Journal of Statistical Software

February 2005, Volume 13, Issue 3. http://www.jstatsoft.org/

Visualizing Experimental Designs for Balanced ANOVA Models using Lisp-Stat

Philip W. Iversen
Eli Lilly and Company

Mervyn G. Marasinghe
Iowa State University

Abstract

The structure, or Hasse, diagram described by Taylor and Hilton (1981, American Statistician) provides a visual display of the relationships between factors for balanced complete experimental designs. Using the Hasse diagram, rules exist for determining the appropriate linear model, ANOVA table, expected means squares, and F-tests in the case of balanced designs. This procedure has been implemented in Lisp-Stat using a software representation of the experimental design. The user can interact with the Hasse diagram to add, change, or delete factors and see the effect on the proposed analysis. The system has potential uses in teaching and consulting.

Keywords: Lisp-Stat, Hasse diagram, experimental design.

1. Introduction

Hasse diagrams have been described previously for their usefulness in visualizing experimental designs and determining the correct model for orthogonal (or balanced), mixed effects designs (Taylor and Hilton 1981; Lohr 1995). They show the nesting relationships, as well as other information, about the factors in an experiment. This paper describes a Lisp-Stat system for visualizing experimental designs using the Hasse diagram. Given information about the factors in a design, the system will produce:

- A Hasse diagram of the nesting relationships among the factors.
- The terms in an appropriate linear model.
- An outline of the ANOVA table with appropriate F-tests.
- Formulas for the expected mean square of each term in the model.

The system uses heuristic rules presented in Taylor and Hilton (1981) and the object-oriented and dynamic graphics programming features available in Lisp-Stat (Tierney 1990). The user is able to interact with the diagram to change factors and their settings and to see the effects of these changes on the model and the analysis. Section two describes response structures and their relationship with Hasse diagrams, also called structure diagrams. Section three summarizes factor sets, which are used to construct the expected mean squares, appropriate F tests, and the ANOVA table. Section four describes our system and the underlying software representation. We illustrate the system with several examples in section five.

2. Response structures and Hasse diagrams

Responses that are outcomes of a designed experiment can be classified by **factors** of the experiment. The factors may consist of experimental treatments, or may correspond to various spatial or temporal arrangements of the experimental units, such as plots and blocks. **Levels** of factors partition the set of responses into disjoint, nonempty subsets. Upper case letters denote factors and subscripts denote levels of factors. For example, let $X = \{A,\ B,\ C\}$ be a set of factors of interest in an experiment. Then $A_1,\ B_3,\ C_j$ denote various levels of each of the factors. A combination of a set of factors is a set of responses formed by the intersection of individual levels of each factor. For example, $x_{ijk} = A_i \cap B_j \cap C_k$ is a combination of X. The set of responses and a corresponding set of factors, X, are said to form a **response structure**. A response structure in which every possible combination of the set of factors is nonempty and all combinations of every subset of factors contain the same number of responses, is called a **balanced complete response structure**.

We use the following definition of nesting. Given two factors A and B of a response structure, if the partition of responses generated by B is a refinement of the partition given by A, then factor B is said to be **nested** in A. More formally, factor B is said to be nested in A if each level of B appears with one, and only one, level of A. Throckmorton (1961) shows that, by considering the **nesting** relationship among the factors as a partial ordering relation on the set of responses, a balanced complete response structure can be represented by a lattice. A lattice is a partially ordered set in which every pair of elements has a least upper bound and a greatest lower bound. It can be shown that the set of factors of a response structure and the nesting relationship satisfy this definition.

A lattice structure can be represented visually by a Hasse diagram, in which a node that covers other nodes is placed immediately above those nodes in the diagram and is connected to those nodes with line segments. Hasse diagrams are discussed extensively by Throckmorton (1961) and Kempthorne (1982). Thus balanced complete response structures can be represented by Hasse diagrams in which the nodes correspond to the factors and line segments denote nesting relationships.

In the Hasse diagram of Figure 1, **u** represents the grand mean and **E** represents the error term. All factors are nested in **u**, and **E** is nested in all other factors. In addition, **C** is nested in **A**. If two factors are not connected by an upward link, they are **crossed**. Thus, **B** is crossed with **A** and **C**. The number in parentheses after each factor is the **range**, or number of levels. For a nested factor, say **C**, this number indicates that **C** has three levels for *each* level of **A**. Finally, fixed factors are underlined while random factors are not.

The set of factors of the original response structure augmented by the interactions of crossed

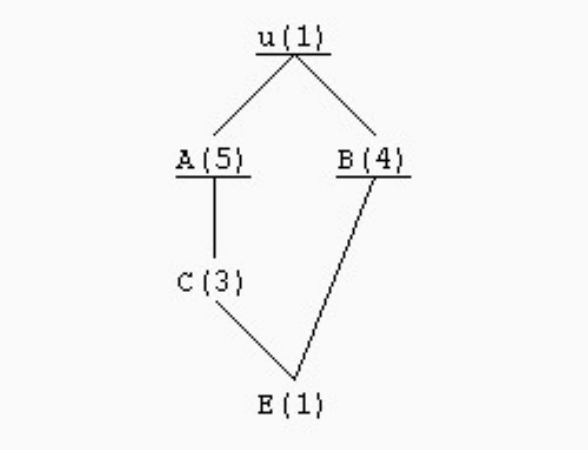

Figure 1: A Hasse diagram.

factors from the original response structure and **u**, also form a lattice, which is useful for determination of the corresponding linear model and associated computations as discussed in Taylor and Hilton (1981). The augmented lattice for the above example is given in Figure 2. From Figure 2, it can be ascertained that the interaction AB is nested in **u**, and that the interaction BC is nested in AB. Carney (1967) established that each term of the augmented lattice identify "components" which correspond to "sources of variability" in an ANOVA table or "effects" in a linear model. Using conventional labelling, the responses in the response structure discussed above are y_{ijk} with i, j, k denoting levels of factors A, B, and C respectively. The linear model corresponding to the above structure is thus

$$y_{ijk} \quad = \quad \mu + A_i + B_j + AB_{ij} + C_{k(i)} + \epsilon_{jk(i)},$$

where, for simplicity, the factor names are used to denote the effects, except for the BC interaction, which is the error term, $\epsilon_{jk(i)}$. Note that parenthesized subscripts denote levels of factors that next the factor which corresponds to the effect name. Thus the effect $C_{k(i)}$ should be read as "C nested in A."

3. Factor sets

Symbolic Factors (SF)	Live Factors: factors that appear in the effect name (LF).	
	Dead Factors: factors that nest the live factors (DF).	
Complement Factors (CF)	Random Complement Factors (RCF)	Simple RCF: not nested in any other RCFs (SRCF).
		Non-simple RCF (NRCF).
	Fixed Complement Factors (FCF).	

Table 1: Factor Sets. The partition becomes finer as one moves from left to right.

Factors sets are defined in terms of the nesting relationships and factor types (fixed or random). They play a central role in determing the ANOVA table and expected mean squares,

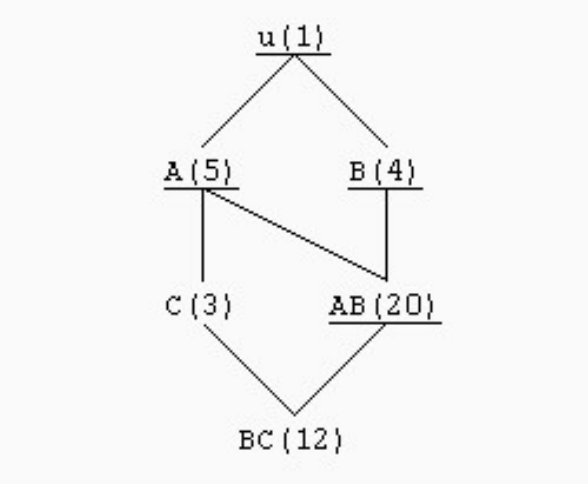

Figure 2: An augmented Hasse diagram.

and are easily obtained with reference to the Hasse diagram. The set of all factors in an experimental design is called the design set (DS). An effect is any single factor or any combination of 2 or more crossed factors. Every effect defines a different partition of the design set. The breakdown of the design set into factor sets is shown in table 1. Note the following relationships among the factor sets:

$$DS = SF \cup CF$$
$$SF = LF \cup DF$$
$$CF = RCF \cup FCF$$
$$RCF = SRCF \cup NRCF$$

Two numerical quantities associated with each effect and required for the ANOVA table are obtained from the factor sets. First we define the **range** of a factor as its number of levels and the **diminished range** as the range minus one. The **symbolic product** of an effect is computed as the product of the diminished ranges of its live factors and the ranges of its dead factors. The symbolic product generates the degrees of freedom for the given effect. The **complement product** of an effect is the product of the diminished ranges of its complement factors and generates the expected mean square coefficient for the given effect.

The final ingredient is the notion of formal interaction. The formal interaction of two effects is found by first juxtaposing all factors in the effects and then removing duplicates and factors that nest any other factors. The result is always an allowable effect. The next section describes our Lisp-Stat system.

4. Lisp-Stat implementation

The Lisp-Stat environment provides an excellent platform for object-oriented programming and customized dynamic graphics. The symbolic processing power of LISP is also available. We make use of these features to manipulate factor sets and to allow the user to interact with the Hasse diagram.

Object-oriented programming involves the use of computing entities, called **objects**, that contain both data and the procedures for processing that data. The data for each object are stored in **slots**. Procecures are called by sending a **message** to an object to execute one of its **methods**. For example, `reg-model` is a Lisp-Stat regression-model object, and we could ask it to plot its residuals with the expression

```
(send reg-model :plot-residuals)
```

In the Lisp-Stat object system, there exist certain objects called **prototypes**. A prototype object contains the blueprint for creating new objects, called **instances**. Each instance has its own **slots** to hold data, but looks to its prototype for the code to execute methods. However, an instance may also have its own methods that either add to, or override, the prototype methods. Thus, code can be shared by many objects. In addition, prototypes may be arranged in an **inheritance hierarchy**, that allows prototypes to inherit, i.e. reuse, code from other prototypes. For more information on Lisp-Stat and its object system, see Tierney (1990).

Our system defines three prototype objects: `balanced-anova-proto`, `factor-proto`, and `Hasse-diagram-proto`. The first two prototypes inherit from the Lisp-Stat root object `*object*`, and `Hasse-diagram-proto` inherits from the Lisp-Stat prototype `graph-window-proto`. These are described in detail below, following the system overview.

4.1. System overview

Balanced experimental designs are implemented using the prototype, `balanced-anova-proto`. It contains slots to hold data about a design and methods to process that data. The user begins by calling the function `balanced-anova-model`, for example,

```
(def ba (balanced-anova-model))
```

which returns a new instance of `balanced-anova-proto` to the variable `ba`.

This function also prompts the user to select a design file, which is a plain text file that described the factors in a design. Or, a default design file can be loaded automatically. The design file for the experiment shown in Figure 1 is:

```
(A 5 fixed nil)
(C 3 random (A))
(B 4 fixed nil)
(E 1 random (A C B))
```

There is one line for each factor in the design. Each factor is represented as a list of four elements: the factor name (or letter designation), number of levels, type (either fixed or random) and a list of all factors in which the given factor is nested. An empty list, `nil`, in the fourth position indicates that the given factor is not nested in any other factors. Characters may be entered in upper- or lowercase; however, Lisp-Stat converts them to uppercase. The error term factor is required, and it must be nested in all of the other factors.

A new instance of `factor-proto` is created for each factor, and these objects are stored as a list in the `balanced-anova` object. Next, a list of all allowable effects in the model is computed

and saved. The Hasse diagram is displayed, and a menu is installed that allows access to the Hasse diagram, ANOVA table, expected mean square formulas, and factor sets. The Hasse diagram has two modes of interactivity. First, the user can re-position nodes horizontally to improve the visual appearance of the diagram. Second, the user can change a factor's settings by shift-clicking on a node. A dialog window shows the current settings (name, range, type, and nested-in factors) and allows changes. The Hasse diagram is automatically updated when this information is changed and the user clicks the OK button. A factor can be deleted by checking the Delete button in the dialog window. A factor can be added to the design by clicking on the white space between nodes. The user may also load another design from the menu to replace the current design.

4.2. Balanced ANOVA prototype - Slots and primary methods

Balanced ANOVA objects have the following slots:

- factors - list of factor objects
- effects - list of allowable effects
- menu - menu object
- factor-names - list of factor names
- hasse-diagram - hasse diagram object

The following `balanced-anova-proto` methods correspond to items in the menu. Method names always begin with a colon (:) and are followed by an argument list, where `()` means that no arguments are required.

- `:load-design ()` loads a design file. This method is run automatically by the function, `balanced-anova-model`.
- `:draw-hasse ()` draws the Hasse diagram. Each node in the diagram corresponds to a factor in the design. The vertical positioning of the nodes is determined by the nesting relationships. The user is able to rearrange the nodes by dragging them horizontally. One special node is added to complete the diagram: the grand mean which has one level, is fixed, and nests all other factors. Its nesting relationship is added automatically and should not be included in the design file.
- `:augmented-hasse` draws the augmented Hasse diagram, which has a node for every allowable effect.
- `:linear-model ()` writes the list of effects as a linear model.
- `:show-table ()` displays the ANOVA table, which has columns for source of variation, degrees of freedom, and mean square values and F-tests. Each allowable effect produces a line in the ANOVA. The value for the degrees of freedom comes from the symbolic

product, and the F-test formula is derived via the following procedure. Form two sets of effects, S_e and S_o, defined as

$$\begin{aligned} S_e &= \{e|e \text{ is a formal interaction of } LF \text{ with an} \\ &\quad \text{even-way } (0, 2, 4, \ldots) \text{ interaction of} \\ &\quad \text{factors in } SRCF\} \\ S_o &= \{o|o \text{ is a formal interaction of } LF \text{ with an} \\ &\quad \text{odd-way } (1, 3, 5, \ldots) \text{ interaction of} \\ &\quad \text{factors in } SRCF\} \end{aligned}$$

All n-way, $n = (0, 1, 2, \ldots)$, formal interactions of factors in $SRCF$ are computed at once using the `:allowable-effects` method, which takes the set, $SRCF$, and returns the list of effects, $S_e \cup S_o$. Then the even-way and odd-way interactions are extracted depending on the number of factors in the effect (even or odd). Finally, the numerator of the F-test is the sum of the mean squares of the effects in S_e, and the denominator is similarly obtained from S_o.

The sets, S_e and S_o always have the same size, say m. If $m = 1$, the F-test is exact; otherwise, the ratio represents an approximate Satterthwaite test.

- `:ems-list ()` displays formulas for the expected mean square of each effect in the model. Expected mean squares are computed by first forming the set

$$\begin{aligned} S &= \{s|s \text{ is a formal interaction of } LF \text{ with an} \\ &\quad m\text{-way interaction of factors in } RCF, \\ &\quad m = (0, 1, 2, \ldots)\} \end{aligned}$$

This set contains the effects that appear in the expected mean square with coefficients given by the complement product. Thus for a given effect, Q,

$$EMS(Q) = \sum_{s \in S} k(s)\sigma_s^2$$

where $k(s)$ is the complement product of the effect, s, and σ_s^2 is a variance component if s is random, or a mean squared deviation from the treatment mean if s is fixed. An effect is random if any one of its live factors, LF, is random.

- `:factor-sets ()` displays the important factor sets (SF, LF, DF, CF, RCF, and $SRCF$) for a user-selected subset of effects.

4.3. Balanced ANOVA prototype - Other methods

The following `balanced-anova-proto` methods are used by the methods described above and are available for investigating the balanced-anova object.

- `:factors (&optional factors)` returns the list of factors stored in the `factors` slot. If an argument is supplied, it will become the new slot value. The argument should be a list whose items are instances of `factor-proto` (see section 4.4).

- `:effects (&optional effects)` returns the list of effects stored in the `effects` slot. If an argument is supplied, it will become the new slot value. This slot can be reset with the following expression:

```
(send ba :effects
      (send ba :allowable-effects
            (send ba :factors)))
```

- `:allowable-effects (factor-list)` computes all allowable effects that can be derived from the list of factors and their nesting relationships. An effect is allowable if no factor in the effect is nested in another factor in the effect, i.e., if all of the factors in the effect are crossed with each other.

- `:complement-factors (effect)` returns the set of complement factors for `effect`.

- `:complement-prod (effect)` returns the complement product of `effect`.

- `:dead-factors (effect)` returns the set of dead factors for `effect`.

- `:effect-string (effect &key extended)` returns a string that writes the effect in the usual notation, e.g. B*C. if `:extended` is `t` then the factors in which the effect is nested are included in parentheses, e.g. B*C(A).

- `:F-test (this-effect)` displays the F-test formula for `this-effect`.

- `:gfi (effect-list effect)` computes the generalized formal interaction of the `effect` with each item in effect-list and returns a list of the results. This method is used when computing the F-test and expected mean square formulas.

- `:MS-string (effect &key expected maxlen)` writes the effect name as a mean square name or expected mean square name (if `:expected` is `t`), e.g. MS-B*C or EMS-B*C. the argument `maxlen` can be used to control the width of the printed text.

- `:random-CF (effect)` returns the set of random complement factors for the `effect`.

- `:simple-RCF (effect)` returns the set of simple random complement factors for the `effect`.

- `:symbolic-factors (effect)` returns the set of symbolic factors for the `effect`.

- `:symbolic-product (effect)` returns the symbolic product of the `effect`.

- `:write-ems (this-effect)` displays the expected mean square formula for `this-effect`, in which a variance component is denoted by the effect name in parentheses, and a mean squared deviation from a treatment mean is denoted by the effect name in square brackets. For the example shown in Figure 1,

```
EMS-B = 15*[B] + 1*(C*B)
```

since `B` is fixed, but `C*B` and `E` are random effects.

4.4. Factor prototype

The prototype, `factor-proto`, is used to create factor objects. Each factor object contains the following slots, which can be accessed and changed using a method that matches the name of the slot, e.g., `:name`.

- `name` - factor name, or letter designation.
- `range` - the number of levels in the factor.
- `type` - factor type, either `'FIXED` or `'RANDOM`.
- `nested-in` - list of factor objects in which the factor is nested.
- `parents` - list of factor objects that are direct parents of the factor. See `Hasse--diagram-proto` for more details.
- `x` - x-coordinate of the factor in the Hasse diagram. Used by `Hasse-diagram-proto`.
- `y` - y-coordinate of the factor in the Hasse diagram. Used by `Hasse-diagram-proto`.
- `depth` - the depth, or level, of the factor in the Hasse diagram. The grand mean is level zero, so the total number of levels is the maximum depth plus 1. This is used by `Hasse-diagram-proto` to draw the Hasse diagram.

The only other method for `factor-proto` is `:print` which overrides the default method in the Lisp-Stat object, `*object*`. For example, a factor named, A, would be printed as, `#<factor A>`.

4.5. Hasse diagram prototype

The `hasse-diagram` prototype contains the methods for drawing and updating the Hasse diagram. When the `:draw-hasse` message is sent to a balanced-anova object, the following actions take place:

- A new hasse-diagram instance is created. It receives the list of factors from the balanced-anova object and stores it in the slot, `nodes`.
- A root node is added to represent the grand mean. All other nodes are nested in this node.
- For each factor, the direct parents are computed and stored in the `parents` slot of the factor.
- For each factor, the depth is computed and stored in the `depth` slot of the factor. The grand mean has depth zero; its children have depth one, and so on.
- An initial location, (x, y), is computed for each factor and stored in the `x` and `y` slots of the factor.
- The diagram is displayed and saved in the `hasse-diagram` slot of the balanced-anova object.

5. Examples

5.1. Example 1:

The first example is a two-way ANOVA with fixed factors, A and B, which are crossed. The input file is specified as follows:

```
(A 5 fixed nil)
(B 4 fixed nil)
(E 2 random (A B))
```

The corresponding Hasse diagram is shown in Figure 3.

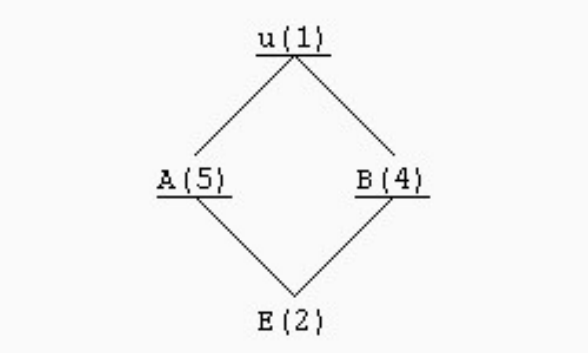

Figure 3: Hasse diagram for Example 1.

The output includes the model, the ANOVA table with appropriate F-tests, and the expected mean squares. Note that there are two levels for the error term, indicating two independent experimental units for each combination of A and B.

```
Model: Y = u + A + B + A*B + E(A,B)

    ANOVA TABLE:

SOURCE            d.f.  Mean Sqr  F-test
-------------     ----  --------  -----------
A                    4  MS-A       MS-A / MS-E.
B                    3  MS-B       MS-B / MS-E.
A*B                 12  MS-A*B     MS-A*B / MS-E.
E(A,B)              20  MS-E       ** / **.
-------------     ----
Total (corr.)       39

Expected Mean Squares:

EMS-A = 8*[A] + 1*(E)
EMS-B = 10*[B] + 1*(E)
```

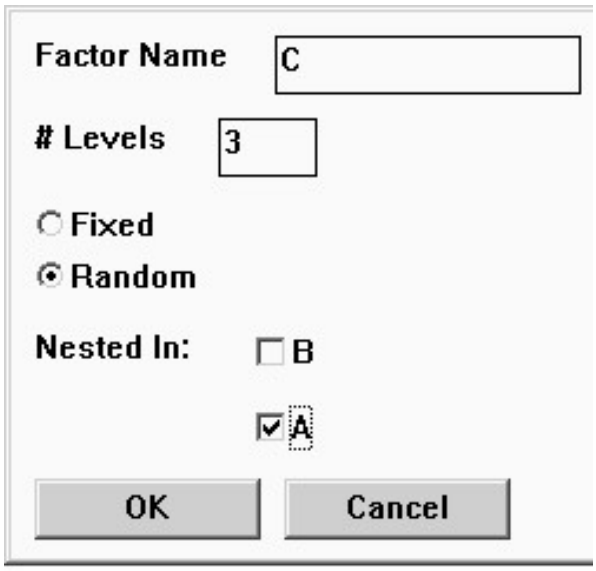

Figure 4: Dialog for adding a factor

```
EMS-A*B = 2*[A*B] + 1*(E)
EMS-E = 1*(E)

Note: (.) is a variance component.
      [.] is a mean squared deviation from a treatment mean.
```

5.2. Example 2:

The next example corresponds to the Hasse diagram in Figure 1. There are two fixed factors, A and B, that are crossed, and a random factor, C that is nested in A. This diagram corresponds to that of a split-plot experiment where A is the whole-plot treatment, B is the sub-plot treatment, and C is the whole-plot experimental unit. The input file needed to specify this design is as follows:

```
(A 5 fixed nil)
(C 3 random (A))
(B 4 fixed nil)
(E 1 random (A C B))
```

This Hasse diagram could also be obtained from the design in Example 1 by adding a random factor, C, that is nested in factor A. This is done using the dropdown menu system and selecting the `Add a Factor ...` option, as shown in Figure 4, with the options required to make this change. A similar dialog is displayed when one shift-clicks on a node in the diagram to change its settings.

The ANOVA table that corresponds to this Hasse diagram shows that the correct error term for A is MS_C, and the correct error term for B and A*B is MS_{C*B}. Note that when the error

term has only one level (i.e., no replication at the finest factor partition), there is an effect with zero degrees of freedom shown in the ANOVA table and the expected mean squares. It is suppressed in the equation for the linear model.

```
Model: Y = u + A + C(A) + B + A*B + C*B(A)

    ANOVA TABLE:

SOURCE            d.f.  Mean Sqr  F-test
-------------     ----  --------  -----------
A                    4  MS-A       MS-A / MS-C.
C(A)                10  MS-C       MS-C / **.
B                    3  MS-B       MS-B / MS-C*B.
A*B                 12  MS-A*B     MS-A*B / MS-C*B.
C*B(A)              30  MS-C*B     MS-C*B / **.
E(A,C,B)             0  MS-E       ** / **.
-------------     ----
Total (corr.)       59

Expected Mean Squares:

EMS-A = 12*[A] + 4*(C) + 1*(E)
EMS-C = 4*(C) + 1*(E)
EMS-B = 15*[B] + 1*(C*B) + 1*(E)
EMS-A*B = 3*[A*B] + 1*(C*B) + 1*(E)
EMS-C*B = 1*(C*B) + 1*(E)
EMS-E = 1*(E)

Note: (.) is a variance component.
      [.] is a mean squared deviation from a treatment mean.
```

5.3. Example 3:

The last example is the 7-factor mixed effects design example from Taylor and Hilton (1981). In an industrial experiment three factories (F) were chosen at random to conduct a study of two kinds (K) of tool rests (R) and two angles (A) of edges for lathe tools (T). Each factory received four tools rests, two of each kind. In each factory three machinists (M) were assigned randomly to use each of the tool rests on four randomly setected lathes (L). Each factory was assigned 96 tools, with one third of the tools being of each angle. To control wear and assure uniformity of tool usage, each tool was used only with a specific lathe and a specific tool rest. The structure diagram used in Taylor and Hilton (1981) for all calculations is shown below.

The input file needed to specify this design is as follows:

```
(F 3 random nil)
(M 3 random (F))
(L 4 random (F))
```

```
(K 2 fixed nil)
(A 3 fixed nil)
(R 2 random (F K))
(T 2 random (F L R K A))
(E 1 random (F M L R K A T))
```

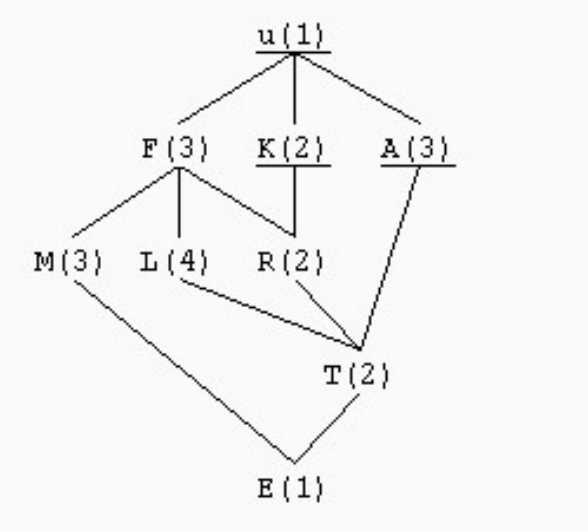

Figure 5: Hasse diagram for Example 3.

```
Model- Y = u + F + M(F) + L(F) + M*L(F) + K + F*K + M*K(F) + L*K(F) +
           M*L*K(F) + A + F*A + M*A(F) + L*A(F) + M*L*A(F) + K*A + F*K*A +
           M*K*A(F) + L*K*A(F) + M*L*K*A(F) + R(F,K) + M*R(F,K) + L*R(F,K) +
           M*L*R(F,K) + A*R(F,K) + M*A*R(F,K) + L*A*R(F,K) + M*L*A*R(F,K) +
           T(F,L,K,A,R) + M*T(F,L,K,A,R)


    ANOVA TABLE:

SOURCE         d.f.  Mean Sqr  F-test
-------------  ----  --------  -----------
F                 2  MS-F       MS-F + MS-M*L + MS-M*R + MS-L*R
                                 -------------------------------
                                 MS-M + MS-L + MS-R + MS-M*L*R.
M(F)              6  MS-M       MS-M + MS-M*L*R
                                 ---------------
                                 MS-M*L + MS-M*R.
L(F)              9  MS-L       MS-L + MS-M*L*R
                                 ---------------
                                 MS-M*L + MS-L*R.
M*L(F)           18  MS-M*L     MS-M*L / MS-M*L*R.
K                 1  MS-K       MS-K / MS-F*K.
```

```
F*K              2  MS-F*K     MS-F*K + MS-M*L*K + MS-M*R + MS-L*R
                                ----------------------------------
                                MS-M*K + MS-L*K + MS-R + MS-M*L*R.
M*K(F)           6  MS-M*K     MS-M*K + MS-M*L*R
                                -----------------
                                MS-M*L*K + MS-M*R.
L*K(F)           9  MS-L*K     MS-L*K + MS-M*L*R
                                -----------------
                                MS-M*L*K + MS-L*R.
M*L*K(F)        18  MS-M*L*K   MS-M*L*K / MS-M*L*R.
A                2  MS-A       MS-A / MS-F*A.
F*A              4  MS-F*A     MS-F*A + MS-M*L*A + MS-M*A*R + MS-L*A*R
                                -------------------------------------
                                MS-M*A + MS-L*A + MS-A*R + MS-M*L*A*R.
M*A(F)          12  MS-M*A     MS-M*A + MS-M*L*A*R
                                -------------------
                                MS-M*L*A + MS-M*A*R.
L*A(F)          18  MS-L*A     MS-L*A + MS-M*L*A*R
                                -------------------
                                MS-M*L*A + MS-L*A*R.
M*L*A(F)        36  MS-M*L*A   MS-M*L*A / MS-M*L*A*R.
K*A              2  MS-K*A     MS-K*A / MS-F*K*A.
F*K*A            4  MS-F*K*A   MS-F*K*A + MS-M*L*K*A + MS-M*A*R + MS-L*A*R
                                ----------------------------------------
                                MS-M*K*A + MS-L*K*A + MS-A*R + MS-M*L*A*R.
M*K*A(F)        12  MS-M*K*A   MS-M*K*A + MS-M*L*A*R
                                --------------------
                                MS-M*L*K*A + MS-M*A*R.
L*K*A(F)        18  MS-L*K*A   MS-L*K*A + MS-M*L*A*R
                                --------------------
                                MS-M*L*K*A + MS-L*A*R.
M*L*K*A(F)      36  MS-M*L*K*A MS-M*L*K*A / MS-M*L*A*R.
R(F,K)           6  MS-R       MS-R + MS-M*L*R
                                ---------------
                                MS-M*R + MS-L*R.
M*R(F,K)        12  MS-M*R     MS-M*R / MS-M*L*R.
L*R(F,K)        18  MS-L*R     MS-L*R + MS-M*T
                                ---------------
                                MS-M*L*R + MS-T.
M*L*R(F,K)      36  MS-M*L*R   MS-M*L*R / MS-M*T.
A*R(F,K)        12  MS-A*R     MS-A*R + MS-M*L*A*R
                                -------------------
                                MS-M*A*R + MS-L*A*R.
M*A*R(F,K)      24  MS-M*A*R   MS-M*A*R / MS-M*L*A*R.
L*A*R(F,K)      36  MS-L*A*R   MS-L*A*R + MS-M*T
                                -----------------
                                MS-M*L*A*R + MS-T.
```

```
M*L*A*R(F,K)     72  MS-M*L*A*R MS-M*L*A*R / MS-M*T.
T(F,L,K,A,R)    144  MS-T       MS-T / MS-M*T.
M*T(F,L,K,A,R)  288  MS-M*T     MS-M*T / **.
E(F,M,L,K,A,R,T)  0  MS-E       ** / **.
-------------     ----
Total (corr.)      863

Expected Mean Squares:

EMS-F = 288*(F) + 96*(M) + 72*(L) + 24*(M*L) + 72*(R) + 24*(M*R) + 18*(L*R)
         + 6*(M*L*R) + 3*(T) + 1*(M*T) + 1*(E)
EMS-M = 96*(M) + 24*(M*L) + 24*(M*R) + 6*(M*L*R) + 1*(M*T) + 1*(E)
EMS-L = 72*(L) + 24*(M*L) + 18*(L*R) + 6*(M*L*R) + 3*(T) + 1*(M*T) + 1*(E)
EMS-M*L = 24*(M*L) + 6*(M*L*R) + 1*(M*T) + 1*(E)
EMS-K = 432*[K] + 144*(F*K) + 48*(M*K) + 36*(L*K) + 12*(M*L*K) + 72*(R)
         + 24*(M*R) + 18*(L*R) + 6*(M*L*R) + 3*(T) + 1*(M*T) + 1*(E)
EMS-F*K = 144*(F*K) + 48*(M*K) + 36*(L*K) + 12*(M*L*K) + 72*(R) + 24*(M*R)
         + 18*(L*R) + 6*(M*L*R) + 3*(T) + 1*(M*T) + 1*(E)
EMS-M*K = 48*(M*K) + 12*(M*L*K) + 24*(M*R) + 6*(M*L*R) + 1*(M*T) + 1*(E)
EMS-L*K = 36*(L*K) + 12*(M*L*K) + 18*(L*R) + 6*(M*L*R) + 3*(T) + 1*(M*T)
         + 1*(E)
EMS-M*L*K = 12*(M*L*K) + 6*(M*L*R) + 1*(M*T) + 1*(E)
EMS-A = 288*[A] + 96*(F*A) + 32*(M*A) + 24*(L*A) + 8*(M*L*A) + 24*(A*R)
         + 8*(M*A*R) + 6*(L*A*R) + 2*(M*L*A*R) + 3*(T) + 1*(M*T) + 1*(E)
EMS-F*A = 96*(F*A) + 32*(M*A) + 24*(L*A) + 8*(M*L*A) + 24*(A*R) + 8*(M*A*R)
         + 6*(L*A*R) + 2*(M*L*A*R) + 3*(T) + 1*(M*T) + 1*(E)
EMS-M*A = 32*(M*A) + 8*(M*L*A) + 8*(M*A*R) + 2*(M*L*A*R) + 1*(M*T) + 1*(E)
EMS-L*A = 24*(L*A) + 8*(M*L*A) + 6*(L*A*R) + 2*(M*L*A*R) + 3*(T) + 1*(M*T)
         + 1*(E)
EMS-M*L*A = 8*(M*L*A) + 2*(M*L*A*R) + 1*(M*T) + 1*(E)
EMS-K*A = 144*[K*A] + 48*(F*K*A) + 16*(M*K*A) + 12*(L*K*A) + 4*(M*L*K*A)
         + 24*(A*R) + 8*(M*A*R) + 6*(L*A*R) + 2*(M*L*A*R) + 3*(T) + 1*(M*T)
         + 1*(E)
EMS-F*K*A = 48*(F*K*A) + 16*(M*K*A) + 12*(L*K*A) + 4*(M*L*K*A) + 24*(A*R)
         + 8*(M*A*R) + 6*(L*A*R) + 2*(M*L*A*R) + 3*(T) + 1*(M*T) + 1*(E)
EMS-M*K*A = 16*(M*K*A) + 4*(M*L*K*A) + 8*(M*A*R) + 2*(M*L*A*R) + 1*(M*T)
         + 1*(E)
EMS-L*K*A = 12*(L*K*A) + 4*(M*L*K*A) + 6*(L*A*R) + 2*(M*L*A*R) + 3*(T)
         + 1*(M*T) + 1*(E)
EMS-M*L*K*A = 4*(M*L*K*A) + 2*(M*L*A*R) + 1*(M*T) + 1*(E)
EMS-R = 72*(R) + 24*(M*R) + 18*(L*R) + 6*(M*L*R) + 3*(T) + 1*(M*T) + 1*(E)
EMS-M*R = 24*(M*R) + 6*(M*L*R) + 1*(M*T) + 1*(E)
EMS-L*R = 18*(L*R) + 6*(M*L*R) + 3*(T) + 1*(M*T) + 1*(E)
EMS-M*L*R = 6*(M*L*R) + 1*(M*T) + 1*(E)
EMS-A*R = 24*(A*R) + 8*(M*A*R) + 6*(L*A*R) + 2*(M*L*A*R) + 3*(T) + 1*(M*T)
         + 1*(E)
EMS-M*A*R = 8*(M*A*R) + 2*(M*L*A*R) + 1*(M*T) + 1*(E)
```

```
EMS-L*A*R = 6*(L*A*R) + 2*(M*L*A*R) + 3*(T) + 1*(M*T) + 1*(E)
EMS-M*L*A*R = 2*(M*L*A*R) + 1*(M*T) + 1*(E)
EMS-T = 3*(T) + 1*(M*T) + 1*(E)
EMS-M*T = 1*(M*T) + 1*(E)
EMS-E = 1*(E)
```

6. Discussion

This paper has described a Lisp-Stat implementation of Hasse diagrams for balanced, mixed effects designs. It allows the user to describe an experimental design by listing the factors and the nesting relationships among the factors. It provides an interactive Hasse diagram that lets the user visualize even complex crossing and nesting relationships. From the diagram, the system determines the appropriate linear model, expected mean squares, and ANOVA table with F-tests for each of the effects. The user can interactively change the design and see the effects on the output. The Hasse diagram can be used to describe more general designs, but the calculations derived from it only apply to balanced designs.

The system can be used for teaching these design and analysis concepts. A thorough understanding of partial orderings is not required to understand Hasse diagrams. In fact, the Hasse diagram could help students understand nesting and crossing relationships, as well as derivations of appropriate F-tests and expected mean squares.

In addition to the use of Hasse diagrams as a pedagogical tool, Marasinghe and Darius (1990) and Lohr (1995) have pointed out a more proactive use as an interactive and expert aid for designing experiments. If it is possible to derive models and compute analyses of variance resulting from a modified structure instanteously, one would be able to determine and keep track of the effects of changes to, additions, or deletions of one or more nodes from an existing diagram. This ability to provide feedback to possible design modifications complements the process that naturally occurs when complex multifactor experiments with multiple levels of nesting and crossing are developed.

Another application of Hasse diagrams occurs when statistical consultants are called upon to analyze experiments they were not involved with in the planning stages. The familiar query-and-response session between a statistical consultant and an experimenter could lead to a formulation of an appropriate Hasse diagram, from which an acceptable analysis can be obtained. Lohr (1995), through the use of several examples, describes how a Hasse diagram may be constructed interactively using such a technique.

In each of the above cases, the statistician, can proceed by first determining the factors that are not nested in any other factor. These are at the top level of factors in the structure diagram. Then the factors that are nested in the factors in the top-level are determined and so on. One example where the software has been used was to design a microarray experiment to look at the effects of batch, day, operator, dye, and cell culture. There were choices about how to split the experimental units, which gave rise to different crossed and nested relationships. Starting with the highest level factor(s), we constructed a diagram for each design. Then we compared different designs to see which effects were estimable and thus choose a design that met the scientists' needs.

7. Software availability

The program, along with several example design files and instructions for getting started, is available at this web site: http://www.public.iastate.edu/~stat/Modules/Hasse.

References

Carney EJ (1967). *Computation of Variances and Covariances of Variance Component Estimates*. Ph.D. thesis, Deptartment of Statistics, Iowa State University.

Kempthorne O (1982). "Classificatory Data Structures and Associated Linear Models." In G Killianpur, PR Krishnaiah, JK Ghosh (eds.), "Essays in Honor of C. R. Rao," pp. 397–410. North Holland, New York.

Lohr SL (1995). "Hasse Diagrams in Statistical Consulting and Teaching." *The American Statistician*, **49**(4), 376–381.

Marasinghe MG, Darius PL (1990). "A Structure-based Approach for Model Determination in Experimental Designs." In "Proceedings of the Statistical Computing Section," pp. 143–150. American Statistical Association.

Taylor WH, Hilton HG (1981). "A Structure Diagram Symbolization for Analysis of Variance." *The American Statistican*, **35**(2), 85–93.

Throckmorton TN (1961). *Structures of Classification Data*. Ph.D. thesis, Deptartment of Statistics, Iowa State University.

Tierney L (1990). *Lisp-Stat: An Object-Oriented Environment for Statistical Computing and Dynamic Graphics*. Wiley, New York.

Affiliation:

Philip Iversen
Drop Code 2233
Lilly Corporate Center
Eli Lilly and Company
Indianapolis, IN 46268, United States of America
E-mail: piversen@lilly.com

Mervyn G. Marasinghe
Department of Statistics
Iowa State University
117 Snedecor Hall
Ames, IA 50011, United States of America
E-mail: mervyn@iastate.edu
URL: http://www.public.iastate.edu/~mervyn/

Journal of Statistical Software
February 2005, Volume 13, Issue 3.
http://www.jstatsoft.org/

Submitted: 2004-05-06
Accepted: 2004-12-20