

Hexagonal Grids for Choice Experiments

WILLIAM H. WILSON, DEBORAH J. STREET and D. A. MAELZER

ABSTRACT

In this note we construct designs on the hexagonal grid to be used for choice experiments.

1. Introduction.

Hexagonal grids have been used to investigate competition between two varieties of plants (Martin (1973), Veevers and Boffey (1975,1979), Boffey and Veevers (1979)), to examine the effects of crowding on one variety (Veevers (1982)) and in plant breeding work (Fasoulas (1981)) (see Street and Street (1987, Chapter 15) for a summary of this and related work). In this paper we consider the use of hexagonal grids for choice experiments.

Entomologists conduct a variety of 'choice' experiments to determine what stimuli insects respond to when exhibiting certain behaviour, for example when searching for food. In some such experiments, each treatment is applied to a particular area of a 'choice chamber', often a 'porthole' at the end of a small corridor. Insects are then introduced singly to the choice chamber and each insect is observed as it moves towards (responds to) one or other of the treatment areas (Vet, van Lenteren, Heymans and Meelis (1983)). In other choice experiments, insects are given a choice between different sorts of plants - usually in randomized blocks - to measure ovipositional preferences (Ahmad (1983)), or plant resistance to insects (Maxwell and Jennings (1980)), or the searching efficiency of parasitoids (Waage (1983)).

We were concerned to plan an experiment for snails to test the hypothesis that each of 6 varieties (cultivars) of roses was equally acceptable to climbing snails (Symon and Maelzer (1987)). We rejected a randomized block design because we thought it was critical that stems of each of the 6 cultivars be equidistant from an initially seeded snail so that the snail had an equal chance of finding and climbing up the stem of any of the 6 cultivars. Thus, initially, each snail lies at the centre of a hexagon of rose stems. We also decided that it would be desirable, because of a scarcity of rose stems, to allow the rose stems to be 'shared'. We did this by laying the rose stems out in an extended hexagonal grid. An example appears in Figure 1. The cultivars are represented by the letters A,B,C,D,E and F. In the experimental layout the cultivars are placed at the centre of the hexagonal region indicated. Thus the hexagon of rose stems surrounding snail 1 of Figure 1 is A-E-C-F-B-D.

The idea of the design was to have one stem of each variety neighbouring each snail. We balanced the design for adjacencies, as 'seen' by the snails, of varieties of roses. Thus, in the design of Figure 1, any of the distinct pairs of letters appear in adjacent hexagons ten times. For instance, A and B appear in adjacent hexagons for snails 8, 9, 10, 18, 19, 20, 22, 23, 24 and 25. We also ensured that each variety appeared five times in the border cells. In addition, to reduce the size of the problem, we made the right half of the array the complement of the left half, where by complement we mean that the mirror image of a cell contains the complementary label

with respect to the pairing $A \leftrightarrow F$, $B \leftrightarrow E$, $C \leftrightarrow D$. The 'mirror axis' is shown in Figure 1. This design was found by a computer search procedure described below.

Other experiments in which designs balanced for 'cultivar adjacency' are appropriate include those in which a set of cultivars are 'screened' for resistance to insects. This is often done by placing the same number of insects on each cultivar and recording, at some later time, the number of insects on each cultivar. Since the final numbers recorded are usually dependent on some insects walking off some cultivars and accumulating on other cultivars, each cultivar should have an equal chance of accumulating insects from the other cultivars. Such experiments will require designs similar to the ones in this paper but for values other than 6.

We will present the results of the experiments and discuss the analysis of these designs in a subsequent paper.

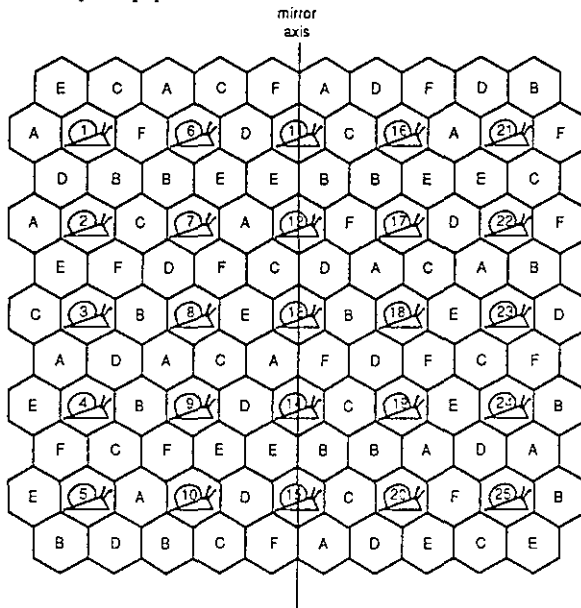


Figure 1: Snails and roses on a hexagonal grid.
The treatment labels A-F indicate the varieties of rose stem.
The snails are numbered 1-25.

2. The Search Procedure

The search program used a backtrack procedure which attempted to place each of the six treatment labels, in turn, into each of the 45 treatment cells to the left of the mirror axis. At the same time, the program would place the complementary label in the mirror image treatment cell. If this could be done without violating any of the constraints listed below, then the backtrack procedure would call itself (i.e. recurse) in order to try to fill the next treatment cell. If no label could legally be placed in a particular treatment cell, then the procedure would backtrack as far as necessary to find a legal continuation of the filling procedure. The sequence in which the treatment cells were filled was chosen so as to try to promote pruning of the search tree as early as possible. This filling sequence is shown in Figure 2. The principle used in deciding the filling sequence was first to fill six cells surrounding some snail with the

six treatment labels A-F (clearly no generality is lost by doing this) and then to fill cells adjacent to the mirror axis, with the aim of obtaining search space cutoffs by rapidly filling all of the treatment cells neighbouring the five snails lying on the mirror axis. The sequence continues in this vein, completing the neighbours of snails 9, 8, and 7, then snails 2, 3, 4, 5, 1 and finally snail 6.

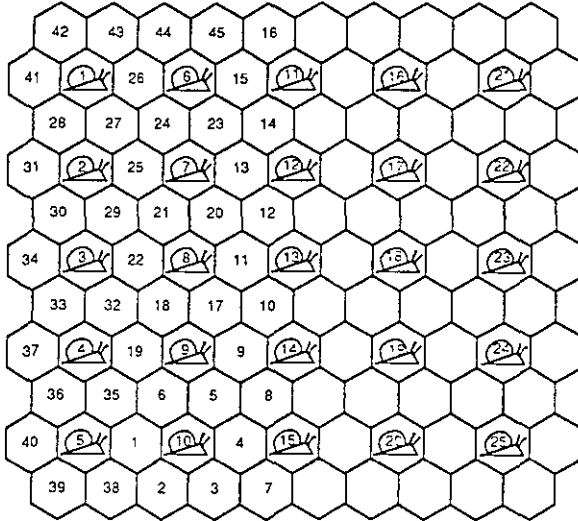


Figure 2: The filling sequence used by the search algorithm

The constraints which were checked before allowing a treatment label to be placed in a cell were as follows:

- There should be at most one instance of any treatment label adjacent to any particular snail.
- There should be at most 5 instances of any particular treatment label in a border cell;
- There should be at most 10 instances of any particular treatment label in an interior cell;
- Each snail 'sees' 6 pairs of adjacent treatments. For example, in Figure 1, snail 7 sees the pairs {C,B}, {B,E}, {E,A}, {A,F}, {F,D}, and {D,C} (clockwise, starting from the left). We wish to balance the design for these adjacencies. Since there are 25 snails, there are 150 neighbouring pairs altogether. There are 15 possible unordered pairs of labels. So it would be desirable for each of the possible adjacent pairs to occur $150/15 = 10$ times on the grid. During the search, we check that each pair does not occur more than 10 times.

Following is an outline of the backtrack procedure central to the search program, written in a Pascal-like pseudocode:

```

const
    MaxInteriorRep = 10;
    MaxBorderRep = 5;

type Labels = (A,B,C,D,E,F);
    
```

```

pseudoprocedure TryToFill(CellNumber: integer);
var TreatmentLabel: Labels;
begin
  for TreatmentLabel := A to F do
    {Try putting this TreatmentLabel in cell number CellNumber}
    Check that no snail adjacent to this cell already has
      TreatmentLabel in an adjacent cell
    Check that TreatmentLabel has not already been used
      MaxInteriorRep times (interior cell) or MaxBorderRep times
      (border cell)
    Check neighbour balance would be OK, and if so, update it
    if all OK then
      begin
        Place this TreatmentLabel in the cell
        Place complementary TreatmentLabel in mirror image cell
        Update balance/replication information
        if all cells now filled
          then print the solution we've just found
          else TryToFill(CellNumber + 1)
        {When we reach this point, we have either failed to find any
        solution in this branch of the search tree, or else we've
        found one or more solutions, but have now exhausted the
        possibilities in this part of the tree. The next thing to do
        is to try for a solution with some other treatment in the
        current cell, or else to backtrack if we've now tried all
        the treatments.}
        Restore balance/replication information specific to this
        treatment label
        Remove treatment label from the cell and from the mirror
        image cell
      end
    end;
  end;
end; {TryToFill}

```

Cells 1 to 6 can be filled with treatment labels A to F without loss of generality, as we mentioned before. The central task of main program, then, is to set up the balance and replication information for this filling of cells 1 to 6, and then make the call TryToFill(7). The number of designs found which satisfy the criteria above was 604,506.

3. Designs with Additional Properties

As there were over 600,000 designs, even with mirror symmetry and the balance properties, we decided to filter the designs found by the search program to see whether there were any designs with the following additional properties. The first additional property relates to the situation where a snail ignores all of the immediately neighbouring rose stems, and instead prefers one of the stems next furthest away. We refer to these stems as the distance 2 neighbours of a snail. Figure 3(a) shows the distance 2 neighbours of snail 13. Not all snails will have as many as four neighbours at distance 2: for example, snail 1 has only two (to its right). The extra property that we sought was that designs should not have repeated treatments at distance 2 from any snail. Since there are at most four neighbours at distance 2, we cannot ask that all treatment labels be represented at distance 2.

The second additional property relates to the distance 3 neighbours of a snail. Figure 3(b) illustrates this concept. As with neighbours at distance 2, some snails have two neighbours at distance 3, and some have four such neighbours. Again, we made the requirement that there be no repeated treatment labels among the distance 3 neighbours of any snail. Table 1 gives the number of designs found by the basic search process, and by the two filtering processes which checked for the additional properties. In fact, the design in Figure 1 has no repeated neighbours at either distance 2 or distance 3.

The complete listing of the 605 designs with no repeated neighbours at distances 2 or 3 is available from the authors.

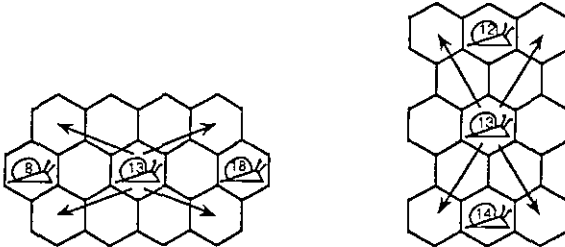


Figure 3

(a) The distance 2 neighbours of snail 13 (b) The distance 3 neighbours of snail 13

Type of Design	Number of designs found
Basic snail design	604,406
No repeated neighbours at distance 2	5,637
No repeated neighbours at distances 2 or 3	605

Table 1: Numbers of designs found

REFERENCES

- Ahmad, S. (ed.) (1983). *Herbivorous Insects: Host-seeking Behaviour and Mechanisms*. Academic Press.
- Boffey, T.B. and Veevers, A. (1979). On the non-existence of balanced designs for two-variety competition experiments. *Utilitas Mathematica* 16, 131-43.
- Fasoulas, A. (1981). *Principles and methods of plant breeding*. Publication No. 11, Department of Genetics and Plant Breeding, Aristotelian University of Thessaloniki, Greece.
- Martin, Frank B. (1973). Beehive designs for observing variety competition. *Biometrics* 29, 397-402.
- Maxwell, F.G. and Jennings, P.R. (1980). *Breeding Plants Resistant to Insects*. John Wiley.
- Street, Anne Penfold and Street, Deborah J. (1987). *Combinatorics of Experimental Design*. Oxford Science Publications.
- Symon, D.E. and Maelzer, D.A. (1987). Personal communication.
- Veevers, Alan (1981). Balanced designs for observing intra-variety nearest-neighbour interactions. *Euphytica* 31, 465-8.
- Veevers, Alan and Boffey, T.B. (1975). On the existence of levelled beehive designs. *Biometrics* 31, 963-7.
- Veevers, Alan and Boffey, T.B. (1979). Designs for balanced observation of plant competition. *Journal of Statistical Planning and Inference* 3, 325-31.
- Vet, L.E.M., van Lenteren, J.C., Heymans, M. and Meelis, E. (1983). An airflow olfactometer for measuring olfactory responses of hymenopterous parasitoids and other small insects. *Physiological Entomology* 8, 97-106.
- Waage, J. (1983). Aggregation in field parasitoid populations: foraging time allocation by a population of *Diadegma* (Hymenoptera; Ichneumonidae). *Ecological Entomology* 8, 447-53.

William H. Wilson,
 Discipline of Computer Science,
 Flinders University of South Australia,
 Bedford Park, SA 5042
 AUSTRALIA

Deborah J Street, D. A. Maelzer,
 Waite Agricultural Research Institute,
 The University of Adelaide,
 Glen Osmond, SA 5064
 AUSTRALIA