II. STRUCTURES

Exercises

Midterm 2018: 3
Midterm 2019: 5-8
Midterm 2021: 3
Final 2019: 5b

1. Introduction

1.1 Aims

(1) What **structures** (candidates for models) are there, and

(2) How evaluate the **complexity** of a structure

1.2 Nomenclature: models & structures

Model = structure + data

Structure = data-independent, except for variable **cardinalities**

Exploratory modeling is searching over structures, each – with data – giving a model of the data, and then evaluating model.

Need an ***explicit representation*** of structures that are possible; this **not** well **articulated** in **log-linear** literature, which tends to assume that we're doing confirmatory modeling. The structures that are possible are called/organized in the **Lattice of Structures.**

One might want only **certain kinds** of structures which would define **sub-lattices** of the Lattice of Structures.

1.3 Two criteria for good model

Evaluate models on two criteria:

1. (positive formulation:) **information** (captured) in model
   (negative formulation:) information lost in model = **error**
2. (positive formulation:) **simplicity** of model
   (negative formulation:) **complexity** of model

Error(model) = **T**(model) **= distance down to model from top reference (data)**

Information(model) **= distance up to model from bottom reference (independence)**

Both are information theoretic measures. If doing prediction, one might use model selection criterion of **%correct**, but this ***not an information theoretic measure***,

%correct **isn't collinear** with information (though it's *roughly so*).

Complexity is **degrees of freedom, df**. Will explain this soon.

This doesn't exhaust idea of complexity, but it's all I know how to use.

A "good model" has

high information & high simplicity	or (equivalently)
high information & low complexity	or (equivalently)
low error & low complexity	or (equivalently)
low error & high simplicity	

1.4 Significance of complexity

***Why does one care* about simplicity/complexity?**

To avoid **overfitting**, to **generalize successfully** to new data.

Show data points nearly linear what overfitting does.

Overfitting shows up in **plots of information (or error) vs. complexity** in **training** & **test** data. In **training** data, one fits model to the data, and **information monotonically increases (& error monotonically decreases)** with complexity. SHOW SUCH PLOTS.

But in **test data**, using models fitted with training data, information or error has a **sweet spot with respect to complexity**: **information** has a **local maximum** (or **error** has a local **minimum**). Want to know what that **sweet spot** is.

Have various measures (BIC, AIC, p-values, etc.) that **guess** at that spot.

1.5 How use two criteria to pick best model (guess at sweet spot)

1.5.1 Tradeoff (integrate the two criteria)

Different ways to tradeoff:

1. **Statistical significance**, probability(Type I or II error), using Chi-square distribution. As noted earlier, get L^2 and df (or Δdf) and go into Chi-square table and get p. (Here 'p' is not data!)

2. **Integrated measures** AIC (Akaike Information Criterion), BIC (Bayesian Information Criterion), which do **linear combination of error and complexity**. Also there's a modified-AIC for small sample sizes & other variations on AIC, BIC.

You would think that statistical **significance** is all about **generalizability**, but it turns out that integrated measures do better for p(cutoff)=0.05, the traditional value.

Specifically **BIC** works best in my experience. But tends to be **conservative**: guesses a model that is less complex than sweet spot. **Show sweet spot**.

1.5.2 Optimize one criterion subject to constraint of the other

E.g., want simplest model with 90% of information.

1.5.3 Pick model empirically (if have enough data)

Split data 3-way into **training**, **evaluation (pseudo-test)**, **test**.

Pick model that **generalizes** best from **training** data to **evaluation** data.

2. Lattice of relations, ordinality

SHOW THE (3-variable) LATTICE OF RELATIONS from K, p.37

Can one get a higher ordinality relation from set of lower ordinality relations?

No. this is Angyal's argument that **dyadic relations are not (in general) sufficient** (but for some data they are sufficient). One can't in general get triadic from all dyadic projections.

K, p.34, talks about difference with modeling that implicitly or explicitly assumes only dyadic relations.

Interestingly enough, when parts are insufficient to specify the whole, the whole turns out to be *less than* the 'sum of the parts' in that it is less in the ***number of states*** observed (**lower entropy**) compared to the number of states implied by the parts.

But the whole is more than the 'sum of the parts' in the **constraint** of the whole (the whole is more constrained than sum of constraints of parts).

When we say that the whole is **more** than sum of parts", we mean this latter notion: that *constraint* reflected in whole *is **more** than* the sum of the separate constraints manifested in parts.

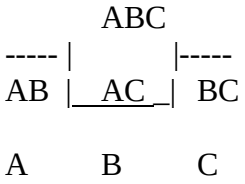
Ordinality & cylindrance

K, p.71. reduced lattice of all monadic terms (m^1), all dyadic terms, etc. all the way up to data. Useful for searching large lattices. A sub-lattice of Lattice of Structures.

3. Lattice of structures; structure types

Structure is set of relations, and these don't have to be same ordinality.

A structure is a **cut thru lattice of relations**. For example, AB:BC is this cut, includes everything below the cut



K also discusses it but diagram isn't that clear (**K, p. 37)**.

Go over (3 variables), lattice of structures on **K, Figure 25, p.40**. Then explain grouping of these into structure types "general structures".

Note that for 4 variables, there are many general structure, and still many more specific structures (by **permuting** variables, but not simply # structure types x 4!, since some permutations will generate same structure). SHOW INTRO, slide 19

4. Directed vs. neutral systems

Show (3 variables) that directed systems have fewer models than neutral systems INTRO, slide 18

Not all models that one can imagine, e.g., AZ:BZ !!

(consider Z replacing C, so this comes from AC:BC)

(a "naïve Bayes" model); but we will ultimately include this in Occam (or AZ:BCZ)

5. Generating the lattice of neutral structure types

5.1 Generating downwards

Go thru, for 4 variables, generation of part of lattice of structure types. Process of "refinement" or simplification.

Occam also has algorithm to generate upwards.

There are other strategies for generating lattice: depth first, etc.

For sub-lattices, one wants algorithms to **generate only** the **sub-lattice**, so don't need to create, then discard; one wants a downwards and upwards algorithm. Show OCCAM MANUAL, p.39, of what searches are implemented now.

5.2 Nearest ancestor, descendant

Go over nearest common **ancestor**, **nearest** common **descendent** (**K, p.39**).

Also **furthest** common ancestor and descendant: data (saturated model) & independence model)

5.3 Simpler models & reduced cover

Bottom model is independence model.

For some purposes, **go lower** and use **uniform** distributions as hypotheses (recall we could test **univariate** deviation from uniformity).

For 3 variables, along with AB:C → A:B:C, one has AB:C →AB:Φ, Φ is uniform.

Also A:B:C → A:B: Φ → A: Φ:Φ → Φ:Φ:Φ = Φ.

Using A:B:C as bottom keeps cover at 3 variables.

But could also consider an **augmented Lattice of Structures** that includes all lattices with reduced cover (fewer variables).

6. Models with & w/o loops; disjoint models

6.1 Loops

Algorithm for detecting loops in structures: **K, p.42**. *(It would be nice to have an algorithm that detected loops in structure matrix used in the linear algebra representation of the set of constraints.)*

Difference between loops as defined here and loops as defined in logic of causal order: here **directions don't matter**.

Useful also to define the subset of loopless structures, because these are computationally much easier to analyze. **Fewer**: show slide 20 from **INTRO**.

Consider 30 binary variables. $2^{30} = 10^9$ = state space.

For directed systems IV component can be used to reduce state space; for neutral systems, where this cannot be done, loops may make state space impossible.

Loopless models for **feature selection** in directed systems; one predicting component.

6.2 Disjoint models

Difference in Occam between **disjoint** models for **neutral** and **directed** systems

Still **simpler** than **loopless for neutral systems. Not so for directed systems**

Full decomposability, re Simon's "near-decomposability". See **K, figure 41, p.80**. also relevant for **aggregations of states** for optimal binning. This lattice would use states rather than variables and show all possible disjoint aggregations. Note that we could use state aggregations that are not disjoint. Then ordinary lattice would be used.

Used for RAGA; could be used to break up optimization problems.

6.3 Chain models

For RAGA study only; not of general interest.

6.4 Occam implementations

			Implemented?	
			variable-based	state-based
directed	up	all	yes	**yes**
directed	up	disjoint	yes	no
directed	up	loopless	yes	**yes**
directed	down	all	yes	no
directed	down	disjoint	**no**	no
directed	down	loopless	yes	no
neutral	up	all	yes	**yes**
neutral	up	disjoint	yes	no
neutral	up	loopless	yes	no
neutral	down	all	yes	no
neutral	down	disjoint	yes	no
neutral	down	loopless	yes	no
directed	up*	chain	yes	n/a
neutral	up*	chain	yes	n/a

* n/a = not applicable. For chain models, "up" vs. "down" searches are meaningless, but one needs to specify "up" to get a chain search done.

7. Degrees of freedom

Must clearly differentiate between **df and Δdf**, where **Δ** is relative to some reference (top or bottom). Must not confuse the two.

df is number of parameters, high for top model, i.e., data.

We will look at Δdf (OCCAM calls it dDF) for **transitions between models**, i.e., for degrees of freedom LOST in going down from one model to another, or GAINED in going up, and will use these in conjunction with information distance between models.

Alternating sign expression for df: K, p.49, equation 10.2

df(AB:BC) = df(AB) + df(BC) – df(B).

For binary variables, df(AB:BC) = 3 + 3 – 1 = 5.

df(AB) needs 3 parameters, BC adds only 2.

Consider from K&B book the model: MER: MV: EV where M = R = V = 2, E = 3.

df(MER:MV:EV) = df(MER) + df(MV) + df(EV)
– df(MER ∩ MV) – df(MER ∩ EV) – df (MV ∩ EV)
+ 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

Alternative Log-Linear recipe for df calculation (K & B, p.36-7, but obscure)

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

Now calculate it in K&B way (based on interaction accounting).

MER: (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
M already counted
V: (2-1) = 1

EV: (3-1) (2-1) = 2
E already counted
V already counted

Sum = df(MER:MV:EV) = 2 + 2 + 1 + 2 + 1 + 2 + 1 + 1 + 1 + 2 = 15

df(data) = df(MERV) = 2 3 2 2 - 1 = 23

Δdf = df(MERV→ MER:MV:EV) = df(MERV) – df(MER:MV:EV) = = 23 - 15 = 8

OCCAM uses both ways of calculating df: K is old way, LL is new way.

Normalized df

df(model) – df(independence)
---
df(data) – df(independence)

Use LL for small differences between very large df values.

All this doesn't work to calculate df for models with **structural zeros**. More on this later.

SHOWING THAT K METHOD AGREES WITH LL METHOD

Illustrating summing df(components), then subtracting overlap:

Here using Krippendorff notation where N is the number of states, NOT sample size

0	ABC	N_{ABC}-1	
1	AB:AC:BC	N_{AB}-1 + N_{AC}-1 + N_{BC}-1 df of AB, AC, BC separately First term in eqn-10.2 (**K, p.49**)	- (N_A-1) - (N_B-1) - (N_C-1) projections must agree 2nd term
			= N_{AB} + N_{AC} + N_{BC} - N_A - N_B - N_C
2	AB:AC	N_{AB}-1 + N_{AC}-1 - (N_A-1)	= N_{AB} + N_{AC} -N_A -1
3	AB:C	N_{AB}-1 + N_C -1	= N_{AB} + N_C - 2
4	A:B:C	N_A-1 + N_B-1 + N_C-1	= N_A + N_B + N_C – 3

Going down just one step:

df(ABC→AB:AC:BC) = [N($_{ABC}$) -1] - [N($_{AB}$) + N($_{AC}$) + N($_{BC}$) - N$_{(A)}$ - N($_B$) - N($_C$)]

= N_{ABC} - N_{AB} - N_{AC} - N_{BC} + N_A + N_B + N_C - 1

= N_{ABC} - N_{AB} - N_{AC} - N_{BC} + N_A + N_B + N_C – 1

= $N_AN_BN_C$ -N_AN_B -N_AN_C -N_BN_C +N_A +N_B +N_C -1

= **(N_A-1) (N_B-1) (N_C-1) as LL method indicates (K, p.50)**

This is basis of **log-linear way of calculating df**

dF IS ALSO RANK OF STRUCTURE MATRIX.

8. State-based & latent variable models

8.1 State based RA

As Ashby noted, manipulations of variables are less general than manipulations of states.

So far, lattice of structures keeps all variables intact. We could consider working instead with states.

Show example of .1, .1, .1, .7. Show that df=1 gives better fit than df=2, the independence model.

df 3

XY

	y_1	y_2	
x_1	.1	.1	.2
x_2	.1	.7	.8
	.2	.8	1

2

X:Y

	y_1	y_2	
x_1	.04	.16	.2
x_2	.16	.64	.8
	.2	.8	1

1

X_2Y_2

	y_1	y_2	
x_1			
x_2		.7	
			1

T(X:Y) > 0, with df = 2

But $T(X_2Y_2) = 0$ with df = 1 A simpler model with less (no) error!

Note here need **uniform** distribution model as **bottom** model.

This proposed by Bush Jones (but joined to k-systems, a different idea).

SB modeling is much more general; **many more structures**.

Show **Occam manual** what SB structures considered.

Selected Works page has two state-based papers.

8.2. Latent variable based models

Example AB, **|A|=|B| = 4, df(AB) = 15.**

Assume T(A:B) > 0, i.e., there is constraint between A and B

Find some ABL distribution, where |L| = 2, such that

AB projection of ABL agrees with data, & for which

T(AL:LB) = 0, i.e., AL:LB has all the information in ABL

df(AL:LB) = 7+7-1=13

AL:LB is a latent-variable based simplification of the data.

Possibility of "deep RA," i.e., deep latent-variable RA, like deep (NN) learning

9. Discussion: complexity & decomposability

Can think of all this as a **methodology** to deal with data.

Can *also* think of it as way of describing how systems are organized, how they change over time, as about **real world**.

Complexity = df means **highest order relation most complex**

Certainly it is the most holistic.

But other issues: if two structures have same df, but one has **loops** and the other doesn't, shouldn't this factor into complexity?

Minimum Description Length idea.

Disjoint (neutral) structures have fully separate subsystems (re near decomposability of Simon).

Progressive segregation and **systematization** of von Bertalanffy.

Would be nice to show a **movie** of system **changing diachronically** by going up or down lattice.

Advantages and disadvantages of top structure and of bottom structure

Related to general philosophical issue of *reductionism* and *holism*.

Are parts sufficient to specify the whole?

There are situations where parts *are* sufficient and there are different situations where parts *are not* sufficient to determine, specify, and exhaust the whole. If there is no error in a decomposed structure, then its parts ARE sufficient to specify the whole.

10. Grouping structure types (ρ, C, P structures)

Vertical levels define complexity for K.

There are other ways to group structures other than by complexity.

Klir (pp. 237). **Rho-structures** based on whether each variable is directly connected to others.

These divide structure types into classes. Within each class, can defined the least refined (most data-true) and most refined structures, call these C and P. these also form lattices.

To see how this classification fits K's lattice: see Klir, p.238.

These various groupings can aid in searching lattice for a good model of data for cases where the number of variables make the structure-type lattice too big.

They allow for **hierarchical search**.

V-structures in Bayesian graphs look like P but are C!! See Marcus paper on relation of BN to RA.