Modeling Crop Rotation with Discrete Mathematics

Kellen Myers, Victoria Ferguson, Yekaterina Voskoboynik*

<u>Course Packet</u>

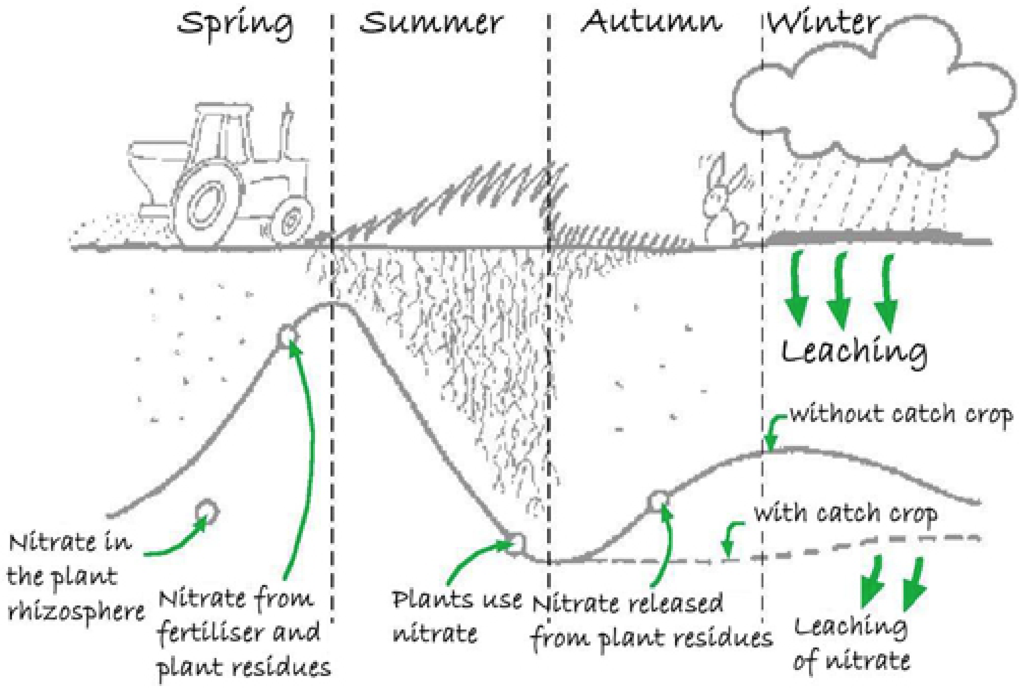
This packet contains several pages – modified somewhat – meant to be ready to photocopy and give directly to students working through the module. Not all pages apply to every class (e.g. the tables of crops used for exercises) so take care when printing the master copy that you will photocopy. Irrelevant pages should be omitted. Changes to this packet can also be made, to tweak the exercises, examples, etc. to fit the course or student background level.

Agricultural Background

Plants are divided into three primary groups: cash, soil, and forage crops. They are defined as follows:

1. Cash crops are those that will be used for income, or processed for the community or private good. These are such thing as sweet corn, popcorn, field corn (for animal use), tomatoes, peppers, potatoes, cabbages, squash, cucumbers, beans, some flowers, herbs.
2. Soil crops are those that give back to the soil, or trap nutrients so that they are not leached out of the soil. These are also known as cover crops, which are further divided into catch crops ('catching' nitrogen) and green manure (which deposit nitrogen through their own processes). Most of these are not income producers.

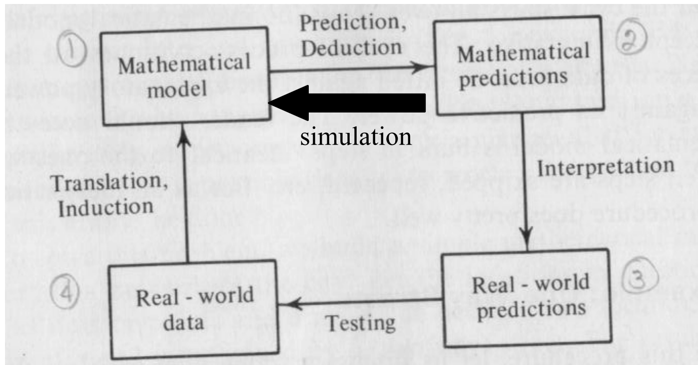
Forage crops are those which are used for grazing animals. Many of these are also cover crops. Forage crops are not usually income producers, but they may be when dried and sold to other livestock farmers. One of the primary concerns addressed by crop rotation is nitrogen depletion in soil.



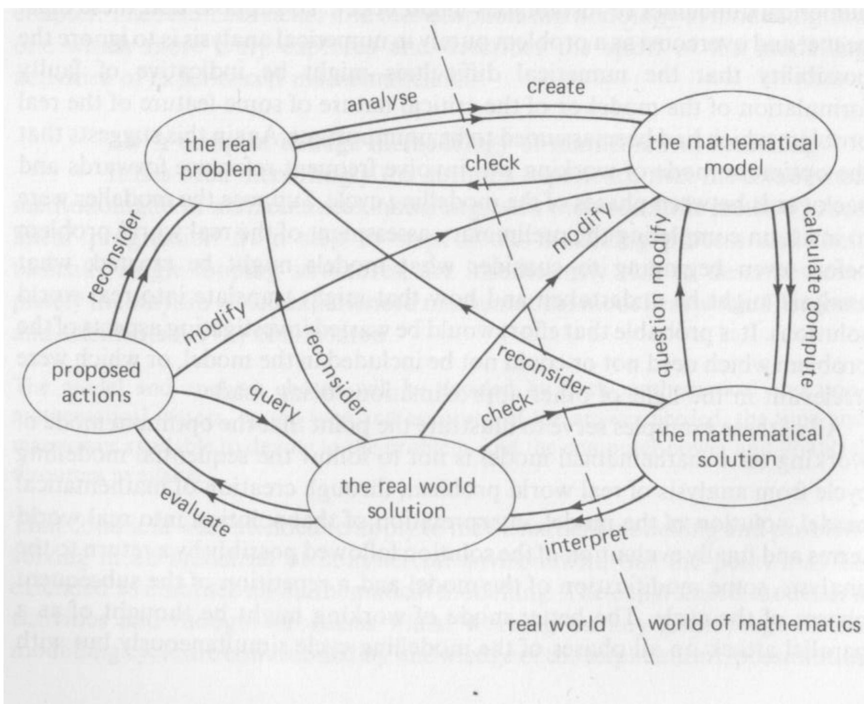
Typical Nitrogen Cycle over a year [ØST]

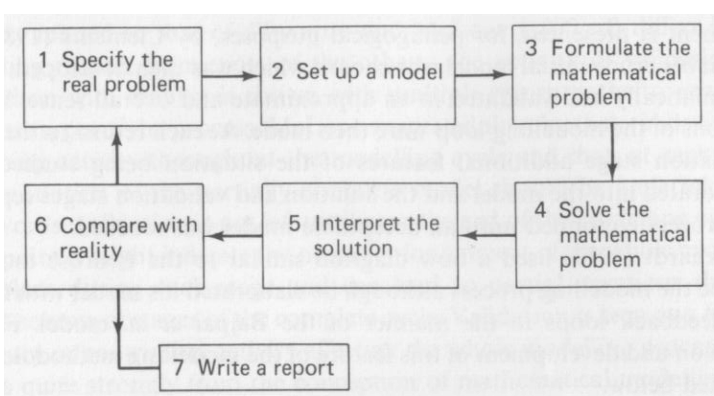
The Nature of Mathematical Modeling

This illustration, from *Discrete Mathematical Models* by Fred S. Roberts, shows the four-step process of modeling. An arrow indicating the role of simulation has been added.



The following two illustrations provide a more complex illustration of mathematical modeling. They are taken from *Mathematical Modeling* by R. R. Clements.





The Definition of our Model

For the purposes of this lesson, we devise the following model for a crop rotation.

A mathematical crop rotation consists of three structures with the following properties:

1. a set of crops, wherein each crop has three attributes:
   (a) the type of the crop (e.g. consumer crop)
   (b) the soil properties of the crop (e.g. nitrogen fixer)
   (c) a set of crops that are contraindicated as neighbors for the crop
2. a planar region, subdivided into smaller subregions
3. a function that assigns to each subregion an ordered list of crops

Block Designs

A block design is an abstract mathematical structure. The most basic notion of a block design is simply a set S, sometimes called the varieties, and a set **B** of which each element B is a subset of S. Each B is called a block, and **B** itself is sometimes referred to as the design or block design. This is fair, in that S could be inferred from **B**, at least in the sense that any elements of S that do not occur in any member of **B** are mostly irrelevant.

Two parameters inherent to all block designs are v = |S| and b = |B|. The v stands for variety, instead of using s or some other letter. There are other properties that may also be imposed on block designs, some of which result in further useful parameters:

1. If every block has the same size, then **B** is said to be uniform.* We let k = |B| for all B in **B**.
2. If **B** is uniform and if every element of S occurs in the same number of blocks, then we let r be the number of blocks in which each element of S occurs. Here **B** is said to be regular.
3. If **B** is regular, and if r = k, then **B** is called complete.

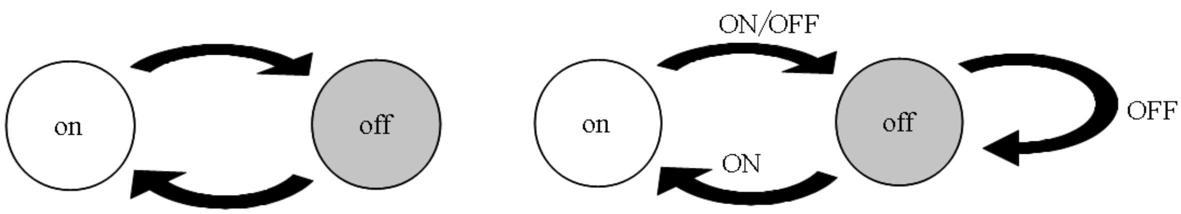
Latin squares are an example of complete block designs that give interesting structure – each block is endowed with an order (something quite relevant to our model, since this is a part of our subject that is not incorporated into basic block designs). A complete block design is a Latin square if each block is given an order B(i) = B(i,1), B(i,2) , … B(i,k), and no two blocks ever agree at a given "time" that is, if for any i, j, and t, if B(i,t) = B(j,t), then i = j.

Latin squares designs are "squares" in the sense that they can be arranged with blocks listed in order as rows, and the number of rows and columns is equal. Each row contains each element of S exactly once, and so too does each column – that is what the B(i,t) = B(j,t) condition means. In this way, Latin squares also reflect a geometric property, similar (indeed, a much stronger condition) to some of the conditions we might impose about crop adjacency.

Finite State Automata / Machines

The most basic structure is a Finite State Automaton (FSA), which is a machine that is fixed throughout its entire computation. The FSA may have input, but this input is a finite amount of input that is known when the machine begins computing. In that sense, the input determines a fixed machine that then carries out the computation with no further input during the computation.

A Finite State Machine (FSM) is an FSA but with the additional parameter that input at any point during the computation might determine different transitions. For this reason, an FSM has much more complexity than an FSA, although neither is as complex as the standard model of computation, the Turing Machine (which is, itself, similar to actual computers).



A blinking light, as an FSA and an FSM (with ON/OFF switch)

A Deterministic Finite State Automaton/Machine (DFSA or DFSM) is the same as above, except that certain states are considered to be "final" states, at which point the computation is considered to be finished (and, if "output" is desired, the output is the corresponding final state).

For advanced (or some basic) mathematics students, it is useful to give the standard mathematical definition of each structure. A FSA consists of four things, the tuple (a,S,s,f), where:

- A is the input alphabet, the set of all possible inputs;
- S is the set of all possible states;
- s1 is the initial state, a member of S;
- f is the transition function, a function that maps SxA to S.

Here, the transition function is dependent on the input a, but there is only one initial input. So the function would be computed as s2 = f(s1,a), then s3 = f(s2,a), s4 = f(s3,a), etc.

A FSM is defined only slightly differently (A,S,s,f):

- A is the input alphabet;
- S is the set of all possible states;
- s1 is the initial state;
- f is the transition function, a function that maps SxA to S;

Everything here is the same except that there is a different input at each step, so instead of what is above, we get s2 = f(s1,a1), then s3 = f(s2,a2), s4 = f(s3,a3), etc.

And in the case of a deterministic FSA or FSM, the set F is also added to the structure:

- F is the set of final states, a subset of S, for which f is not necessarily defined.

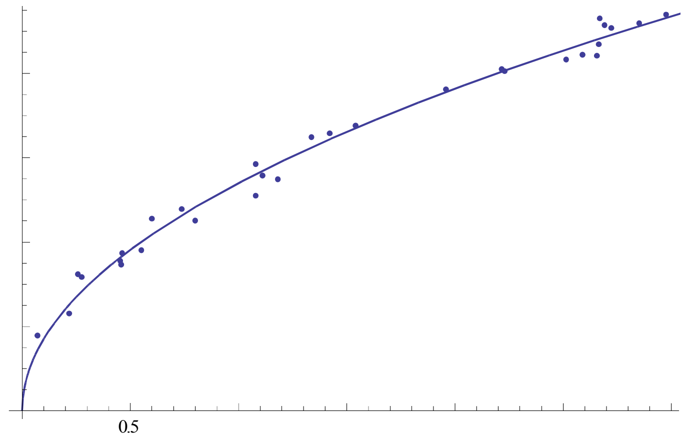
One important caveat for DFSA or DFSM is that, in addition to allowing terminal states, we must require that no cycles exist in the machine – in other words, that a final state *must* be reached, regardless of the input. If the machine is drawn as a directed graph, this means the digraph is acyclic.

An Example of Modeling

How long does it take an object to fall to the ground when dropped?

Consider the following dataset and graph:

height	drop time
0.277454	0.316615
0.454152	0.356153
1.10958	0.557436
1.95683	0.762136
2.2312	0.804299
0.600281	0.456651
0.738612	0.477814
2.58771	0.843939
1.17919	0.548234
2.85075	0.918419
1.07743	0.51068
2.66808	0.930758
0.799976	0.450971
2.21392	0.809947
2.6904	0.9151
0.257097	0.323524
0.459196	0.347605
2.65682	0.841634
2.72058	0.908641
0.0691029	0.178125
1.5414	0.676535
0.218348	0.231829
0.463968	0.373768
0.550272	0.381364
2.66399	0.869332
1.33635	0.648566
2.97359	0.939772
1.07925	0.585594
2.5144	0.833181
1.42243	0.657192



In this case, the "model" of this phenomenon is simply a function that produces the time it takes an object to fall. Based on observation, we can suppose that for most objects, this is only a function of the initial height when dropped. Try to discern what equation best fits this data.

If you can obtain, by some means, a function that gives the height of the object, h(t), you can set this function equal to zero to find the time at which the object hits the ground. Does it relate to the initial height? How? Does this match your expectations based on the data?

In what cases is this model insufficient? How could the model be improved or modified to be more accurate, more often?

References & Further Reading

Agriculture & Horticulture

[BUL] Bullock, D.G., *Crop Rotation*, Critical Reviews in Plant Sciences, 1992.

[DAB] Dabney et al., *Advances in Crop Management*, 2008.

[HAA] Haas, G., H. Brand, M. Puente de la Vega, *Nitrogen from Hairy Vetch (Vicia villosa Roth) as Winter Green Manure for White Cabbage in Organic Horticulture*, Biological Agriculture and Horticulture, 2007.

[HOO] Hoorman, J.J., et al., *Sustainable Crop Rotations with Cover Crops*, Fact Sheet: Agriculture and Natural Resources, The Ohio State University Extension, 2009.

[ØST] Østergaard HS, *Catch Crops*, Baltic Deal, www.balticdeal.eu, 2010 .

[REE] Reeves D.W., *Cover Crops and Rotations*, CRC Press, Inc, 1994

[SUL] Sullivan P., *Overview of Cover Crops and Green Manures: Fundamentals of Sustainable Agriculture*, ATTRA, www.attra.ncat.org, 2003

Block Designs

[BET] Beth, Thomas, et al., *Design Theory*, Cambridge University Press. 2nd ed., 1999.

[CAM] Cameron, P. J. & J. H. van Lint, Designs, Graphs, Codes and their Links, Cambridge University Press, 1991.

[HAL] Hall Jr., Marshall, *Combinatorial Theory*, 2nd ed., Wiley-Interscience, 1986.

[HUG] Hughes, D.R. & E.C. Piper, *Design theory*, Cambridge University Press, 1985.

[STI] Stinson, Douglas R., *Combinatorial Designs: Constructions and Analysis*, Springer, 2003.

[STR] Street, Anne P. & Deborah J. Street, *Combinatorics of Experimental Design*, Oxford University, 1987.

[WAL] Wallace, W.D., *Combinatorial Designs*, Marcel Dekker, 1988.

Finite State Machines

[CAR] Carroll, J. & D. Long, *Theory of Finite Automata with an Introduction to Formal Languages*, Prentice Hall, 1989.

[KOH] Kohavi, Z., *Switching and Finite Automata Theory*, McGraw-Hill, 1978.

[GIL] Gill, A., *Introduction to the Theory of Finite-state Machines*, McGraw-Hill, 1962.

[GIN] Ginsburg, S., *An Introduction to Mathematical Machine Theory*, Addison-Wesley, 1962.

Mathematical Modeling

[ARI] Aris, Rutherford, *Mathematical Modelling Techniques*, Dover, 1978.

[BEN] Bender, E.A., *An Introduction to Mathematical Modeling*, Dover, 1978.

[CLE] Clements, R.R., *Mathematical Modeling: A Case Study Approach*, Cambridge University, 1989.

[LIN] Lin, C.C. & L. A. Segel, *Mathematics Applied to Deterministic Problems in the Natural Sciences*, SIAM, 1988.

[GER] Gershenfeld, N., *The Nature of Mathematical Modeling*, Cambridge University, 1998.

[ROB] Roberts, Fred. *Discrete Mathematical Models with Applications to Social, Biological, and Environmental Problems*, Prentice-Hall, 1976

[ROB2] Robertson, David et al., *Eco-Logic, Logic-Based Approaches to Ecological Modelling*, Massachusetts Institute of Technology, 1991.

Full Table of Crops

Crop Family	Common name crops	Type (Season*)	Properties	Interactions**	Notes
Grains **N depletion** [x]	corn (maize)	cash, forage			
	cereal rye	forage, cover	1,4-8,11-13		
	annual ryegrass	forage, cover	1,4-8,11		
	sorghum-Sudan grass	forage, cover	1,4-8,10		
	oats	cash, forage, cover	1,3-8,11,13		
Brassicacea	cabbage	cash (F)		dill, beets, lavend	
	mustard	cash, cover	8, deters worms		
	kale	cash (Sp, F, W)	deters worms		
	turnip	cash, cover	deters worms		
Fabaceae	clover	cover	2,7,9-11	garlic	
	vetch	cover	2,7,10	garlic	
	alfalfa	forage, cover	2	garlic	
	cowpea	cover	2,3,6,7,10	garlic	
	mungbean	cash, cover	2,6,10	garlic	
Solanaceae **must not be in same field within 3 years**	tomatoes	cash		dill, beets	
	potatoes	cash		pumpkins	
	peppers	cash			
	eggplants	cash			
Cucurbitaceae	cucumbers	cash (Su)	deters aphids		
	melons	cash (Su)			
	squash	cash (Su, F, W)			
	pumpkins	cash (Su, F)		potato	
Herbs [y]	dill	cash (F)		tomato, cabbage	
	lavender	cash (Su,F)		cabbage	
	calendula	cash/cover (Su,F)	8,9		medicinal
Aliaceae	garlic	cash (Sp)		all beans	
	onion	cash (Sp)			
	leek	cash (Sp)			
	scallion	cash (Sp)			
Betaceae	beets	cash (Sp,W)		tomato, cabbage	cool weather

Key:

Sp	Spring	6	prevents soil erosion
Su	Summer	7	recaptures nutrients
F	Fall	8	weed suppressant
W	Winter	9	attracts beneficial insects
1	increases carbon : nitrogen ratio	10	heat/drought tolerance
2	increases nitrogen	11	wet soil tolerance
3	easy non-herbicidal removal	12	cold tolerance
4	compaction reduction (deep roots)	13	nurse crops
5	grazing crop		

* Crop season noted are harvest times – rotations should be labeled accordingly. No label means all seasons.

** These are the “forbidden adjacency” parameters. Crops cannot be adjacent to other crops listed in this column.

x Notes in this column apply to all members of the family.

y Herbs are not all from the same family, but are listed together here as a matter of simplicity.

Table of Crops for use in the Assignment

Crop Family	Common name crops	Type (Season*)	Properties	Notes
Grains	corn (maize)	cash, forage		
N depletion [x]	cereal rye	forage, cover	1,4-8,11-13	
	annual ryegrass	forage, cover	1,4-8,11	
	sorghum-Sudan grass	forage, cover	1,4-8,10	
	oats	cash, forage, cover	1,3-8,11,13	
Brassicacea	cabbage	cash (F)		
	mustard	cash, cover	8, deters worms	
	kale	cash (Sp, F, W)	deters worms	
	turnip	cash, cover	deters worms	
Fabaceae	clover	cover	2,7,9-11	
	vetch	cover	2,7,10	
	alfalfa	forage, cover	2	
	cowpea	cover	2,3,6,7,10	
	mungbean	cash, cover	2,6,10	
Solanaceae	tomatoes	cash		
must not be in	potatoes	cash		
same field	peppers	cash		
within 3 years	eggplants	cash		
Cucurbitaceae	cucumbers	cash (Su)	deters aphids	
	melons	cash (Su)		
	squash	cash (Su, F, W)		
	pumpkins	cash (Su, F)		
Herbs [y]	dill	cash (F)		
	lavender	cash (Su,F)		
	calendula	cash/cover (Su,F)	8,9	medicinal
Aliaceae	garlic	cash (Sp)		
	onion	cash (Sp)		
	leek	cash (Sp)		
	scallion	cash (Sp)		
Betaceae	beets	cash (Sp,W)		cool weather

Key:	Sp	Spring	6	prevents soil erosion
	Su	Summer	7	recaptures nutrients
	F	Fall	8	weed suppressant
	W	Winter	9	attracts beneficial insects
	1	increases carbon : nitrogen ratio	10	heat/drought tolerance
	2	increases nitrogen	11	wet soil tolerance
	3	easy non-herbicidal removal	12	cold tolerance
	4	compaction reduction (deep roots)	13	nurse crops
	5	grazing crop		

* Crop season noted are harvest times – rotations should be labeled accordingly. No label means all seasons.

x Notes in this column apply to all members of the family.

y Herbs are not all from the same family, but are listed together here as a matter of simplicity.

Table of Crops for use in the Assignment

Crop Family	Common name crops	Type
Grains **N depletion** [x]	corn (maize) cereal rye oats	cash, forage forage, cover cash, forage, cover
Brassicacea	cabbage mustard kale turnip	cash cash, cover cash cash, cover
Fabaceae	clover vetch mungbean	cover cover cash, cover
Solanaceae **must not be in same field within 3 years**	tomatoes potatoes peppers eggplants	cash cash cash cash
Cucurbitaceae	cucumbers melons squash pumpkins	cash cash cash cash
Herbs [y]	dill lavender	cash cash
Aliaceae	garlic onion	cash cash
Betaceae	beets	cash

x Notes in this column apply to all members of the family.

y Herbs are not all from the same family, but are listed together here as a matter of simplicity.

Table of Crops for use in the Assignment

Crop Family	Common name crops	Type	Properties	Interactions*
Grains	corn (maize) cereal rye annual ryegrass sorghum-Sudan grass oats	cash, forage forage, cover forage, cover forage, cover cash, forage, cover	3 1,3 1,3 1,3 1,3	
Brassicacea	cabbage mustard kale turnip	cash cash, cover cash cash, cover		dill, beets, lavender
Fabaceae	clover vetch alfalfa cowpea mungbean	cover cover forage, cover cover cash, cover	2 2 2 2 2	garlic garlic garlic garlic garlic
Solanaceae **must not be in same field within 8 turns** [x]	tomatoes potatoes peppers eggplants	cash cash cash cash		dill, beets pumpkins
Cucurbitaceae	cucumbers melons squash pumpkins	cash cash cash cash	deters aphids	 potato
Herbs [y]	dill lavender calendula	cash cash cash/cover		tomato, cabbage cabbage
Aliaceae	garlic onion leek scallion	cash cash cash cash		all beans
Betaceae	beets	cash		tomato, cabbage

Key:
1 increases carbon : nitrogen ratio
2 increases nitrogen
3 nitrogen depletion – unless labeled (2), all crops deplete nitrogen to some amount! These are just the worst offenders and nothing but those labeled (2) should follow.

* These are the “forbidden adjacency” parameters. Crops cannot be adjacent to other crops listed in this column.

x Notes in this column apply to all members of the family.

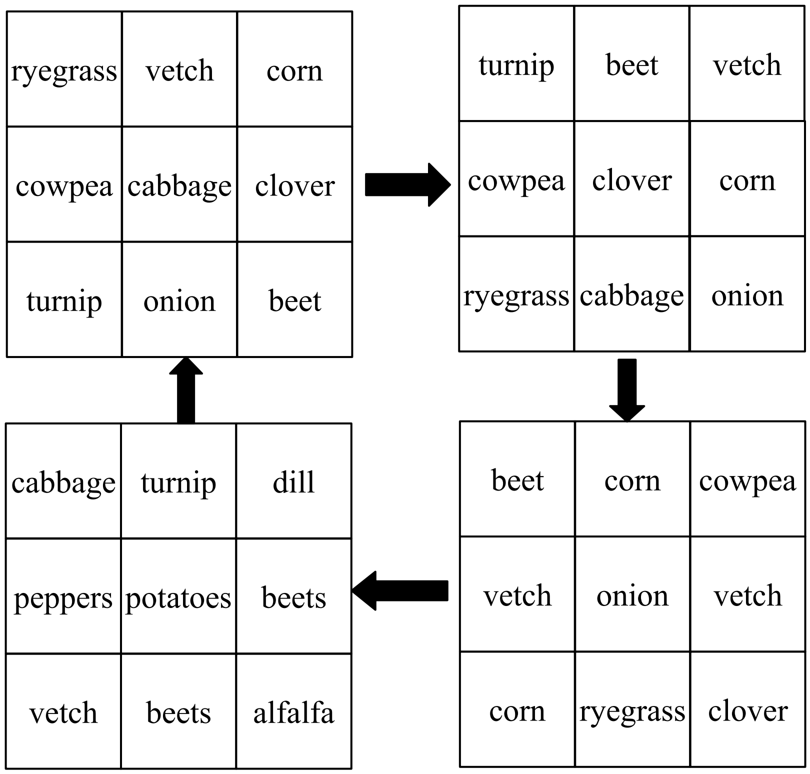
y Herbs are not all from the same family, but are listed together here as a matter of simplicity.

Example Rotation with Bad Outcomes

Plot 1	Plot 2	Plot 3	Plot 4	Plot 5
corn	vetch	beans	corn	eggplant
vetch	beans	rye	rye	vetch
beans	corn	vetch	potatoes	
rye		corn	vetch	
potatoes		potatoes		
eggplant				

Example of a FSA representing crop rotations

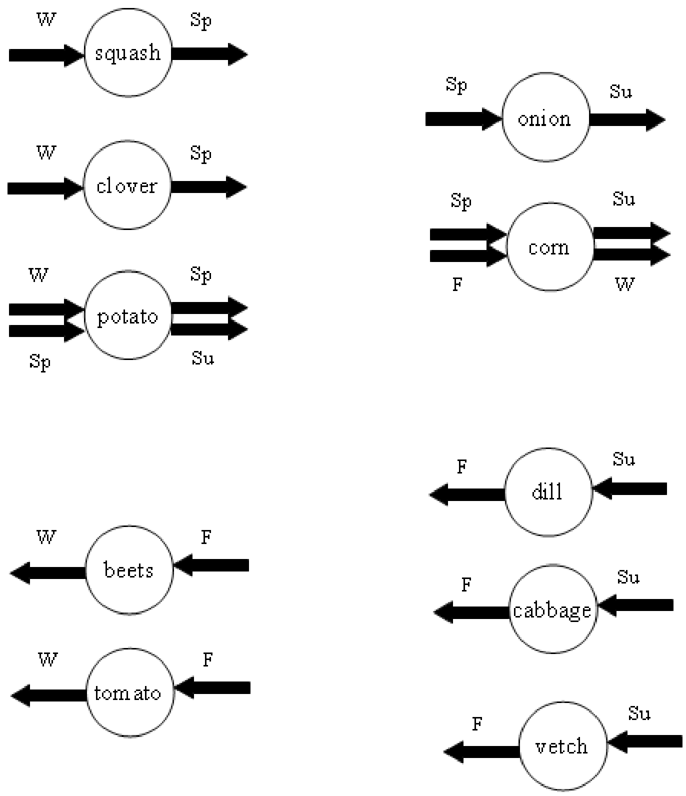
The first example is a fairly rudimentary finite state automaton. The states are, themselves, entire layouts of the field, with all nine plots filled in in a somewhat arbitrary way. This method would be not much more advanced that just filling in a list for each plot arbitrarily. The corresponding list is included below the FSA diagram.



Plot (1,1)	Plot (1,2)	Plot (1,3)	Plot (2,1)	Plot (2,2)	Plot (2,3)	Plot (3,1)	Plot (3,2)	Plot (3,3)
ryegrass	vetch	corn	cowpea	cabbage	clover	turnip	onion	beet
turnip	beet	vetch	cowpea	clover	corn	ryegrass	cabbage	onion
beet	corn	cowpea	vetch	onion	vetch	corn	ryegrass	clover
cabbage	turnip	dill	peppers	potatoes	beets	vetch	beets	alfalfa

Example of a FSM representing crop rotations (p 1)

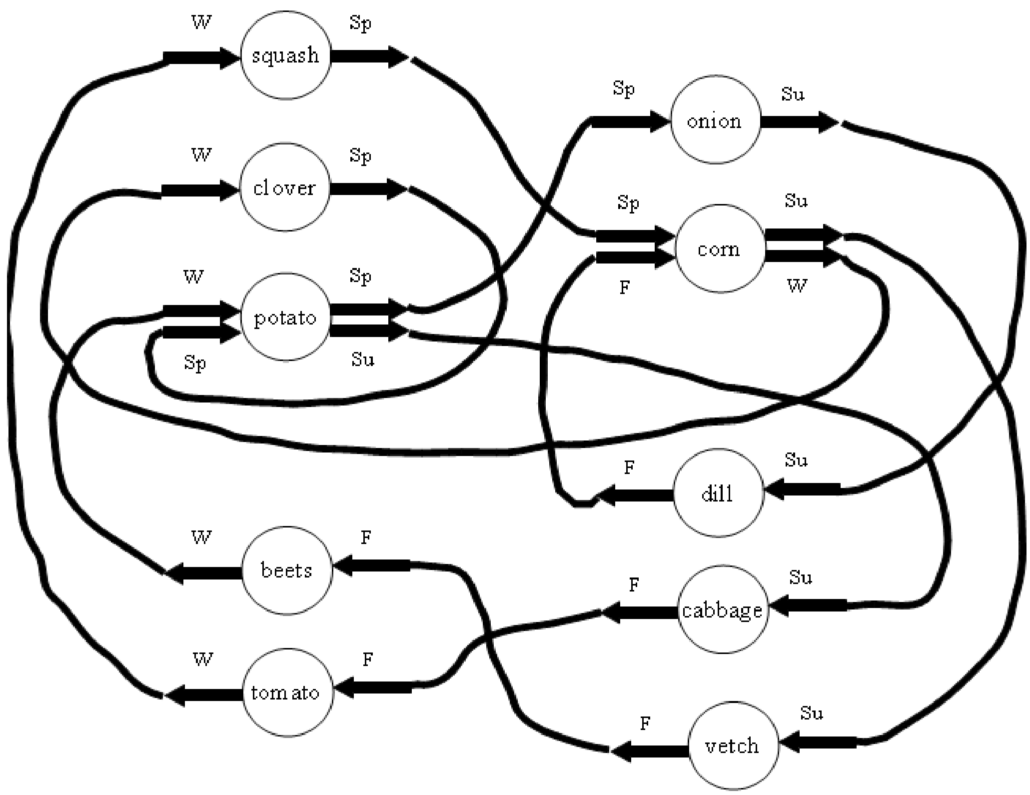
The next example rotation is illustrated in three stages. The first includes crop selection – crops are selected based on some desired output or yield (if you want tomatoes, pick tomatoes). Then, seasons in which to grow the crops are selected (this FSM will use seasons as its input – the input alphabet is the set of seasons, and presumably they are input correctly to correspond to real life). Each season in which the crop is planted will be an incoming arrow, and the subsequent season an outgoing arrow.



Note that these are mostly grouped by season, to allow for the easiest possible connections (in terms of drawing/layout). In a more complex drawing, each crop will have more inputs and outputs, which will lead to more complex rotations.

Example of a FSM representing crop rotations (p 2)

In the next stage, these are connected in ways that are consistent – seasons must match along each arrow, and altogether the rotation should obey the parameters being used. Here, most crops have only one or two outgoing edges, which means that at a specific season, if we are on a crop (say cabbage) and the input is “spring” we have no option. However, because of our design, we know the only input would be “fall” because the previous input must have been “summer” (that is the only incoming edge for cabbage). This is totally fine – we don't need to add extra meaningless edges that will never get used, so long as we know our input will never use them.

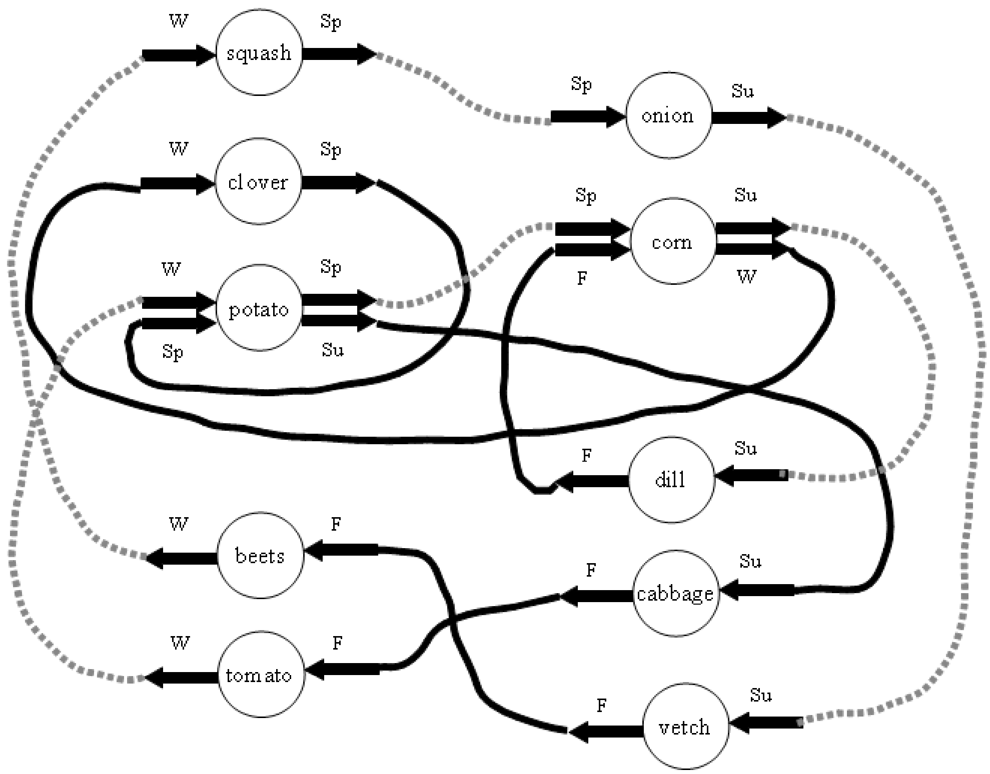


The result of this is a single large cycle of length 12 which is:

squash, corn, vetch, beets, potato, onion, dill, corn, clover, potato, cabbage, tomato

Example of a FSM representing crop rotations (p 3)

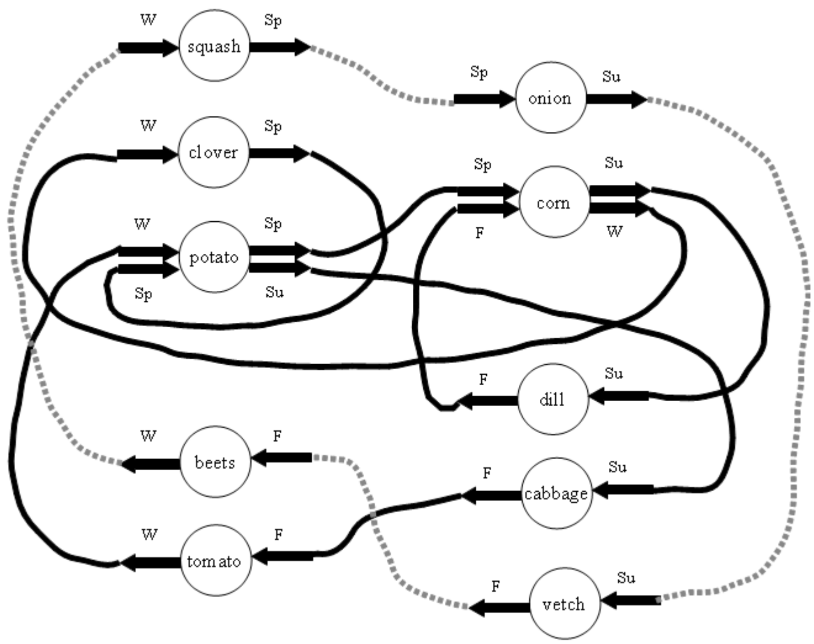
Now, this might not be ideal in some situations or according to our particular agricultural parameters. In particular, potato is repeated twice in too short an interval. There are a few ways to fix this. One is to make another set of rotations:



Note that the arrows that have changed are shaded differently. All other connections are the same as in the previous FSM, but by switching a few connections, the FSM changes drastically.

Example of a FSM representing crop rotations (p 4)

In coming up with this rotation, we notice that there is a cycle of length 4, which will give us something that repeats every year, a very short cycle. The following diagram highlights the two disjoint cycles in the second FSM we have constructed:



Now the two types of arrows show the two disjoint cycles in the FSM (assuming Sp-Su-F-W input).

In fact, in the larger cycle (length 8) we have also created another shorter cycle *in the graph* – dill and corn are both connected to the other. Although the seasons do not dictate that they simply repeat one after the other – instead, this produces (in the graph) a short cycle (length 2) and in the rotation, gives us two corn harvests in a short span of time (for a single plot). So, generally speaking, one might want to avoid short cycles or at least watch for them. The three cycles we have generated are as follows:

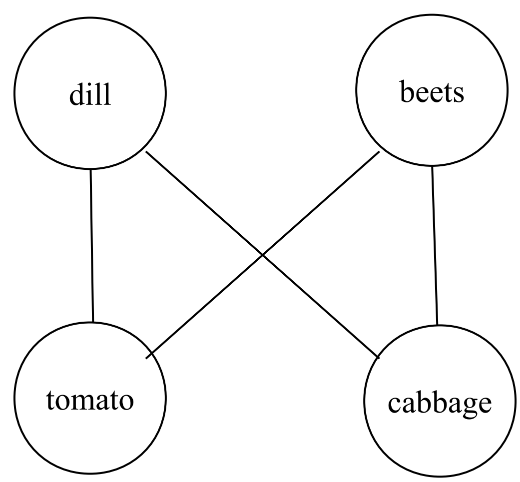
Season	**Cycle 1**	**Cycle 2**	**Cycle 3**
Sp	squash	squash	potato
Su	corn	onion	corn
F	vetch	vetch	dill
W	beets	beets	corn
Sp	potato		clover
Su	onion		potato
F	dill		cabbage
W	corn		tomato
Sp	clover		
Su	potato		
F	cabbage		
W	tomato		

Example of a FSM representing crop rotations (p 5)

We still need to worry about the potatoes – they repeat too often. We can “tweak” the rotation using ad hoc methods. We can simply replace one second occurrence of potato with onion (which goes in the same season). And as for the corn-dill-corn part of the cycle, we can replace dill with vetch in order to bring nitrogen levels up. This sort of ad hoc tweaking is sometimes an important part of this kind of construction, since real world parameters are often too complex to be fully satisfied in a simple mathematical construction. Our final set of possible rotations (changes in bold) is:

Season	**Cycle 1**	**Cycle 2**	**Cycle 3**
Sp	squash	squash	potato
Su	corn	onion	corn
F	vetch	vetch	**vetch**
W	beets	beets	corn
Sp	potato		clover
Su	onion		**onion**
F	dill		cabbage
W	corn		tomato
Sp	clover		
Su	**onion**		
F	cabbage		
W	tomato		

So, these two FSM generates three crop different cycles of crops to use. Multiple such FSM can make many more lists, which can be used one per plot, or one for several plots. In our case we will use these three for all our plots, but in a way that respects reasonable conditions – we will stagger this list *and* make sure to assign adjacency in a way that avoids having neighboring crops that are forbidden – in this case, dill-tomato, dill-cabbage, beets-tomato, and beets-cabbage, which if we'd like to be succinct, can be represented by a graph of forbidden pairings:



Example of a FSM representing crop rotations (p 6)

The three cycles we get from our FSM do not have particular starting points. We can choose arbitrarily, and this allows us to choose several starting points and assign different starting points in the rotation to each plot. One way of doing so gives us this (for a 3x3 plot):

	Plot (1,1)	Plot (1,2)	Plot (1,3)	Plot (2,1)	Plot (2,2)	Plot (2,3)	Plot (3,1)	Plot (3,2)	Plot (3,3)
Sp	squash	clover	potato	clover	potato	clover	squash	squash	squash
Su	onion	onion	corn	onion	corn	onion	onion	corn	onion
F	vetch	cabbage	vetch	cabbage	vetch	cabbage	vetch	vetch	vetch
W	beets	tomato	corn	tomato	corn	tomato	beets	beets	beets
Sp	squash	potato	clover	squash	clover	squash	squash	potato	squash
S	onion	corn	onion	corn	onion	corn	onion	onion	onion
F	vetch	vetch	cabbage	vetch	cabbage	vetch	vetch	dill	vetch
W	beets	corn	tomato	beets	tomato	beets	beets	corn	beets
Sp	squash	clover	potato	potato	potato	potato	squash	clover	squash
S	onion	onion	corn	onion	corn	onion	onion	onion	onion
F	vetch	cabbage	vetch	dill	vetch	dill	vetch	cabbage	vetch
W	beets	tomato	corn	corn	corn	corn	beets	tomato	beets
Sp	squash	potato	clover	clover	clover	clover	squash	squash	squash
Su	onion	corn	onion	onion	onion	onion	onion	corn	onion
F	vetch	vetch	cabbage	cabbage	cabbage	cabbage	vetch	vetch	vetch
W	beets	corn	tomato	tomato	tomato	tomato	beets	beets	beets
Sp	squash	clover	potato	squash	potato	squash	squash	potato	squash
Su	onion	onion	corn	corn	corn	corn	onion	onion	onion
F	vetch	cabbage	vetch	vetch	vetch	vetch	vetch	dill	vetch
W	beets	tomato	corn	beets	corn	beets	beets	corn	beets
Sp	squash	potato	clover	potato	clover	potato	squash	clover	squash
S	onion	corn	onion	onion	onion	onion	onion	onion	onion
F	vetch	vetch	cabbage	dill	cabbage	dill	vetch	cabbage	vetch
W	beets	corn	tomato	corn	tomato	corn	beets	tomato	beets

Note that the length of the table is LCM(4,12,8) = 24. Bad adjacency in bold (see below).

Geometrically, each cycle can be assigned to plots in the field according to this layout:

Cycle 2 (+0)	Cycle 3 (+4)	Cycle 3 (+0)
Cycle 1 (+4)	Cycle 3 (+0)	Cycle 1 (+8)
Cycle 2 (+0)	Cycle 1 (+0)	Cycle 2 (+0)

The parentheses indicate the offset in the cycle.

Examples of block designs

This first example is a Latin square. It is complete, and thus is regular and uniform. Because it is a Latin square, the order of the rows and columns is meaningful.

1	2	3	4	5	6
2	3	4	5	6	1
3	4	5	6	1	2
4	5	6	1	2	3
5	6	1	2	3	4
6	1	2	3	4	5

The next example is a complete design that is not a Latin square (it is not even square). Its parameters are v=5, k=5, r=7, b=7. The order of the blocks, and of elements within blocks, is irrelevant.

1	2	3	4	5
1	2	3	4	5
1	2	3	4	5
1	2	3	4	5
1	2	3	4	5
1	2	3	4	5
1	2	3	4	5

This example is a regular (thus uniform) block design that is more interesting than the previous example of a complete block design. Its parameters are v=6, k=4, r=2, b=3. Notice v r = b k, which must be true of a block design that is regular.

1	2	4	5
1	3	5	6
2	3	4	6

The final two examples have v=4 and b=3. In one case (left), the design is uniform with k=3. On the right, the design would be regular (with r=2) except that it is not uniform (one element is "missing"). Note that if v=4, b=3, and k=3 in a uniform design, there is no way to make it regular because if so, the above-mentioned formula gives r = b k / v = 3*3 / 4, which is 9/4 (not an integer).

1	2	3	1	2	3
2	3	4	2	3	4
1	2	4	1	4	

Example Crop Rotation with Block Designs (p1)

Fill in the gaps, marked [_______?], *with the best possible answer.*

The rotation is based on the following block design:

4	5	6	7	8	9
1	2	3	7	8	9
1	2	3	4	5	6
2	3	5	6	8	9
1	3	4	6	7	9
1	2	4	5	7	8
2	3	4	5	7	9
1	3	5	6	7	8
1	2	_?	6	8	9
2	3	4	_?	7	8
1	3	4	5	8	9
1	2	5	6	7	9

This design is [_______?], and so it has parameters

v = [_____?]   b = [_____?]   k = [_____?]   r = [_____?]

Because each rotation should have at least one nitrogen fixer, our first priority might be to use vetch and [________?] to act as nitrogen producers in each block, and thus we assign:

5 = vetch

7 = [_______?]

Notice that it is perfectly acceptable to assign one crop to more than one number (so both of these could be vetch). Other assignments can be made, taking care not to give any block too many nitrogen-depleting crops and to plant a sufficient number of cash crops (assuming some goal of having high yields of cash crops).

Example Crop Rotation with Block Designs (p2)

Fill this in using your own choices for the rest of this table:

4	vetch	6	[_____?]	8	9
1	2	3	[_____?]	8	9
1	2	3	4	vetch	6
2	3	vetch	6	8	9
1	3	4	6	[_____?]	9
1	2	4	vetch	[_____?]	8
2	3	4	vetch	[_____?]	9
1	3	vetch	6	[_____?]	8
1	2	_?	6	8	9
2	3	4	_?	[_____?]	8
1	3	4	vetch	8	9
1	2	vetch	6	[_____?]	9

Find a way of rotating these crops in order to compensate for nitrogen depletion, forbidden adjacency, etc. Here is where block-designs fall short and ad hoc methods (or methods of FSM) can be used. Simulate and analyze the results of the rotation you devise.