SYSC 551 Discrete Multivariate Modeling
Prof. Martin Zwick
Portland State University
Notes from 1/31/2012
As taken by Juliana Arrighi & checked by MZ

Structures, part 2

3. Lattice of Structures, Structure Types

Nearest Common Ancestor, Nearest Common Descendent (Krippendorf p. 39)

If two different structural models have high goodness measures, we may look either to (a) the nearest common ancestor or (b) the nearest common descendent to (a) merge the two models, and get what's in both of them or (b) select only what they have in common that makes them good models.

To find the **nearest common ancestor** of two structural models in the lattice of structures, take the union of the relations of the two models; that is, combine all component relations of each and eliminate redundancies. For example, the nearest common ancestor of the structural models m_1 = AC:BCDE and m_2 = ABD:CD:CE could be found as follows:

$$m_1 \cup m_2 = AC{:}BCDE \cup ABD{:}CD{:}CE = AC{:}BCDE{:}ABD{:}CD{:}CE = AC{:}BDCE{:}ABD$$

The relations CD and CE were eliminated because they are embedded in BCDE.

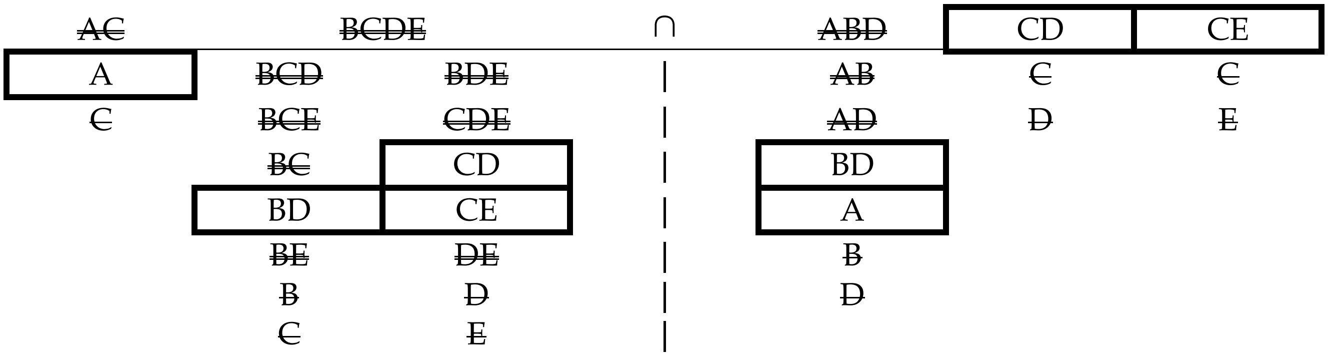
To find the **nearest common descendent**, take the intersection of the two models. The intersection includes all relations that are either components or are embedded in the components of both models. For example

$$m_1 \cap m_2 = AC{:}BCDE \cap ABD{:}CD{:}CE = A{:}BD{:}CD{:}CE$$

A is in both models because it is embedded in both AC and ABD. BD is embedded in BCDE and ABD, and so on. A systematic method for determining the intersection of two structural models is as follows:

1. List all relations and projections of m_1 and m_2
1. Cross out any relation not present on both sides (double strike)
2. Cross out any redundant relation (single strike)

See the following table:



Boxed relations are in both structures.

6. Models With and Without Loops, Disjoint Models

Disjoint Models

Disjoint models are those that have no overlap in their components. We will make a distinction in the criteria for directed and neutral systems.

In a neutral system, a disjoint model will have no overlap in any relations.
Example: AB:CDE
In a directed system, no independent variables overlap in *predicting* relations.
Example: IV:AZ:BCZ, where IV is the relation of all independent variables

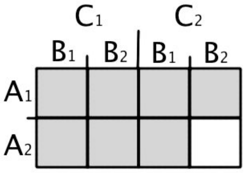
It is important to distinguish between disjoint models and loopless models. In neutral systems, disjoint models are only a subset of loopless models, but in directed systems a disjoint model may contain loops as in the example above. Also unlike with disjoint models, the criteria for looped models is the same in directed and neutral systems.

7. Degrees of Freedom

Krippendorff Method for calculating df, p.48-53

For ABC, df= |ABC| - 1, where |structure| = number of states in the structure
Let cardinality of A be N_A

$df_{ABC}= N_A N_B N_C – 1$
For $N_A = N_B = N_C = 2$, $df = 2 \cdot 2 \cdot 2 – 1 = 7$



For models lower on the lattice of structure, e.g. AB:AC:BC, add the df of the components and subtract the overlap between components.

$df(AB{:}AC{:}BC) = df(AB) + df(AC) + df(BC) – df(A) – df(C) – df(B)$

For $N_A = N_B = N_C = 2$, $df(AB{:}AC{:}BC) = 3 + 3 + 3 – 1 – 1 – 1 = 6$

For ABC:ABD:ACD:BCD, add the df of the components, subtract the df of the overlap between each pair (double overlap) and add the df of the overlap among each set of three components (triple overlap).

Double overlap:
$ABC \cap ABD = AB$
$ABC \cap ACD = AC$
$ABC \cap BCD = BC$
$ABD \cap BCD = BD$
$ABD \cap ACD = AD$
$ACD \cap BCD = CD$

Triple overlap:
$ABC \cap ABD \cap ACD = A$
$ABC \cap ABD \cap BCD = B$
$ABC \cap ACD \cap BCD = C$
$ABD \cap ACD \cap BCD = D$

$$\begin{aligned} df(ABC{:}ABD{:}ACD{:}BCD) = {} & df(ABC) + df(ABD) + df(ACD) + df(BCD) \\ & – df\ (AB) – df(AC) – df(BC) – df(BD) – df(AD) – df(CD) \\ & + df(A) + df(B) + df(C) + df(D) \end{aligned}$$

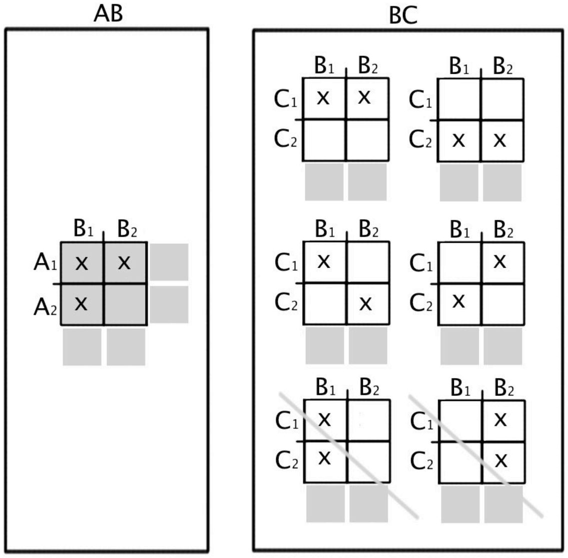
For $N_A = N_B = N_C = N_D = 2$,
$df(ABC{:}ABD{:}ACD{:}BCD) = 4 \text{ x } 7 – 6 \text{ x } 3 + 4 \text{ x } 1 = 14$

For AB:AC
$df(AB{:}AC) = df(AB) + df(AC) – df(A)$

Note that you can replace df by H to get entropy equation, except when there are loops in the structure. Remember that the algebra doesn't work for entropy in these structures with loops, but it does work for df. For example the df of ABC:ABD:ACD:BCD was determined algebraically above, but since the structure has loops, H could not be calculated this way.

Contingency table examples for df.

The table example for ABC was given above. The data table for ABC where $|A|=|B|=|C|= 2$ has 8 values. However only 7 need to be specified. The eighth can be determined by subtracting the other probability values from 1 (or the frequency values from the total sample size).

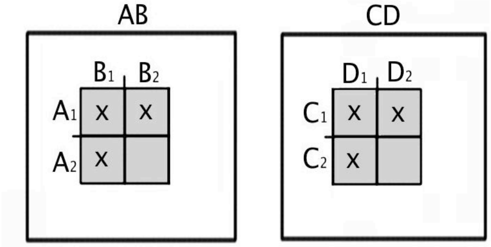


For AB:BC, there are two tables, one for AB and one for BC. Three values need to be specified in AB and only two need to be specified in BC. In the figure, an x represents a specified value. And gray boxes represent values that can be determined from the information specified in the AB table. Both of the B margins are known in the BC table because they can be determined from the AB table. Now, only two more values need to be specified in BC, one in the B_1 column and one in the B_2 column. The remaining values can be obtained by subtracting the specified values from the appropriate B margin value. Specifying both of the values in either column would not be enough since the two values in either column are not independent.

Compare this with the Krippendorff method:

df(AB:BC) = df(AB) + df(BC) – df(B) = 3 + 3 -1 = 5

For AB:CD there are two tables, one for each relation, but for this structure there is no overlap (this is a disjoint structure). Three values need to be specified in each table.



df(AB:CD) = df(AB) + df(CD) = 3 + 3 = 6

Log-Linear method for calculating df (Knoke and Burke p 36-37)

Write down all relations and their projections but do not duplicate projections.
For each relation, multiply one less than the cardinalities of each variable. Add the values for each relation to get df of the structure.

Example: MER:MV:EV, where |M| = |R| = |V| = 2, |E| = 3

Krippendorff Method:

df(MER:MV:EV) = df(MER) + df(MV) + df(EV) – df(M) – df(E) – df(V)
= (2 · 3 · 2 – 1) + (2 · 2 – 1) + (3 · 2 – 1) – (2 – 1) – (3 – 1) – (2 – 1)
= 11 + 3 + 5 – 1 – 2 – 1 = 15

Log-linear method:

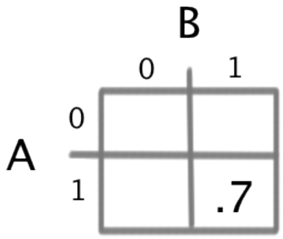
Relations	Product of cardinalities minus one	
MEV	(2-1)(3-1)(2-1)	= 2
ME	(2-1)(3-1)	= 2
MR	(2-1)(2-1)	= 1
ER	(3-1)(2-1)	= 2
M	(2-1)	= 1
E	(3-1)	= 2
R	(2-1)	= 1
MV	(2-1)(2-1)	= 1
V	(2-1)	= 1
EV	(3-1)(2-1)	= 2
	Total	= 15

Log-Linear method is very good for calculating Δdf between two models, since the relations in common can be ignored. The Krippendorff and log-linear methods for calculating df do not apply to models with structural zeros. (e.g. pregnant males)

8. State Based and Latent Variables

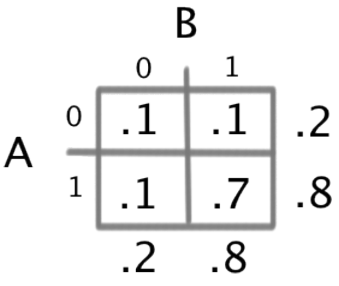
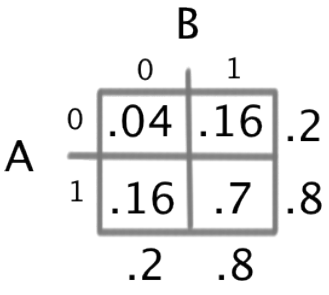
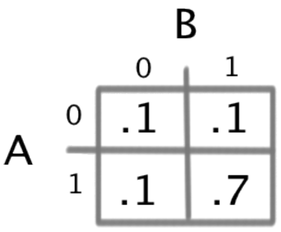
State-Based Models

State-based models specify particular values in the table.
For example A_1B_1 is a state based model. It specifies the value of A_1B_1 in the AB table. In this table, $p(A_1B_1) = .7$



		B	
		0	1
A	0		
	1		.7

A summary the independence model and a state-based model for AB is given below:

AB (p table)	q(A:B)	$q(A_1B_1)$
 B 0 1 A 0 .1 .1 .2 1 .1 .7 .8 .2 .8	 B 0 1 A 0 .04 .16 .2 1 .16 .7 .8 .2 .8	 B 0 1 A 0 .1 .1 1 .1 .7
df = 3 (any three table values)	df = 2 (one A margin, one B margin)	df = 1 (A_1B_1)
	T ≠ 0	T = 0

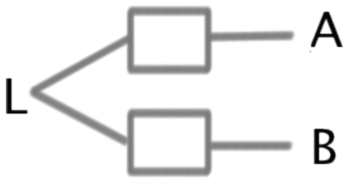
The state-based model, A_1B_1 has only one degree of freedom, because the only constraint is that $p(A_1B_1) = .7$. Entropy is maximized for the set of other probability values, i.e. probabilities or frequencies are uniformly distributed, so margins are irrelevant. Here, A_1B_1 is a simpler model than the independence model, and has no error.

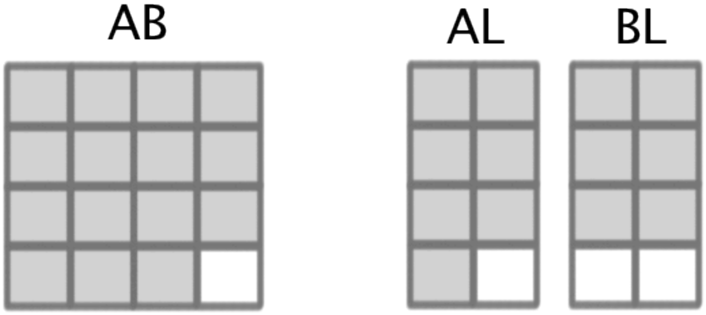
Latent Variable Models

If you have data AB, find ABL such that AL:LB is a good model of ABL. This is a good idea if AL:LB is simpler, i.e., has smaller df, than AB. Latent class analysis is the nominal version of factor analysis

e.g. $|A| = |B| = 4$
$|L| = 2$

$df(AB) = 15$
$df(AL{:}LB) = (4 \cdot 2 - 1) + (4 \cdot 2 - 1) - (2 - 1) = 13$





9. Discussion: Complexity and Decomposability

In reconstructability analysis, complexity is the same as degrees of freedom. However there is more than one way to quantify complexity. For example, consider following equations:

$$z = ax + by$$

$$z = \left(\sqrt[\mathrm{int}(a)]{\tanh(by)}\right)^{\mathrm{int}(ax)!}$$

The second equation seems more complex, although each equation has the same number of variables. Function form could, in principal, enter into a complexity calculation.

Other complexity measures:
Minimum description length – this makes use of functional form in calculating complexity

vonBertalanffy’s progressive segregation, systematization

↑Systematization (compose, complexify)↑ ↓Segregation (decompose) ↓

10. Grouping Structure Types (R, C, P Structures)

The lattice of all possible structures can be broken up into ρ, C and P structures

ρ groupings are determined as follows. In ρ_1 all variables are directly connected to all other variables; that is, they are separated in the structure graph by only one box. In ρ_2, one pair of variable is not directly connected, i.e. those two variables are separated in the structure graph by 2 boxes.

C structures are the most complex of each ρ group. For example in ρ_1 group, the saturated model is the most complex, because the variables are the most interrelated.
P structures are the simplest in each ρ group. In the ρ_1 group for four variables, AB:AC:AD:BC:BD:CD is the simplest way for all variables share a relation with all other variables because this is the only ρ_1 structure with only dyadic relationships.

Search types:

Hierarchical search using ρ, C and P structures: First search representatives of ρ groups by searching among only C or P structures; then, for some given C or P structure, search within its ρ group

Beam (what OCCAM does now): Find the best ‘width’ number of parent models, going up (or child models, going down); from these best models, then consider the best ‘width’ of their parents (or children), etc., as one goes up (or down) from level to level. (One could do a beam search ‘breadth first’ by having a large ‘width’ parameter going up (or down) hopefully only a modest number of levels, or ‘depth first’ by having a small ‘width’ parameter but going up (or down) many levels.)