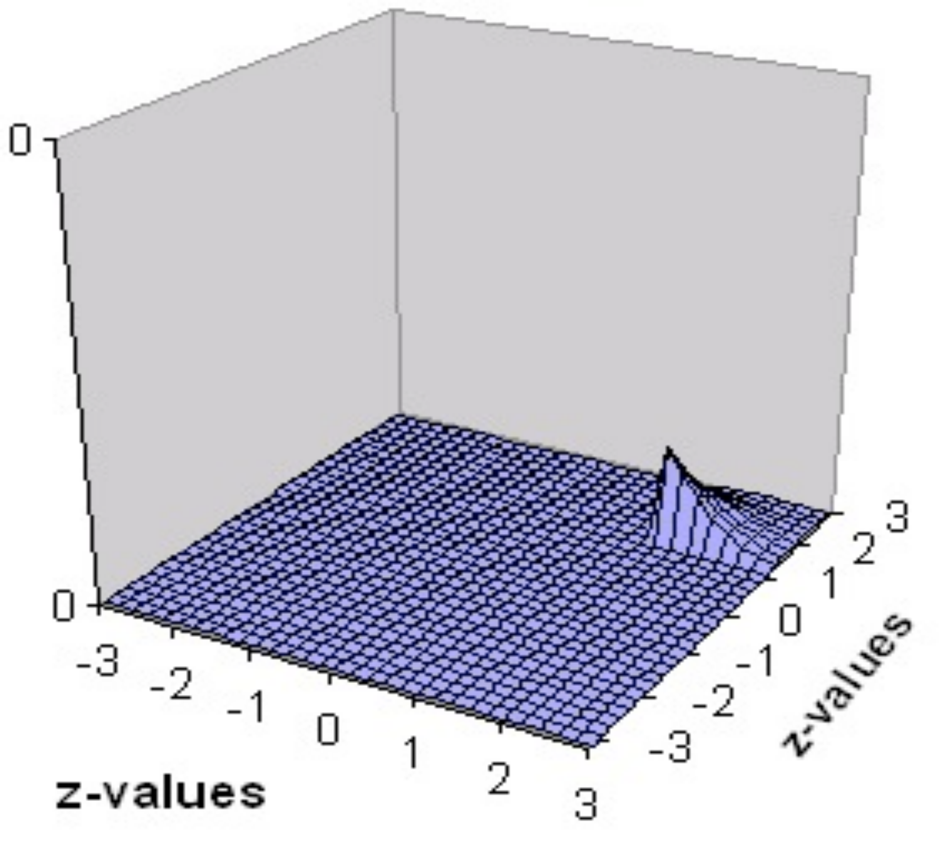


Bivariate Normal (R=0.6) partitioned at threshold 1.4 (z-value) on both liabilities



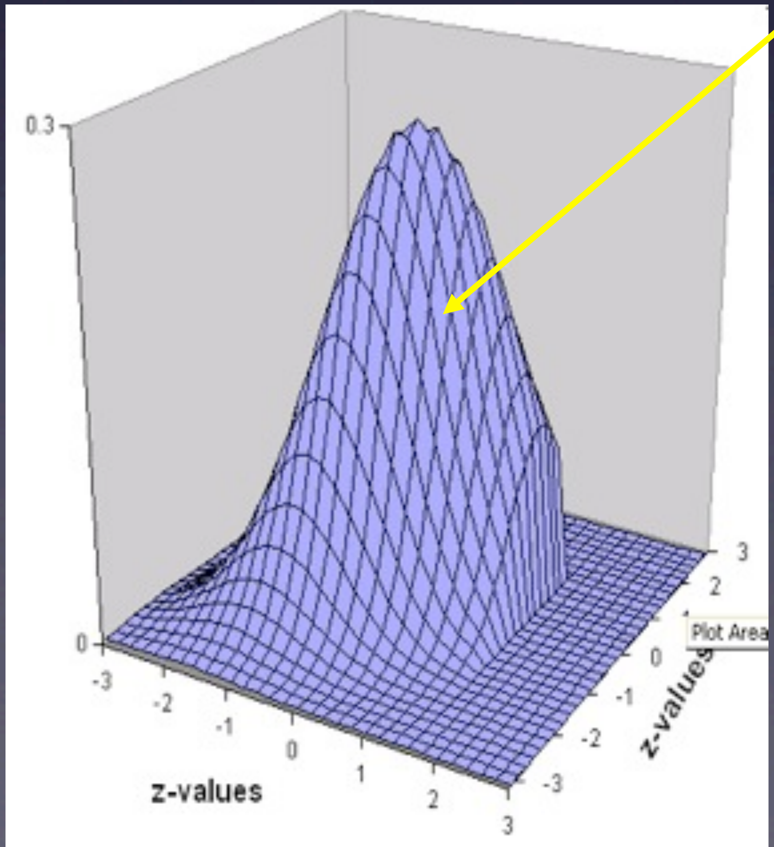


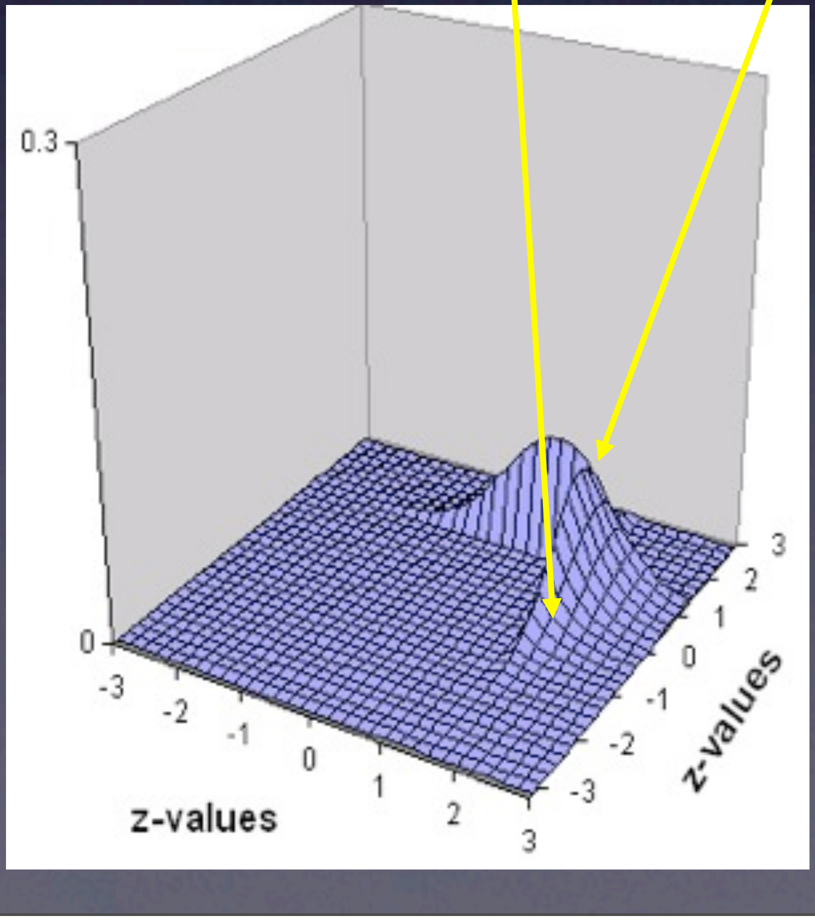


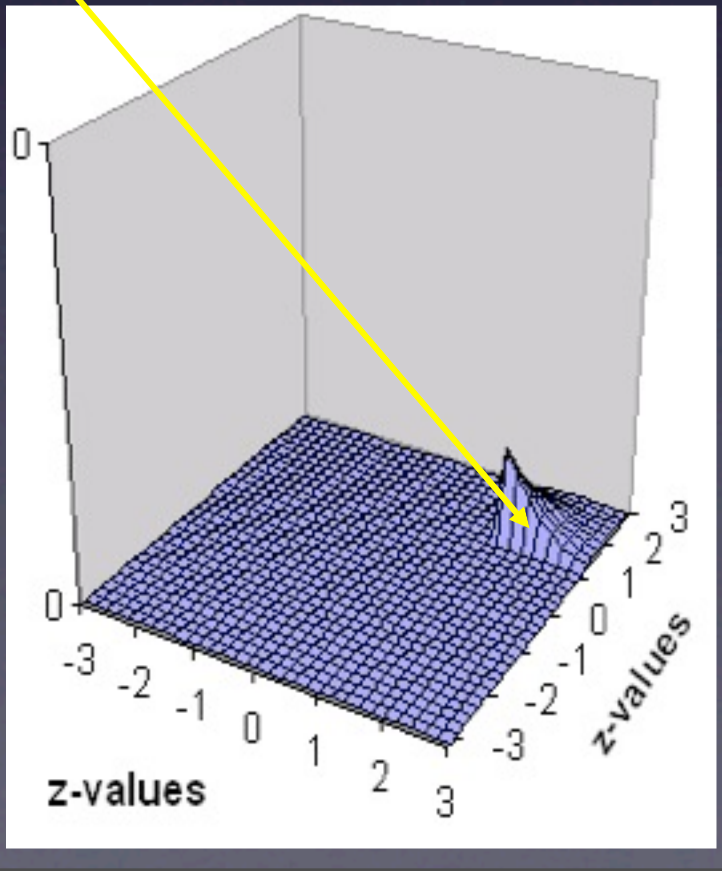


Expected Proportions of the BN, for R=0.6, Th1=1.4, Th2=1.4

Liab 2 Liab 1	0	1
0	.87	.05
1	.05	.03







DOC Model by hand

```
# Make a little labeling function
labFun <- function(name="matrix",nrow=1,ncol=1)
            {
                matlab <- matrix(paste(rep(name, each=nrow*ncol), rep(rep(1:nrow),ncol), rep(1:ncol,each=nrow),sep="_"))
                return(matlab)
            }

# Matrices a, c, and e to store a, c, and e path coefficients
aMat    <-    mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=.6, labels=labFun("a",nv,nv), name="a" )
cMat    <-    mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=.6, labels=labFun("c",nv,nv), name="c" )
eMat    <-    mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=sqrt(.28), labels=labFun("e",nv,nv), name="e" )
BMat    <-    mxMatrix( type="Full", nrow=nv, ncol=nv, free=c(F,T,F,F), labels=labFun("B",nv,nv), name="B")

# Algebras A, C, and E compute variance components
AAlg    <-    mxAlgebra( expression=a %*% t(a), name="A" )
CAlg    <-    mxAlgebra( expression=c %*% t(c), name="C" )
EAlg    <-    mxAlgebra( expression=e %*% t(e), name="E" )

# Algebra to compute total variances and standard deviations (diagonal only)
VAlg    <-    mxAlgebra( expression=diag2vec(A+C+E), name="V" )
UMat    <-    mxMatrix( type="Unit", nrow=nv, ncol=1, name="U")
IMat    <-    mxMatrix( type="Iden", nrow=nv, ncol=nv, name="I")
sdAlg   <-    mxAlgebra( expression=solve(vec2diag(sqrt(V))), name="sd")
thresholdMat    <-    mxMatrix( type="Full", nrow=1, ncol=nv, free=TRUE, values=0,
                            labels=c("thrV1","thrV2"), lbound=-3, ubound=3, name="threshold")
infMat          <-    mxMatrix( type="Full", nrow=1, ncol=nv*2, free=F, values=Inf,name="infinities")
neginfMat       <-    mxMatrix( type="Full", nrow=1, ncol=nv*2, free=F, values=-Inf,name="neginfinities")
thresholdsAlg   <-    mxAlgebra( expression=rbind(neginfinities,cbind(threshold,threshold),infinities),
                                name="thresholds")
Var1Constraint  <-    mxConstraint( V==U, name="Var1")
```

DOC Model by hand

```
# Algebra for Means
MMat        <- mxMatrix( type="Zero", nrow=1, ncol=2*nv, name="M" )
#expMeanAlg  <- mxAlgebra( expression= cbind(M,M), name="expMean" )

# Algebra for expected variance/covariance matrix in MZ
expCovMZAlg <- mxAlgebra( expression= (I%x%solve(I-B)) %&% rbind  ( cbind(A+C+E , A+C),
                                    cbind(A+C    , A+C+E)), name="expCovMZ" )
# Algebra for expected variance/covariance matrix in DZ
expCovDZAlg <- mxAlgebra( expression= (I%x%solve(I-B)) %&% rbind  ( cbind(A+C+E     , 0.5%x%A+C),
                                    cbind(0.5%x%A+C , A+C+E)),  name="expCovDZ")

# Integrals & expected frequencies, MZ
AllintMZAlg     <- mxAlgebra(omxAllInt(expCovMZ, M, thresholds), name="AllintMZ")
MAllintMZAlg    <- mxAlgebra(cbind(AllintMZ[1:4,],AllintMZ[5:8,],AllintMZ[9:12,],AllintMZ[13:16,]),name="MAllintMZ")
MZExpectedFrequenciesAlg    <- mxAlgebra(vech(MAllintMZ+t(MAllintMZ)-vec2diag(diag2vec(MAllintMZ))),name="MZExpectedFrequencies")

# Integrals & expected frequencies, DZ
AllintDZAlg     <- mxAlgebra(omxAllInt(expCovDZ, M, thresholds), name="AllintDZ")
MAllintDZAlg    <- mxAlgebra(cbind(AllintDZ[1:4,],AllintDZ[5:8,],AllintDZ[9:12,],AllintDZ[13:16,]),name="MAllintDZ")
DZExpectedFrequenciesAlg    <-   mxAlgebra(vech(MAllintDZ+t(MAllintDZ)-vec2diag(diag2vec(MAllintDZ))),name="DZExpectedFrequencies")

# Data
MZObservedFrequencesMat <- mxMatrix(type="Full", nrow=10, ncol=1, free=FALSE, values=mzFreqs,
                            name="MZObservedFrequencies")
DZObservedFrequencesMat <- mxMatrix(type="Full", nrow=10, ncol=1, free=FALSE, values=dzFreqs,
                            name="DZObservedFrequencies")
```

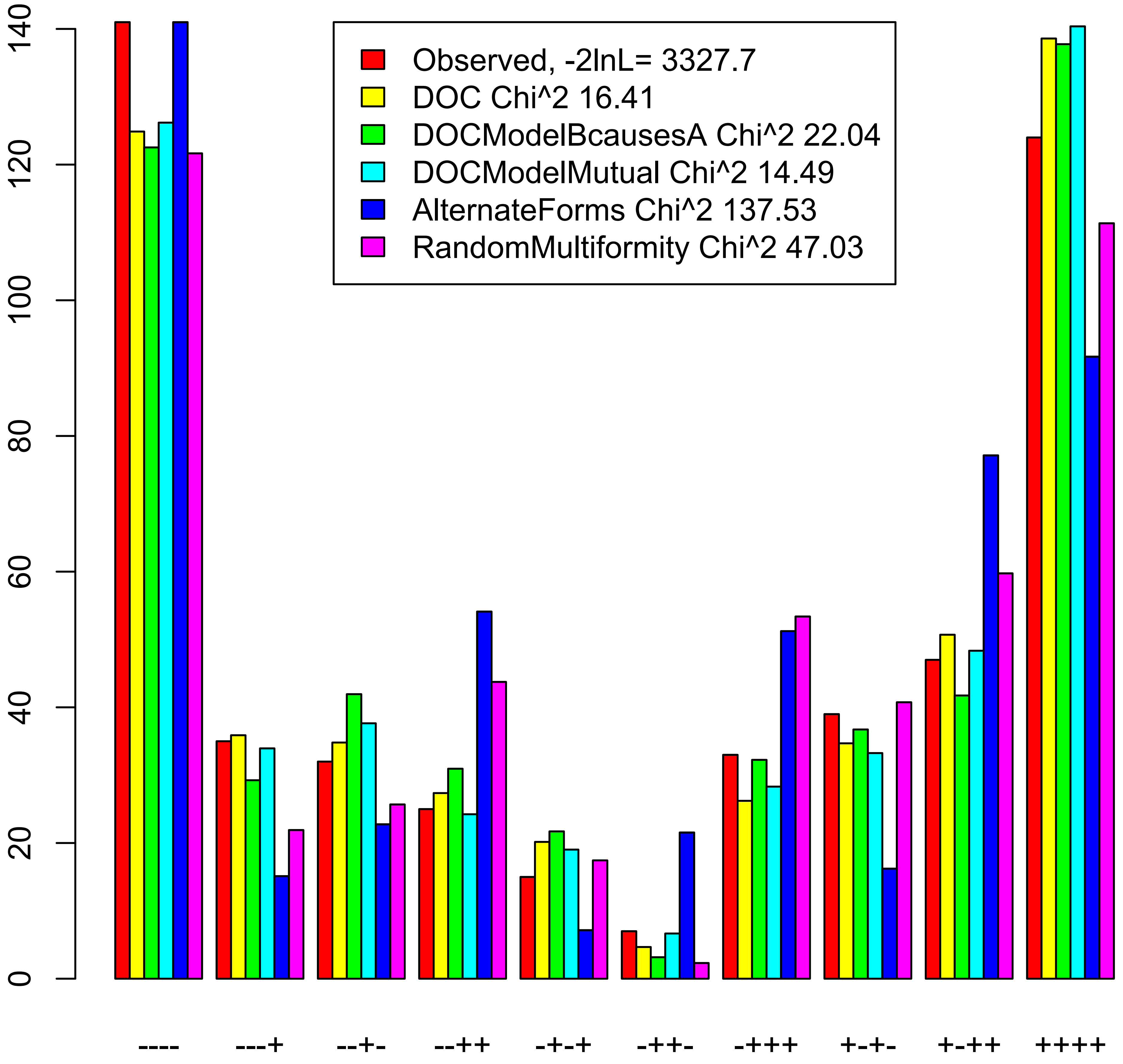
DOC Model by hand

```
# Build some lists for convenience
commonStuff <- list(aMat, cMat, eMat, BMat, MMat, UMat, IMat, thresholdMat, infMat, neginfMat,
                    AAlg, CAlg, EAlg, VAlg, sdAlg,  thresholdsAlg, Var1Constraint)
MZStuff <- list(AllintMZAlg, MAllintMZAlg, expCovMZAlg, MZExpectedFrequenciesAlg, MZObservedFrequencesMat)
DZStuff <- list(AllintDZAlg, MAllintDZAlg, expCovDZAlg, DZExpectedFrequenciesAlg, DZObservedFrequencesMat)

# MZ AlgebraObjective and Model
MZalgobjAlg <- mxAlgebra( -2 * sum(MZObservedFrequencies * log(MZExpectedFrequencies)),name="MZalgobj")
MZMod   <- mxModel("MZ", commonStuff, MZStuff, MZalgobjAlg, mxAlgebraObjective("MZalgobj"))

# DZ Model
DZalgobjAlg <- mxAlgebra( -2 * sum(DZObservedFrequencies * log(DZExpectedFrequencies)),name="DZalgobj")
DZMod   <- mxModel("DZ", commonStuff, DZStuff, DZalgobjAlg, mxAlgebraObjective("DZalgobj"))

# Combined Objective function
neg2sumllAlg    <- mxAlgebra( MZ.objective + DZ.objective, name="neg2sumll" )
neg2sumllObj    <- mxAlgebraObjective("neg2sumll")

# Combined model of the two groups
DOCModel <- mxModel("DOC", MZMod, DZMod, neg2sumllAlg, neg2sumllObj)

#
# Fit Direction of Causation Model with Cell Frequencies Input
# ---------------------------------------------------------------------

summary(DOCModelFit <- mxRun(DOCModel), numStats=20, SaturatedLikelihood=satlnL)
```

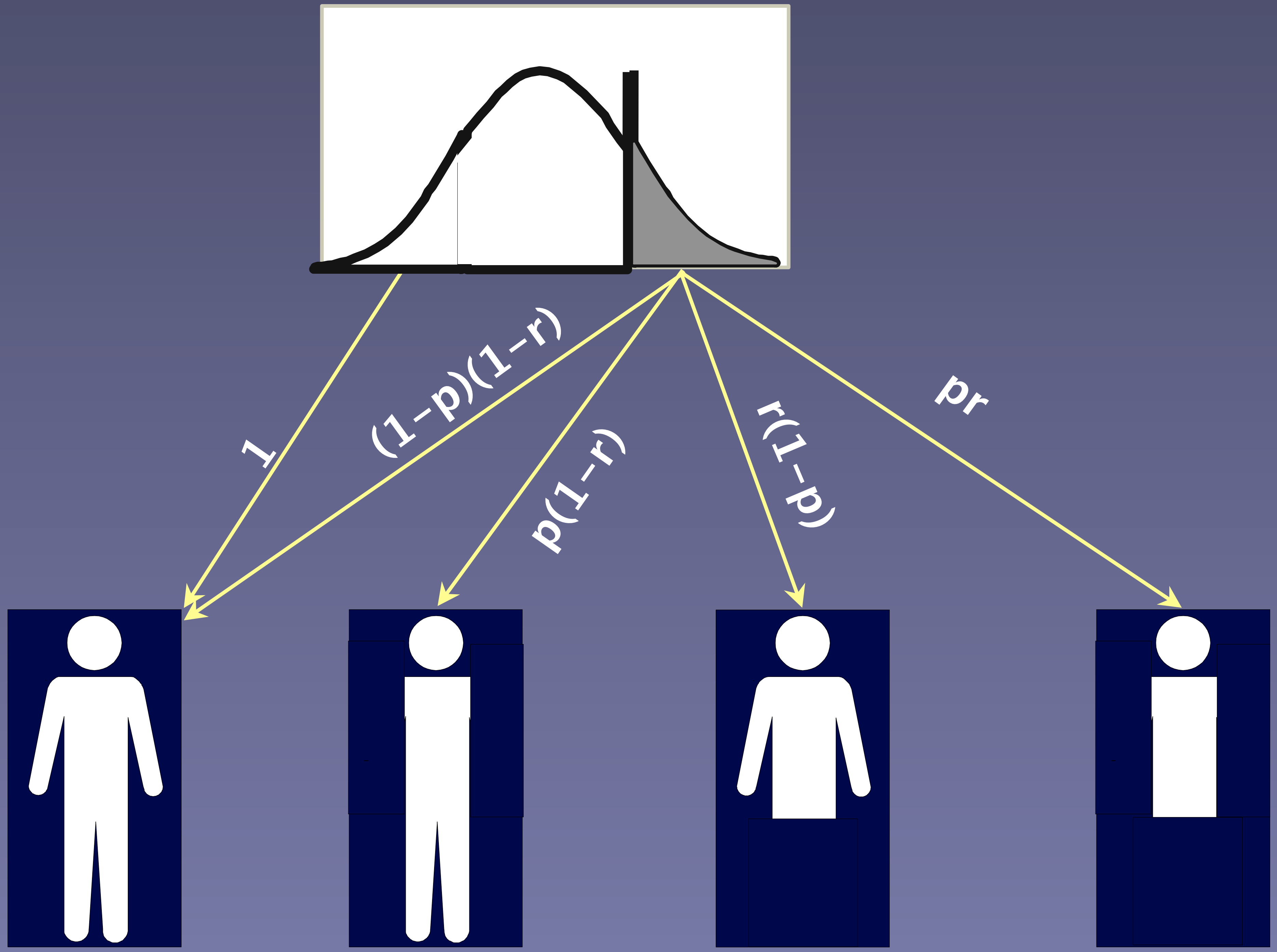
Graphing Results

```
satLogLike <- function(x) {
    proportions <- x/sum(x)
    logliks <- (log(proportions)*x)
    neg2ll <- -2*sum(logliks)
    return(list(neg2lnL=neg2ll))
}
#   models is a list of models to be graphed, MZObserved & DZObserved are data vectors, MZ is whether to print MZ or DZ
comorbidBarplot <- function (models, MZObserved, DZObserved, title="Comorbidity Model Fits", MZ=TRUE) {
#   Compute Saturated -2lnL
    satlnLmz <-  satLogLike(mzFreqs)
    satlnLdz <-  satLogLike(dzFreqs)
    satlnL <- satlnLmz$neg2lnL + satlnLdz$neg2lnL
#   Set up list of models to be plotted
    nModels <- length(models)
    nameList <- vector(mode="list", nModels+1)
    nameList[1] <- paste("Observed, -2lnL=", round(satlnL,2))
    counts <- matrix(0, nrow=nModels+1, ncol=length(MZObserved))
    colnames(counts) <- c("----","---+","--+-","--++","-+-+","-++-","-+++","+-+-","+-++","++++")
    if (MZ) {counts[1,] <- MZObserved} else {counts[1,] <- DZObserved}
    for (i in 1:nModels)
        {
        tempModel <- models[[i]]
        modelName <- tempModel@name
        nameList[i+1] <- paste(tempModel@name, "Chi^2", round(tempModel@output$minimum - satlnL, 2))
        if(MZ) {
            counts[i+1,] <- tempModel$MZ.MZExpectedFrequencies@result * sum(MZObserved)}
        else {
            counts[i+1,] <- tempModel$DZ.DZExpectedFrequencies@result * sum(DZObserved)}
        }
    barplot(counts,
        main=title,
        xlab="Pair Type",
        col=rainbow(nModels+1, s = 1, v = 1, start = 0, end = max(1,nModels)/(nModels+1), alpha = 1),
        legend = nameList, beside=TRUE,
        args.legend = list(x = "top"))
}

comorbidBarplot(list(DOCModelFit,DOCModelBcausesARun, DOCModelMutualRun, AltFormsDiffThreshRun, RandomMultiformityRun), mzFreqs, dzFreqs)
```

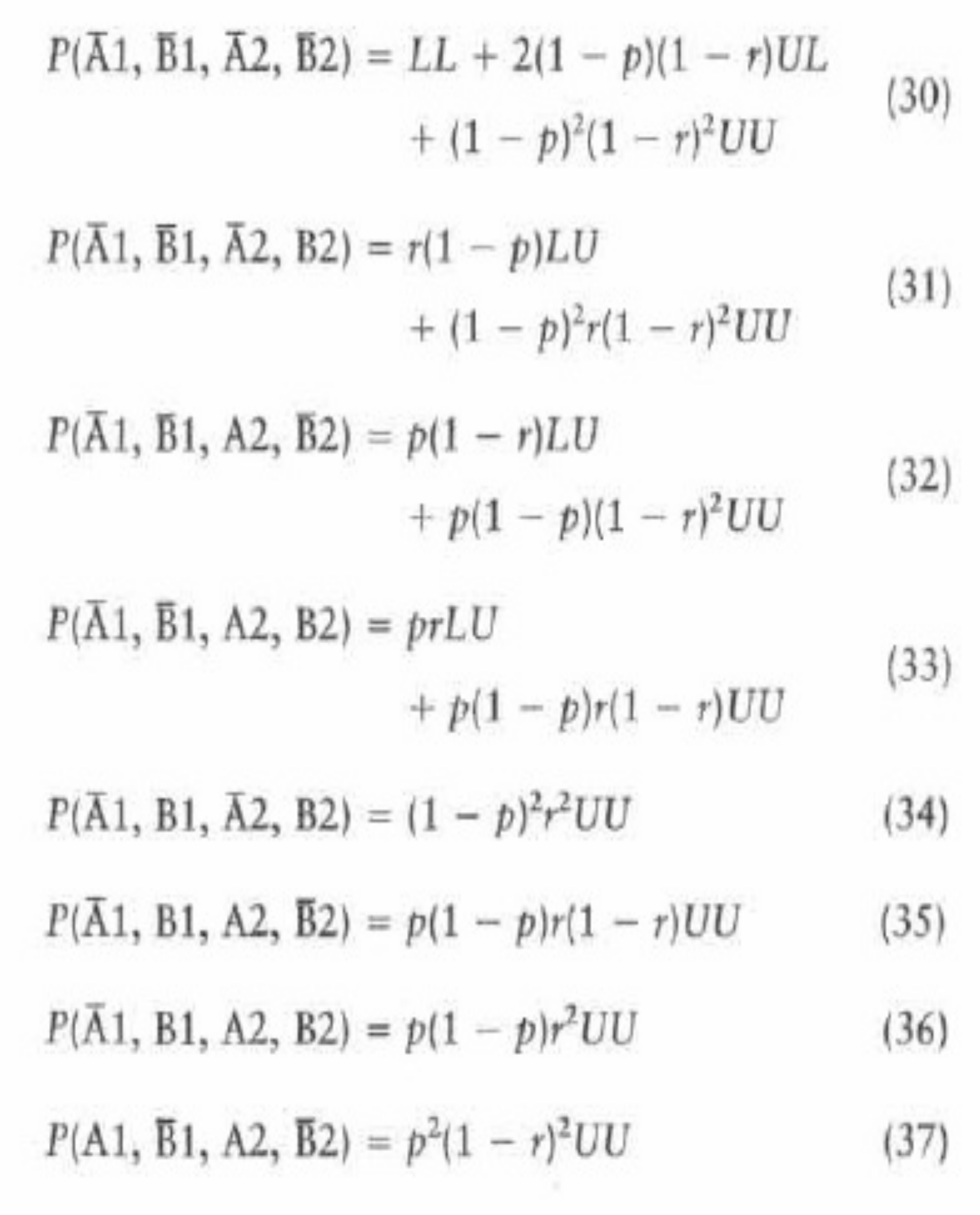
Comorbidity Model Fits



Alternate forms: One underlying continuum



Alternate forms: Detail of pairs



$$P(\bar{A}1, \bar{B}1, \bar{A}2, \bar{B}2) = LL + 2(1-p)(1-r)UL + (1-p)^2(1-r)^2UU \quad (30)$$

$$P(\bar{A}1, \bar{B}1, \bar{A}2, B2) = r(1-p)LU + (1-p)^2r(1-r)^2UU \quad (31)$$

$$P(\bar{A}1, \bar{B}1, A2, \bar{B}2) = p(1-r)LU + p(1-p)(1-r)^2UU \quad (32)$$

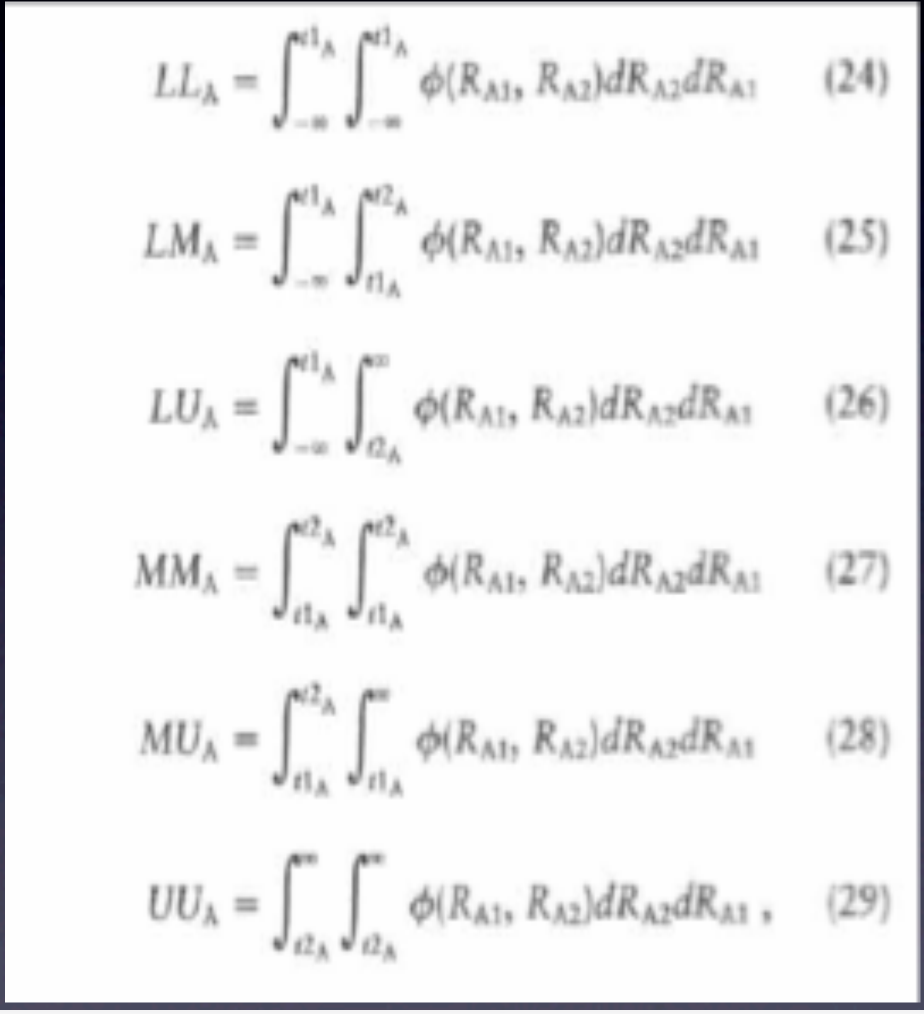
$$P(\bar{A}1, \bar{B}1, A2, B2) = prLU + p(1-p)r(1-r)UU \quad (33)$$

$$P(\bar{A}1, B1, \bar{A}2, B2) = (1-p)^2r^2UU \quad (34)$$

$$P(\bar{A}1, B1, A2, \bar{B}2) = p(1-p)r(1-r)UU \quad (35)$$

$$P(\bar{A}1, B1, A2, B2) = p(1-p)r^2UU \quad (36)$$

$$P(A1, \bar{B}1, A2, \bar{B}2) = p^2(1-r)^2UU \quad (37)$$



$$LL_A = \int_{-\infty}^{t1_A}\int_{-\infty}^{t1_A} \phi(R_{A1}, R_{A2})dR_{A2}dR_{A1} \quad (24)$$

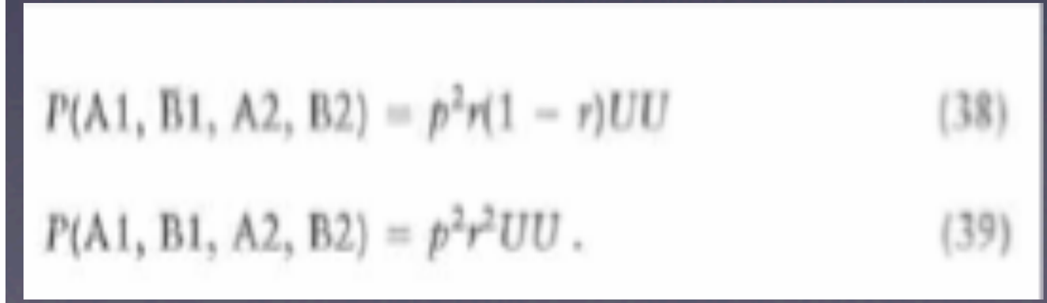
$$LM_A = \int_{-\infty}^{t1_A}\int_{t1_A}^{t2_A} \phi(R_{A1}, R_{A2})dR_{A2}dR_{A1} \quad (25)$$

$$LU_A = \int_{-\infty}^{t1_A}\int_{t2_A}^{\infty} \phi(R_{A1}, R_{A2})dR_{A2}dR_{A1} \quad (26)$$

$$MM_A = \int_{t1_A}^{t2_A}\int_{t1_A}^{t2_A} \phi(R_{A1}, R_{A2})dR_{A2}dR_{A1} \quad (27)$$

$$MU_A = \int_{t1_A}^{t2_A}\int_{t1_A}^{\infty} \phi(R_{A1}, R_{A2})dR_{A2}dR_{A1} \quad (28)$$

$$UU_A = \int_{t2_A}^{\infty}\int_{t2_A}^{\infty} \phi(R_{A1}, R_{A2})dR_{A2}dR_{A1}\,, \quad (29)$$



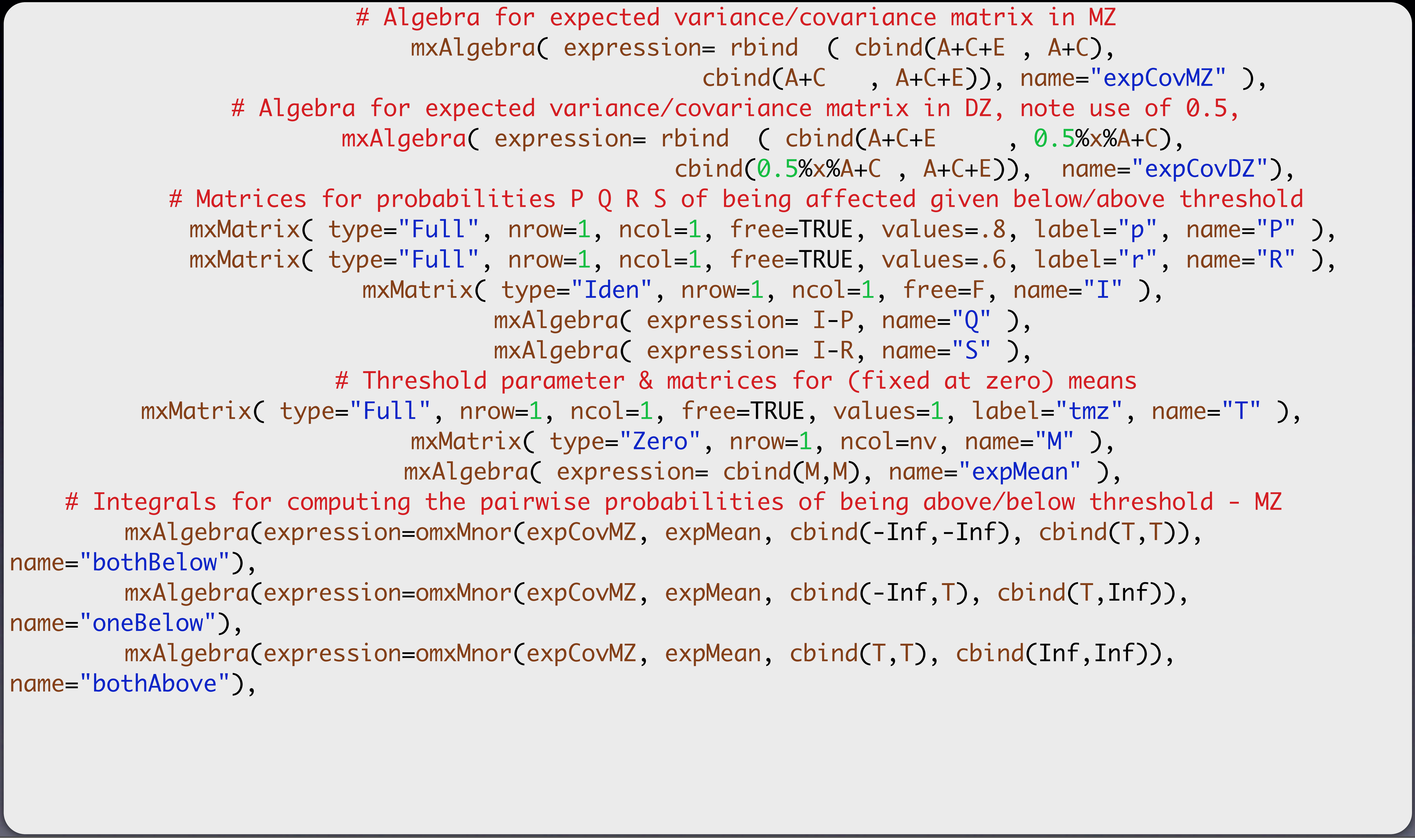
$$P(A1, \bar{B}1, A2, B2) = p^2r(1-r)UU \quad (38)$$

$$P(A1, B1, A2, B2) = p^2r^2UU\,. \quad (39)$$

OpenMx Script Algebra for Alternate Forms

```
# Program: Alternate Forms

require(OpenMx)
nv<-1
# Fit Alternate Forms Model with Cell Frequencies Input, ACE.one overall Threshold
# ----------------------------------------------------------------------
AltFormsModel <- mxModel("AlternateForms",
    mxModel("ACE",
    # Matrices a, c, and e to store a, c, and e path coefficients
        mxMatrix( type="Full", nrow=nv, ncol=nv, free=TRUE, values=.6, label="a11", name="a" ),
        mxMatrix( type="Full", nrow=nv, ncol=nv, free=TRUE, values=.6, label="c11", name="c" ),
        mxMatrix( type="Full", nrow=nv, ncol=nv, free=TRUE, values=sqrt(.28), label="e11",
name="e" ),
    # Matrices A, C, and E compute variance components
        mxAlgebra( expression=a %*% t(a), name="A" ),
        mxAlgebra( expression=c %*% t(c), name="C" ),
        mxAlgebra( expression=e %*% t(e), name="E" ),
    # Algebra to compute total variances and standard deviations (diagonal only)
        mxAlgebra( expression=A+C+E, name="V" ),
        mxMatrix( type="Iden", nrow=nv, ncol=nv, name="I"),
        mxAlgebra( expression=solve(sqrt(I*V)), name="sd"),
    # Constraint on variance of A+C+E latent variables
        mxConstraint( alg1="V", "=", alg2="I", name="Var1"),
```

OpenMx Script algebra for



```
            # Algebra for expected variance/covariance matrix in MZ
                mxAlgebra( expression= rbind  ( cbind(A+C+E , A+C),
                                                cbind(A+C   , A+C+E)), name="expCovMZ" ),
       # Algebra for expected variance/covariance matrix in DZ, note use of 0.5,
                mxAlgebra( expression= rbind  ( cbind(A+C+E     , 0.5%x%A+C),
                                                cbind(0.5%x%A+C , A+C+E)),  name="expCovDZ"),
     # Matrices for probabilities P Q R S of being affected given below/above threshold
      mxMatrix( type="Full", nrow=1, ncol=1, free=TRUE, values=.8, label="p", name="P" ),
      mxMatrix( type="Full", nrow=1, ncol=1, free=TRUE, values=.6, label="r", name="R" ),
            mxMatrix( type="Iden", nrow=1, ncol=1, free=F, name="I" ),
                        mxAlgebra( expression= I-P, name="Q" ),
                        mxAlgebra( expression= I-R, name="S" ),
            # Threshold parameter & matrices for (fixed at zero) means
     mxMatrix( type="Full", nrow=1, ncol=1, free=TRUE, values=1, label="tmz", name="T" ),
                  mxMatrix( type="Zero", nrow=1, ncol=nv, name="M" ),
                  mxAlgebra( expression= cbind(M,M), name="expMean" ),
  # Integrals for computing the pairwise probabilities of being above/below threshold - MZ
    mxAlgebra(expression=omxMnor(expCovMZ, expMean, cbind(-Inf,-Inf), cbind(T,T)),
name="bothBelow"),
    mxAlgebra(expression=omxMnor(expCovMZ, expMean, cbind(-Inf,T), cbind(T,Inf)),
name="oneBelow"),
    mxAlgebra(expression=omxMnor(expCovMZ, expMean, cbind(T,T), cbind(Inf,Inf)),
name="bothAbove"),
```

OpenMx Script algebra for

```
     # Integrals for computing the pairwise probabilities of being above/below threshold - DZ
mxAlgebra(expression=omxMnor(expCovDZ, expMean, cbind(-Inf,-Inf), cbind(T,T)), name="bothBelowDZ"),
 mxAlgebra(expression=omxMnor(expCovDZ, expMean, cbind(-Inf,T), cbind(T,Inf)), name="oneBelowDZ"),
 mxAlgebra(expression=omxMnor(expCovDZ, expMean, cbind(T,T), cbind(Inf,Inf)), name="bothAboveDZ"),

  # Finally, predicted proportions in each of 10 cells for MZ
      mxAlgebra(rbind(
      bothBelow + 2*oneBelow*Q*S + bothAbove*Q*Q*S*S,
      2*(oneBelow*R*Q + bothAbove*Q*Q*R*S),
      2*(oneBelow*P*S + bothAbove*P*Q*S*S),
      2*(oneBelow*P*R + bothAbove*P*R*Q*S),
      bothAbove*Q*Q*R*R,
      2*bothAbove*P*Q*R*S,
      2*bothAbove*P*Q*R*R,
      bothAbove*P*S*P*S,
      2*bothAbove*P*S*P*R,
      bothAbove*P*R*P*R
      ),name="MZExpectedFrequencies"),
```

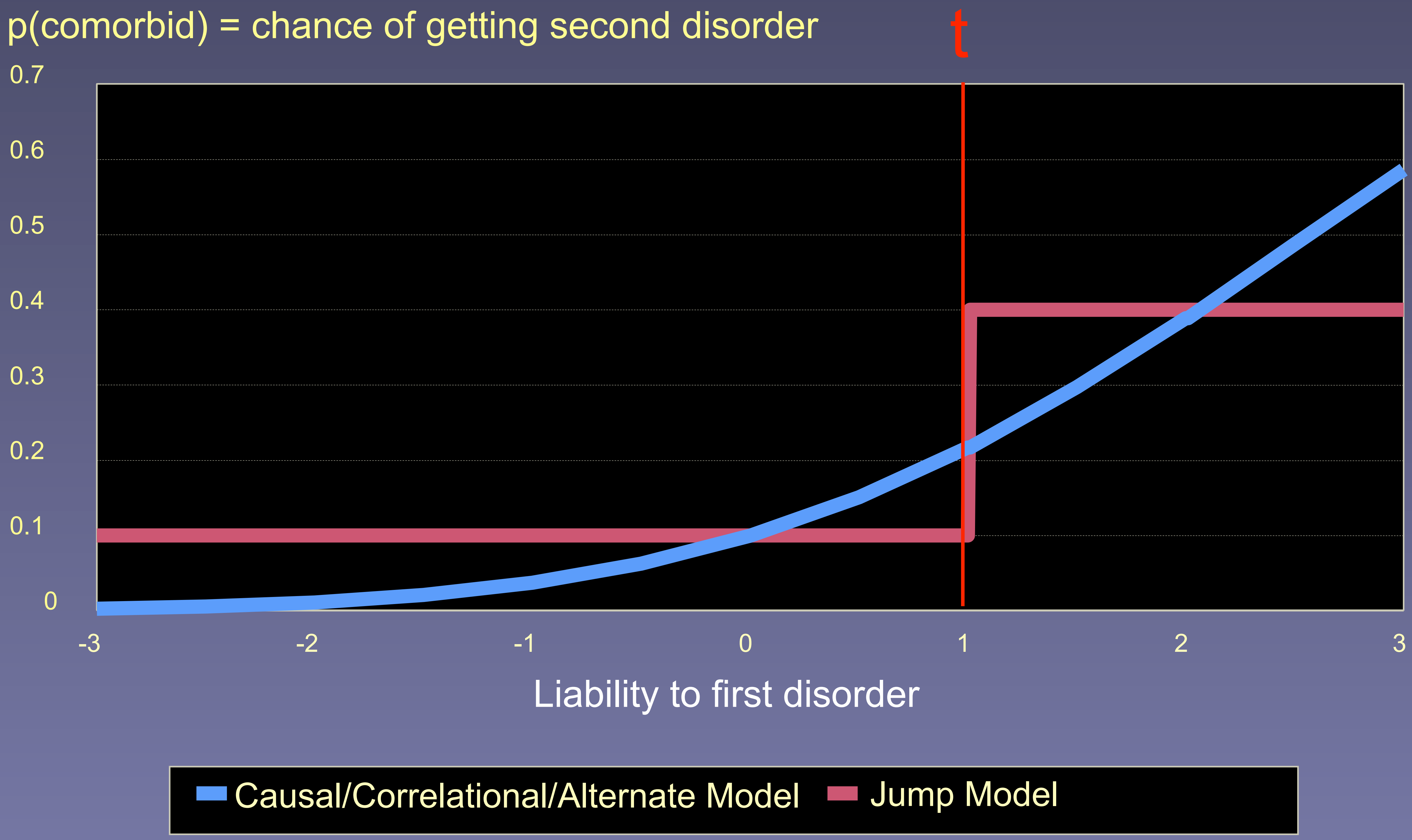
OpenMx Script algebra for

```
# Finally, predicted proportions in each of 10 cells for DZ
mxAlgebra(rbind(
bothBelowDZ + 2*oneBelowDZ*Q*S + bothAboveDZ*Q*Q*S*S,
2*(oneBelowDZ*R*Q + bothAboveDZ*Q*Q*R*S),
2*(oneBelowDZ*P*S + bothAboveDZ*P*Q*S*S),
2*(oneBelowDZ*P*R + bothAboveDZ*P*R*Q*S),
bothAboveDZ*Q*Q*R*R,
2*bothAboveDZ*P*Q*R*S,
2*bothAboveDZ*P*Q*R*R,
bothAboveDZ*P*S*P*S,
2*bothAboveDZ*P*S*P*R,
bothAboveDZ*P*R*P*R),name="DZExpectedFrequencies")),

mxModel("MZ",
mxMatrix(type="Full", nrow=10, ncol=1, free=FALSE,
values=c(141,35,32,25,15,7,33,18,39,47), name="MZObservedFrequencies"),
mxAlgebra( -2 * sum(MZObservedFrequencies * log
(ACE.MZExpectedFrequencies)),name="MZalgobj"),
mxAlgebraObjective("MZalgobj")),
```

OpenMx Script algebra for

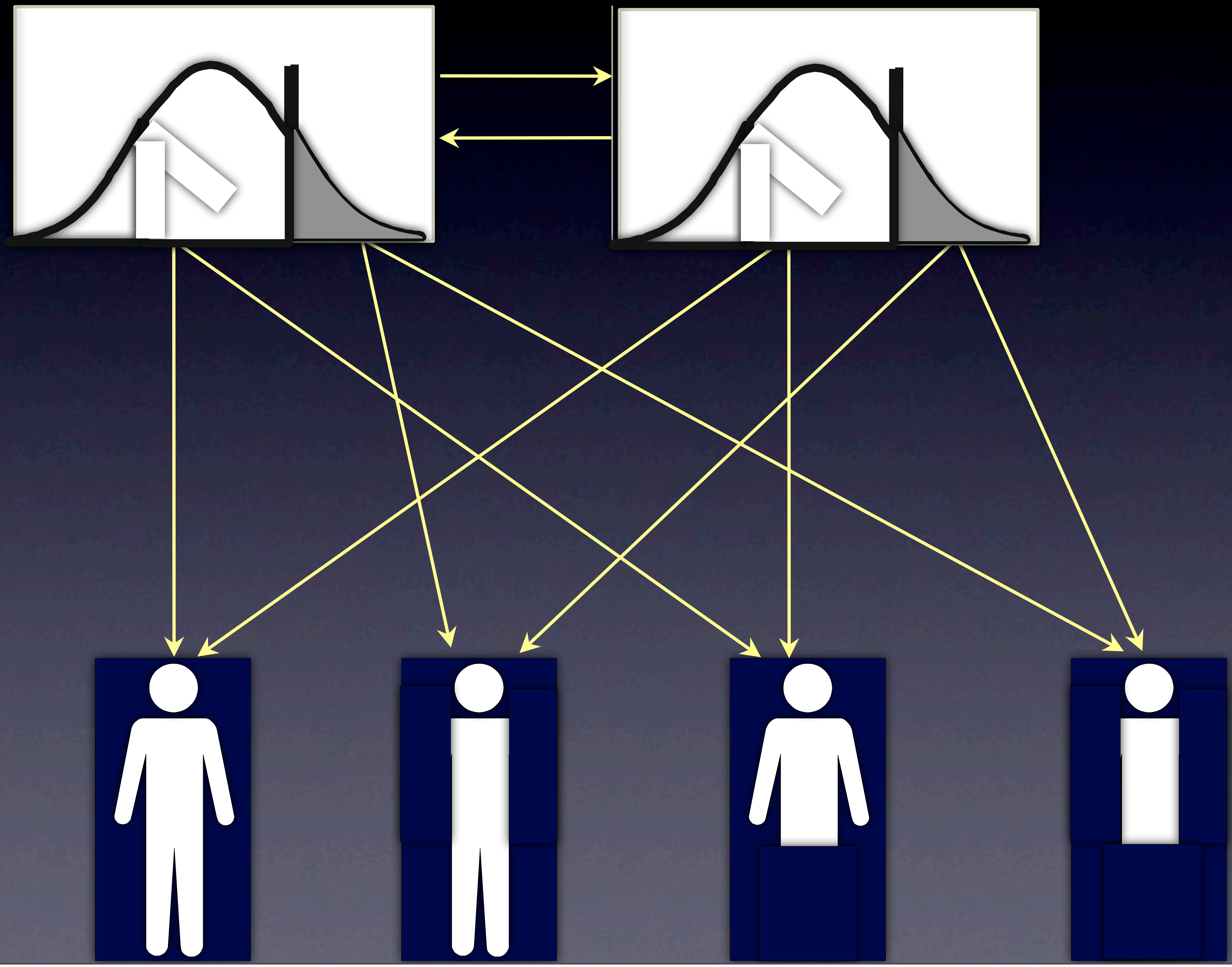
```
    mxModel("DZ",
        mxMatrix(type="Full", nrow=10, ncol=1, free=F, values=c(58,18,27,44,7,6,33,15,38,81),
            name="DZObservedFrequencies"),
        mxAlgebra(
                  -2 * sum(DZObservedFrequencies *
                            log (ACE.DZExpectedFrequencies)),name="DZalgobj"),
mxAlgebraObjective("DZalgobj"),
    mxAlgebra( MZ.objective + DZ.objective, name="-2sumll" ),
    mxAlgebraObjective("-2sumll")))

    AltFormsRun<-mxRun(AltFormsModel)
    summary(AltFormsRun)
```

Alternative models of increasing risk to a second disorder

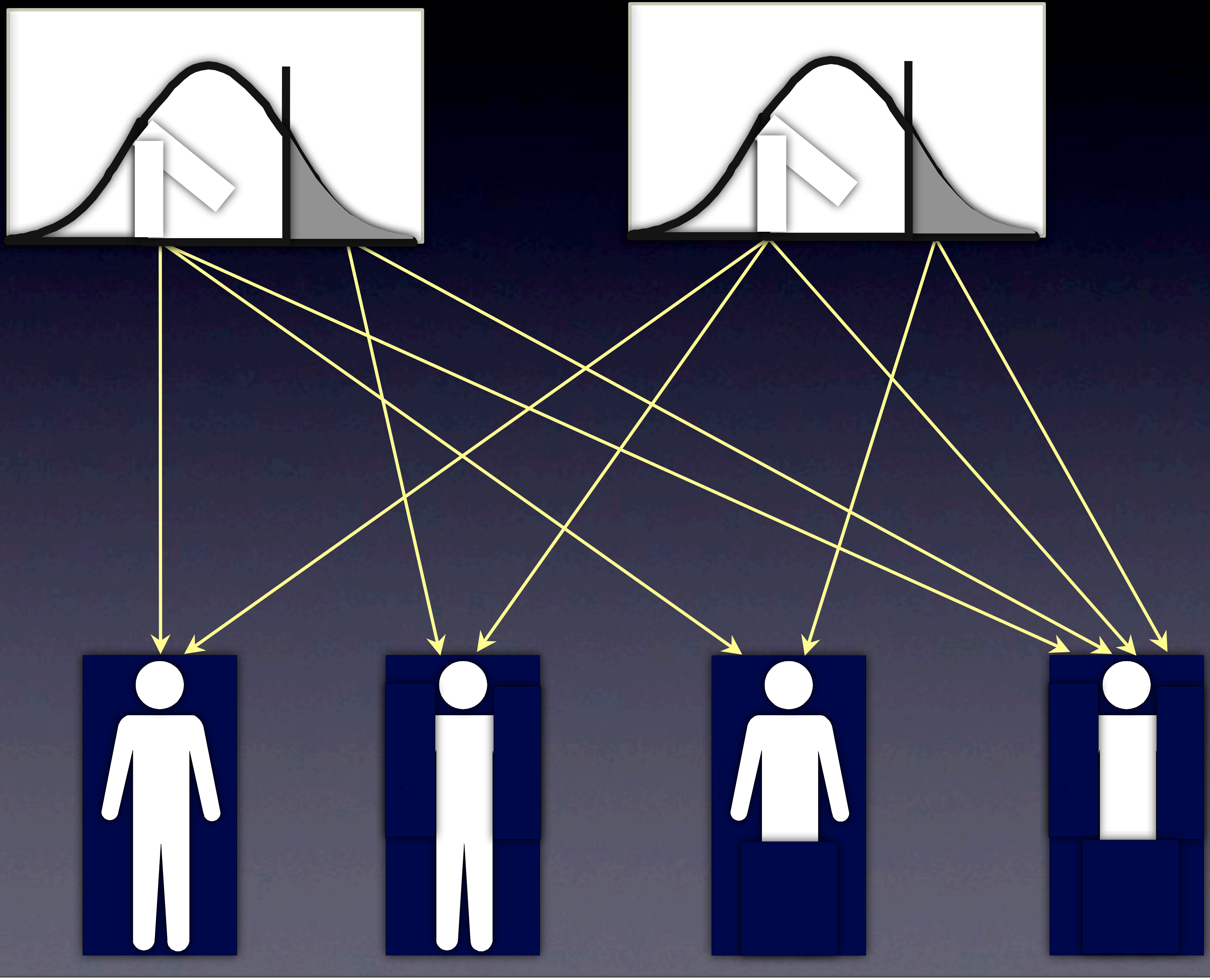


Threshold Model r=.5

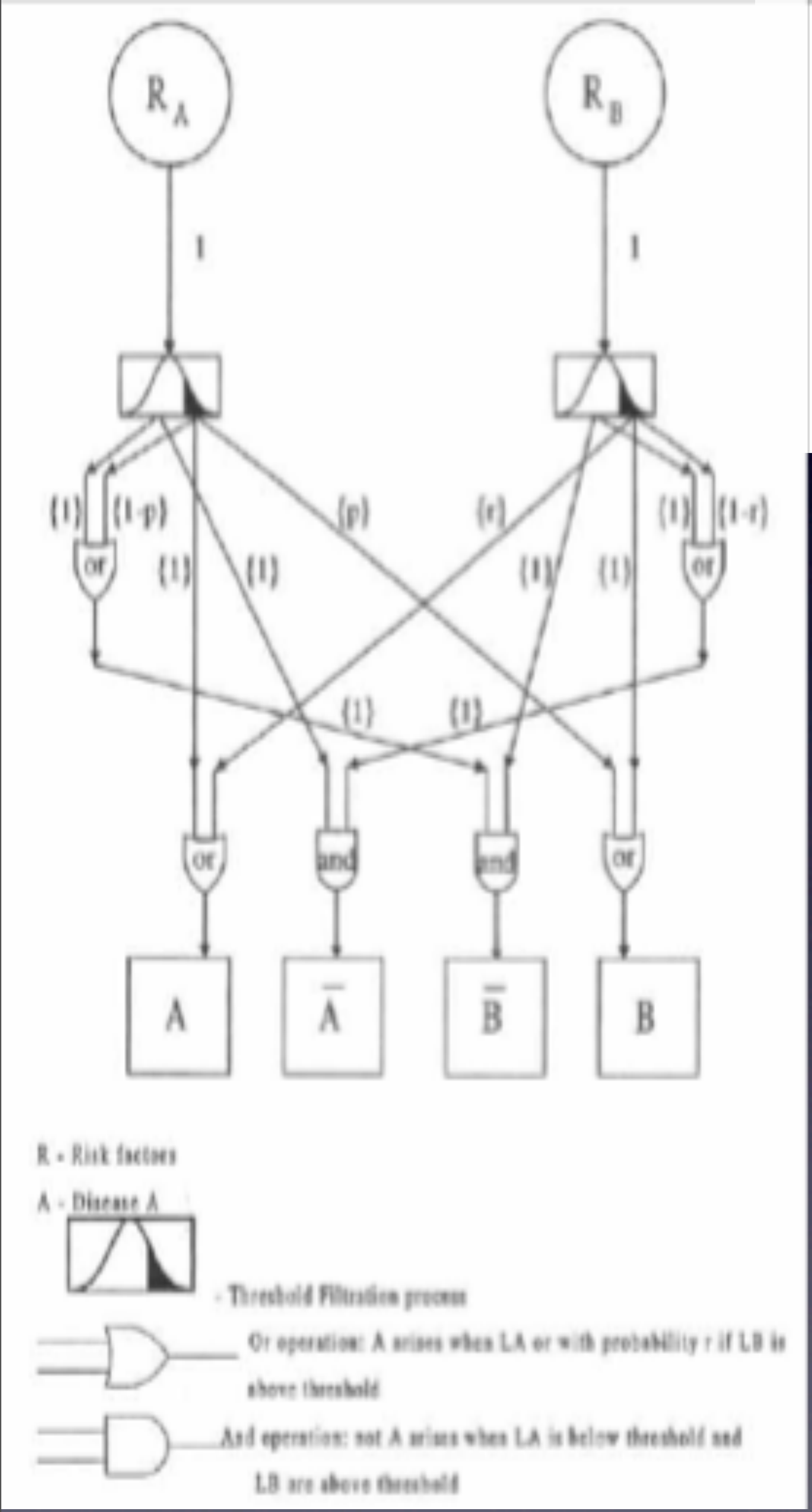
Causal or correlated models



Jump Model: Actually having one disorder raises chance of getting seconc



Random Multiformity: Detail



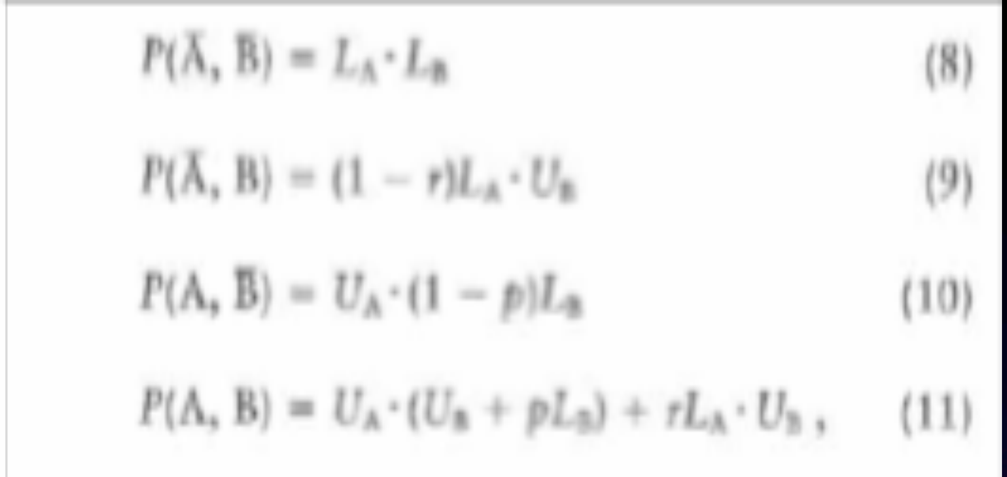
R = Risk factors

A = Disease A

- Threshold Filtration process

Or operation: A arises when LA or with probability r if LB is above threshold

And operation: not A arises when LA is below threshold and LB are above threshold

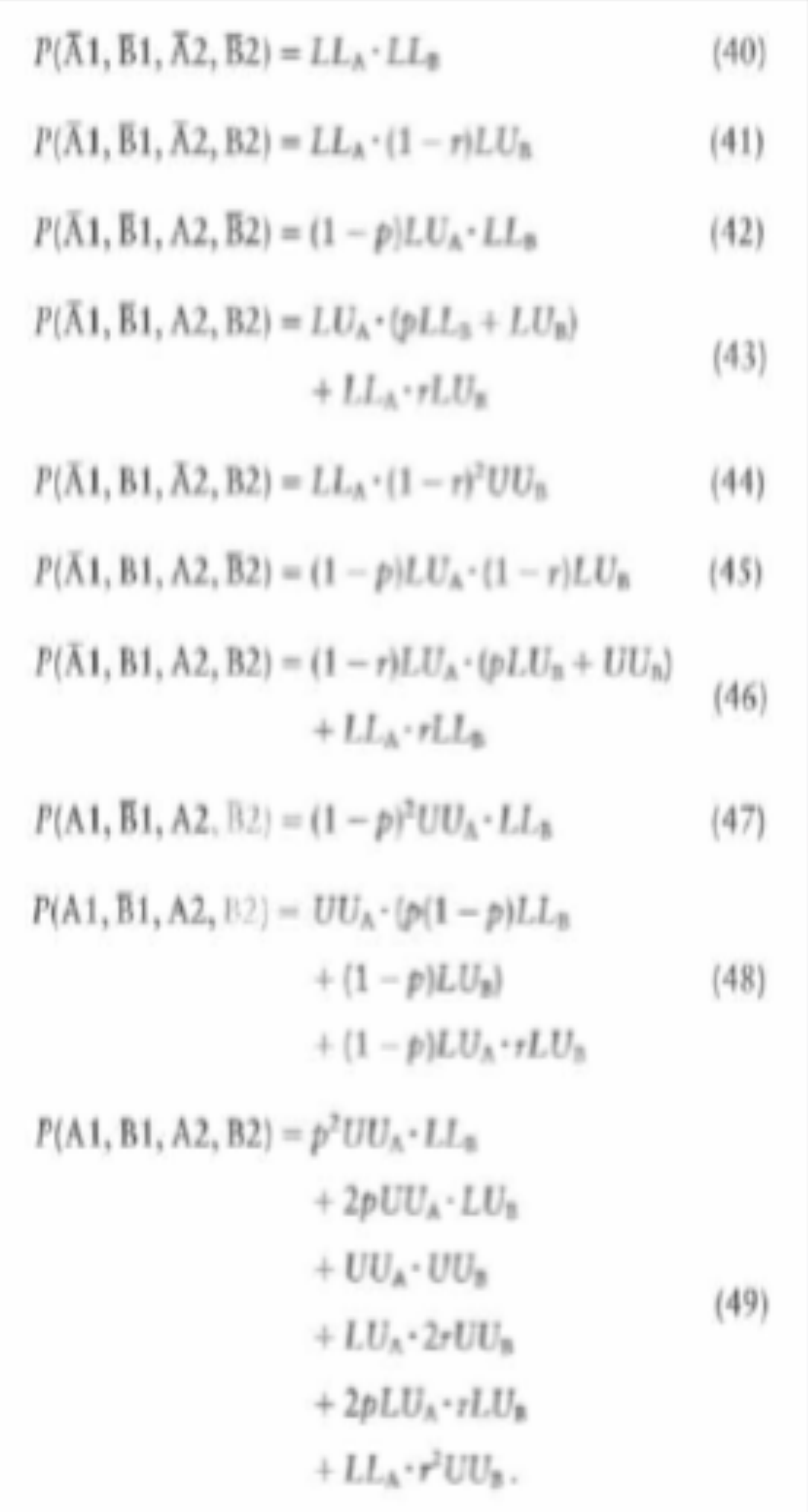


$$P(\bar{A}, \bar{B}) = L_A \cdot L_B \tag{8}$$

$$P(\bar{A}, B) = (1 - r)L_A \cdot U_B \tag{9}$$

$$P(A, \bar{B}) = U_A \cdot (1 - p)L_B \tag{10}$$

$$P(A, B) = U_A \cdot (U_B + pL_B) + rL_A \cdot U_B\,, \tag{11}$$



$$P(\bar{A}1, \bar{B}1, \bar{A}2, \bar{B}2) = LL_A \cdot LL_B \tag{40}$$

$$P(\bar{A}1, \bar{B}1, \bar{A}2, B2) = LL_A \cdot (1 - r)LU_B \tag{41}$$

$$P(\bar{A}1, \bar{B}1, A2, \bar{B}2) = (1 - p)LU_A \cdot LL_B \tag{42}$$

$$\begin{aligned} P(\bar{A}1, \bar{B}1, A2, B2) = {} & LU_A \cdot (pLL_B + LU_B) \\ & + LL_A \cdot rLU_B \end{aligned} \tag{43}$$

$$P(\bar{A}1, B1, \bar{A}2, B2) = LL_A \cdot (1 - r)^2 UU_B \tag{44}$$

$$P(\bar{A}1, B1, A2, \bar{B}2) = (1 - p)LU_A \cdot (1 - r)LU_B \tag{45}$$

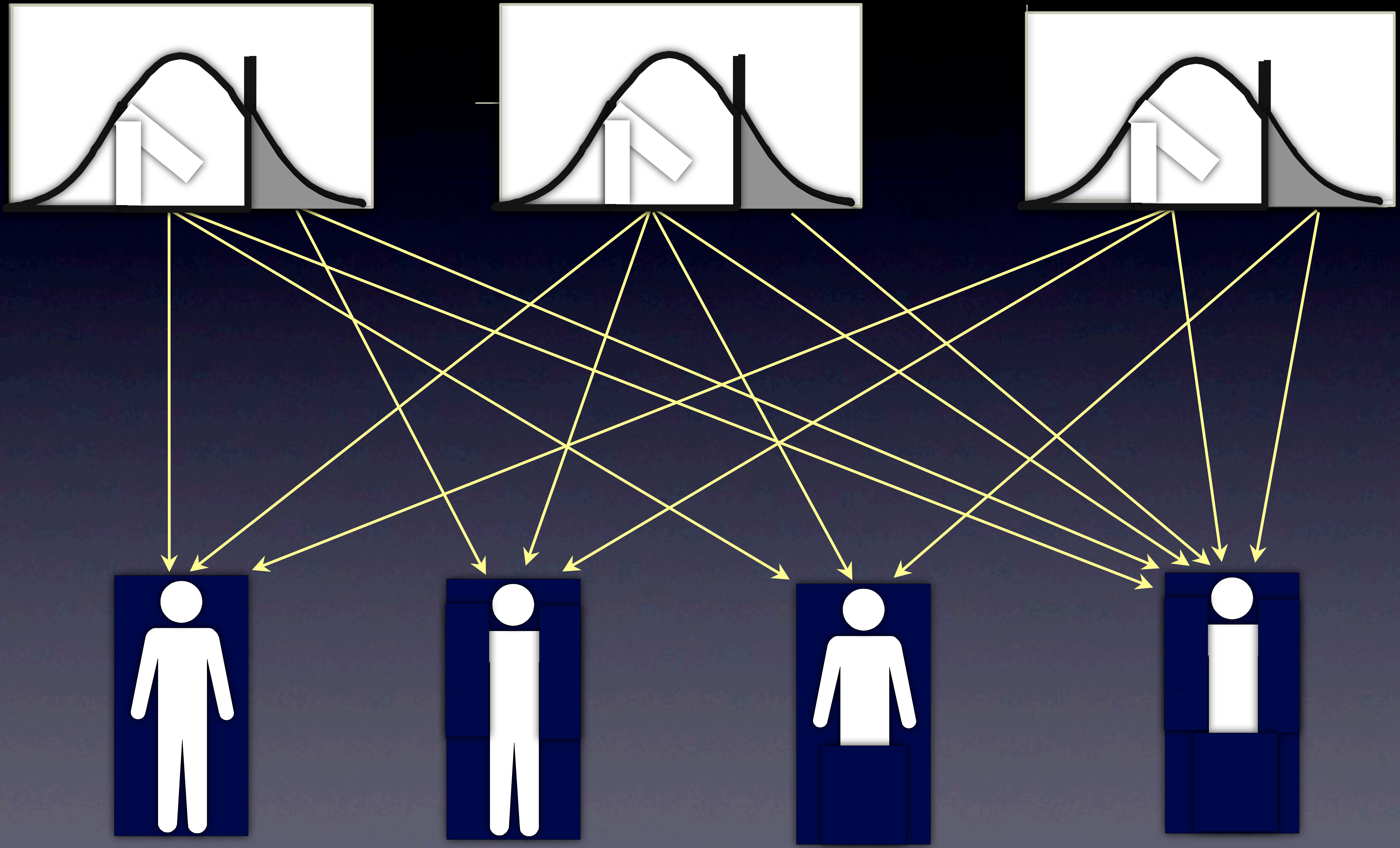
$$\begin{aligned} P(\bar{A}1, B1, A2, B2) = {} & (1 - r)LU_A \cdot (pLU_B + UU_B) \\ & + LL_A \cdot rLL_B \end{aligned} \tag{46}$$

$$P(A1, \bar{B}1, A2, B2) = (1 - p)^2 UU_A \cdot LL_B \tag{47}$$

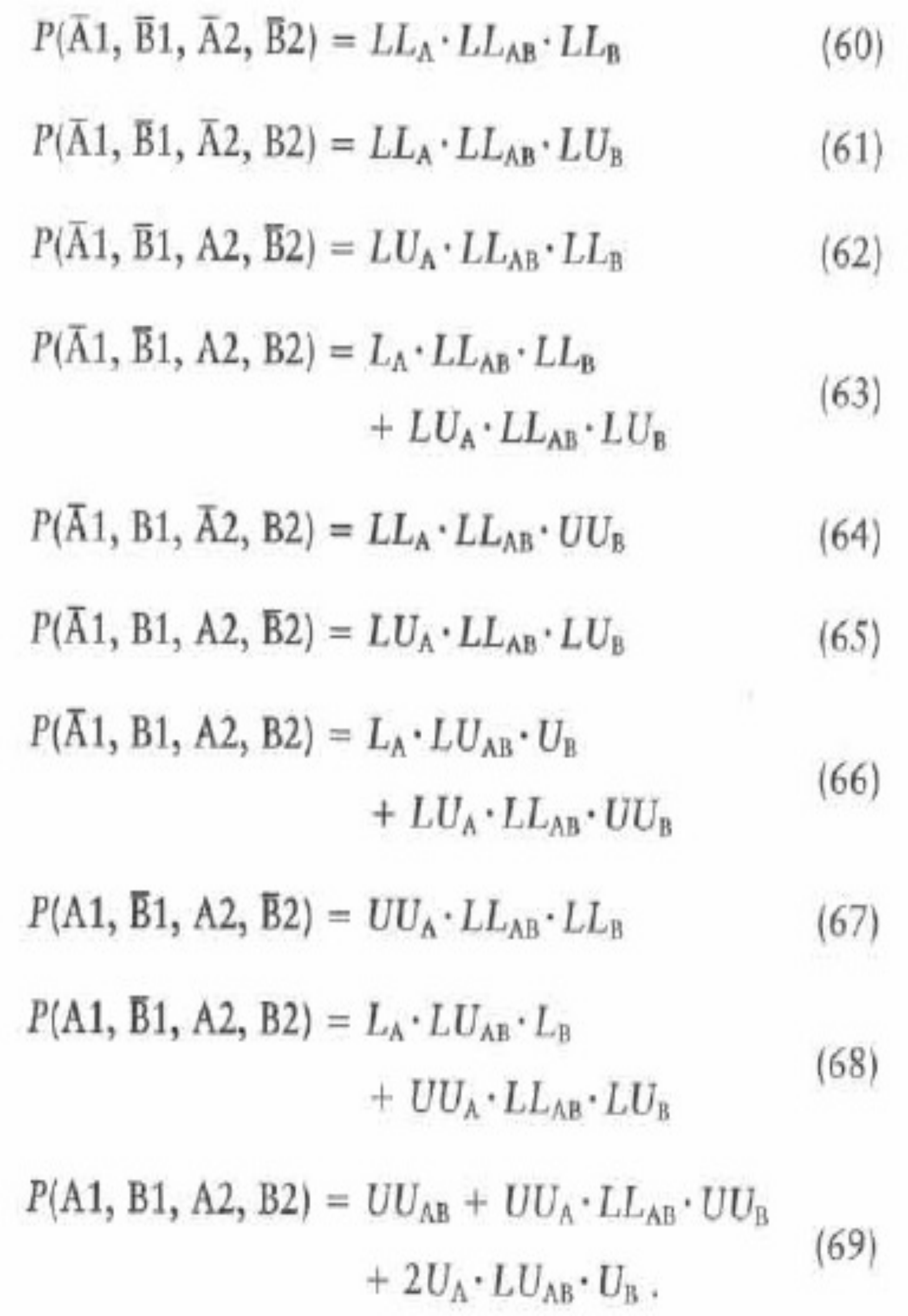
$$\begin{aligned} P(A1, \bar{B}1, A2, B2) = {} & UU_A \cdot (p(1 - p)LL_B \\ & + (1 - p)LU_B) \\ & + (1 - p)LU_A \cdot rLU_B \end{aligned} \tag{48}$$

$$\begin{aligned} P(A1, B1, A2, B2) = {} & p^2 UU_A \cdot LL_B \\ & + 2pUU_A \cdot LU_B \\ & + UU_A \cdot UU_B \\ & + LU_A \cdot 2rUU_B \\ & + 2pLU_A \cdot rLU_B \\ & + LL_A \cdot r^2 UU_B\,. \end{aligned} \tag{49}$$

Three separate disorders



Three Independent Disorders



$$P(\bar{A}1, \bar{B}1, \bar{A}2, \bar{B}2) = LL_A \cdot LL_{AB} \cdot LL_B \quad (60)$$

$$P(\bar{A}1, \bar{B}1, \bar{A}2, B2) = LL_A \cdot LL_{AB} \cdot LU_B \quad (61)$$

$$P(\bar{A}1, \bar{B}1, A2, \bar{B}2) = LU_A \cdot LL_{AB} \cdot LL_B \quad (62)$$

$$P(\bar{A}1, \bar{B}1, A2, B2) = L_A \cdot LL_{AB} \cdot LL_B + LU_A \cdot LL_{AB} \cdot LU_B \quad (63)$$

$$P(\bar{A}1, B1, \bar{A}2, B2) = LL_A \cdot LL_{AB} \cdot UU_B \quad (64)$$

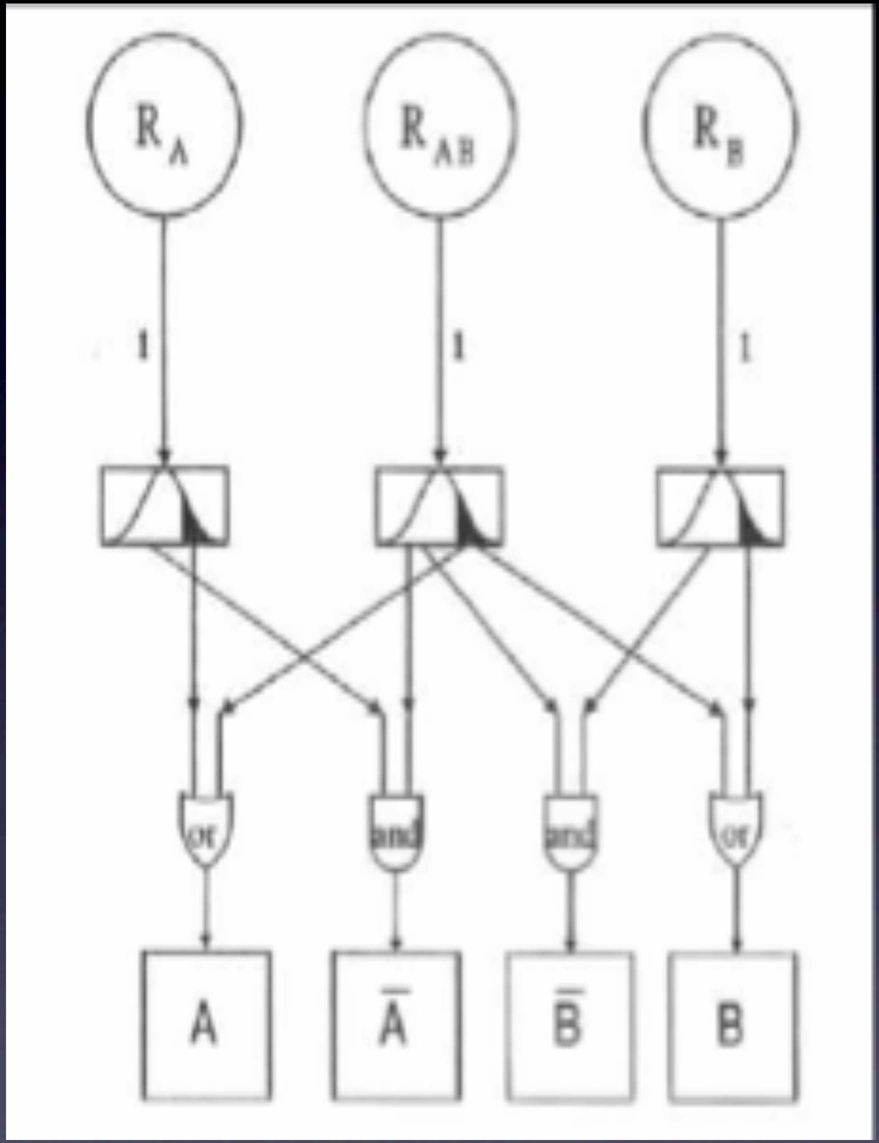
$$P(\bar{A}1, B1, A2, \bar{B}2) = LU_A \cdot LL_{AB} \cdot LU_B \quad (65)$$

$$P(\bar{A}1, B1, A2, B2) = L_A \cdot LU_{AB} \cdot U_B + LU_A \cdot LL_{AB} \cdot UU_B \quad (66)$$

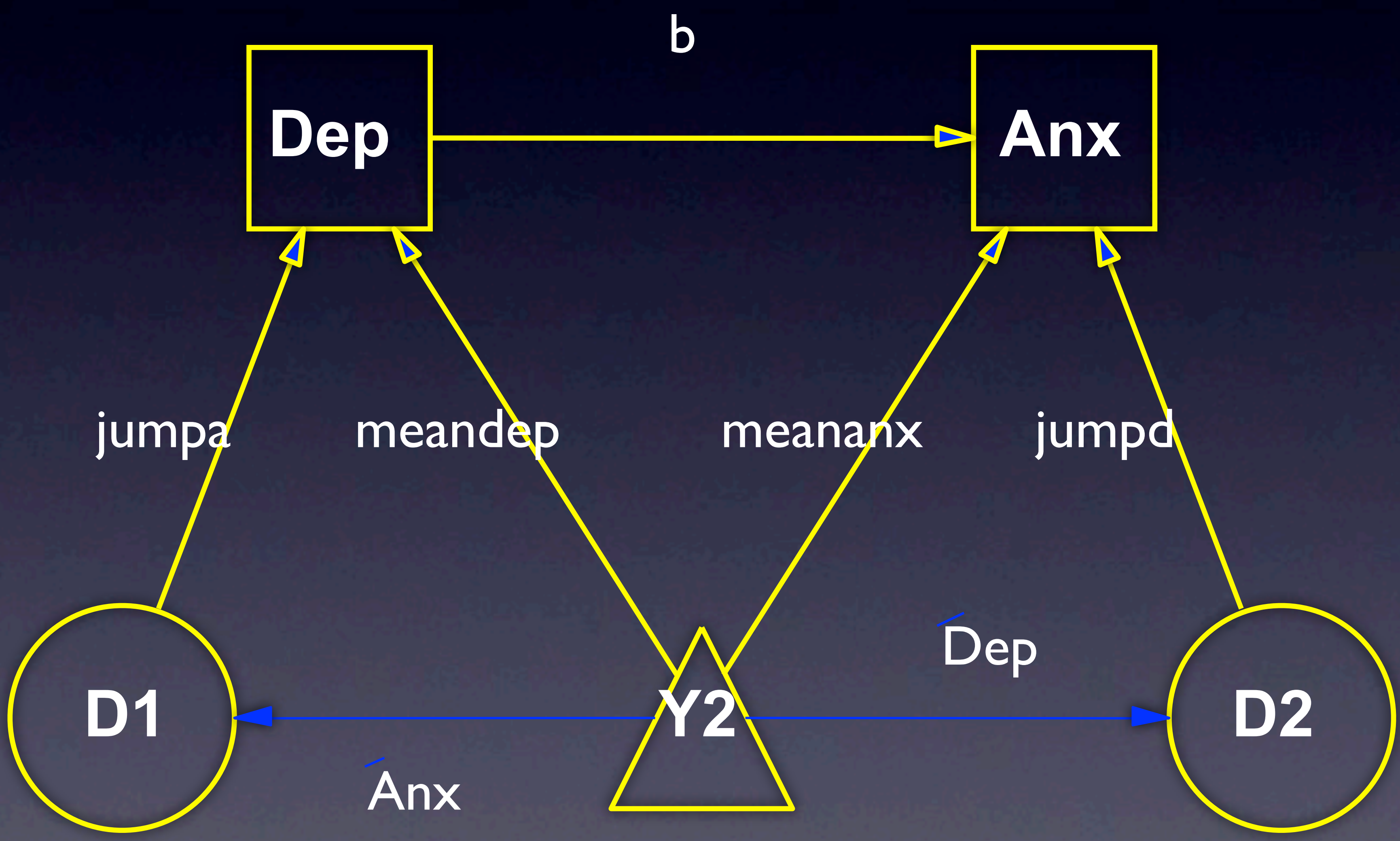
$$P(A1, \bar{B}1, A2, \bar{B}2) = UU_A \cdot LL_{AB} \cdot LL_B \quad (67)$$

$$P(A1, \bar{B}1, A2, B2) = L_A \cdot LU_{AB} \cdot L_B + UU_A \cdot LL_{AB} \cdot LU_B \quad (68)$$

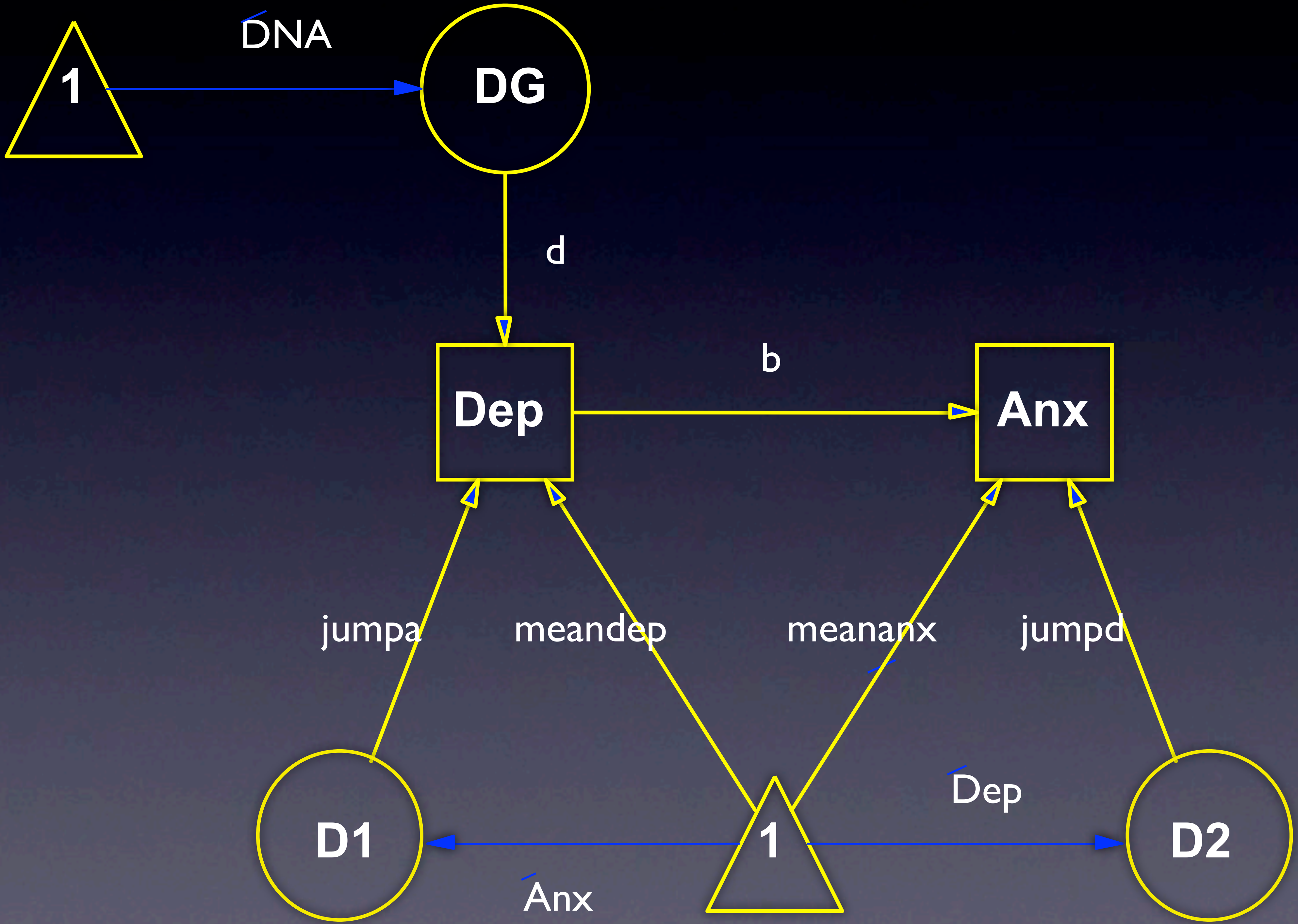
$$P(A1, B1, A2, B2) = UU_{AB} + UU_A \cdot LL_{AB} \cdot UU_B + 2U_A \cdot LU_{AB} \cdot U_B\,. \quad (69)$$



Unified Comorbidity Model?



Unified Genetic Comorbidity Model?



Sources for comorbidity scripts

- http://ibgwww.colorado.edu/cadd/software
- Soo Rhee's website! Excellent!
- Includes covariates e.g., age (Rhee et al submitted)
- Clinical selected samples as well
- Exercise: download and fit the examples and decide on best fit model

- http://www.vcu.edu/mx/examples
- Mike Neale's script website.
- More than a little bit dusty

OpenMx User-defined Functions

- Can specify AlgebraObjective

```
mxAlgebra( MZ.objective + DZ.objective, name="-2sumll" ),
    mxAlgebraObjective("-2sumll"))
```

- Any mxAlgebra you like!
  - Woohoo!
- See, e.g., http://openmx.psyc.virginia.edu/repoview/1/trunk/models/passing/oneLocusLikelihood.R
- One & two locus ABO blood group examples

Comorbidity with Covariates

- Soo Rhee's website again
- http://ibgwww.colorado.edu/cadd/software
- These scripts are in classic Mx
- Look out for updates

Possible Extensions

- More than two disorders
- More than one point in time
- More than pairs of twins
- Covariates & GxE
- Models for symptoms (IRT)
- Dynamical systems models
- Generalization to continuous liability