Contents

Chapter 2 — Input and output formats Release 2-c/11/02

Chapter 3 — Configuration Release 2-c/11/02

Chapter 4 — Elicitation Release 2-c/11/02

Chapter 5 — Creating an AFS (system) file Release 2-c/11/02

Chapter 6 — Designing study programmes Release 2-c/11/02

Chapter 7 — Updating Release 2-c/11/02

Chapter 8 — Reporting Release 2-c/11/02

Chapter 9 — File Management Release 2-c/11/02

Chapter 10 — Auditing Release 2-c/11/02

Appendix A — Glossary Release 2-c/11/02

Appendix B — The Sample Data Set Release 2-c/11/02

Appendix C — Certification Release 2-c/11/02

Appendix D — System Requirements Release 2-c/11/02

Comment Sheet

Preface

ABLE version 2 represents a substantial upgrade of the original ABLE software. The principal new features in version 2 are as follows.

- Version 2 is designed to run under Microsoft Windows. Whilst it is not a full Windows implementation, version 2 is much easier to work with and more suited to a modern business computing environment that the MS-DOS based version 1.
- Version 2 introduces many new operations for building ABLE models, together with a simulation based computation for reporting which accommodates these new operations.
- Version 2 introduces a process of variance learning which greatly simplifies the validation of models. It removes much of the most time-consuming activity in updating models with new data.

There are numerous smaller improvements. The ABLE Development Group is a consortium of ABLE users which has promoted the development of ABLE version 2, and will instigate further initiatives as necessary. For instance, future development is likely to include extension of the Design module to make use of simulation-based computation, and improvements to its optimisation algorithms.

ABLE version 1 was developed primarily for use by water companies in preparing Asset Management Plans. It is now being employed in other industries. The rewriting of this manual reflects its wider readership, although the principal example which illustrates the various ABLE features throughout the manual is drawn from the water industry. Other additions at various points give the user more helpful, informal advice in addition to the formal details of how to use the software.

ABLE version 2.3.6

This is a free upgrade release for users of ABLE 2. Here are the principal differences from the original release of ABLE 2.

- A feature has been provided to enable equations to be pasted from the clipboard into the equations editor of the Create/Edit module. This is activated from a new `Paste` item in the `Edit` menu. It makes building large models easier because these can now be constructed in an ordinary text editor, using all the cut/paste/copy features and then just copied to the clipboard and pasted into ABLE.
- By default, output produced by the Reporting module now includes information about the computation method. It says whether exact or simulation-based computation was used. In the case of simulation-based computation it gives the sample size.
- The procedure by which equations are colour-coded to distinguish visually between base quantities, derived quantities, operations and mappings can take a long time for large models. The user is now offered the choice of not colouring the equations.
- When building or editing the information about a group in the Create/Edit module, the user is expected to progress through the various screens in strict order. However, a feature has now been added to allow the user to

step back through the screens. This is done by a new `Back` item in the `Edit` menu. This is useful when wishing to correct something in an earlier screen, which in the original release would have meant aborting the group and starting again.

- The sequence of screens for building or editing information on a group in the Create/Edit module is different for a group which has not yet been updated from one that has been updated (using variance learning). A feature has been added to allow an updated group to be viewed in the same way as an un-updated group. This is achieved by using Ctrl+Shift+Enter, instead of Ctrl+Enter, to begin viewing the screens for a given group. However, this should be used with great care. If the user continues to the end of the group and saves the results at the end, the group is now perceived by ABLE as being un-updated, and subsequent operations may be affected. (The feature is allowed because an expert ABLE user might wish to achieve exactly this in some circumstances, but even then should be aware that the audit trail will be compromised.)
- A number of minor bugs have been fixed and other minor improvements made.

Finally, users should note the following solution to a bug that has not been fixed yet. If the cursor disappears, it can generally be made to return by pressing the Insert key twice.

Change Control Procedures

The following procedures are followed when issuing updates to this manual. All references to chapters apply equally to appendices.

1. The smallest unit that will be reissued is a chapter.
2. The contents list indicates the documentation release level of each chapter.
3. The documentation release level is defined as: n-α/m/yy
   where

   n is the first digit of the software release described.

   α is an alphabetic character which runs from a to z for each release of a given chapter (within release n of the software).

   m (one or two digits) indicates the month of the release.

   yy (two digits) indicates the year of the release.

   The documentation release level is repeated at the foot of each page.
4. When updated chapters are released within the same software release, substantive changes to the text are indicated by a vertical line in the right margin.

Please note that the Comment Sheet at the back of the manual can be used to notify the Statistical Services Unit at the University of Sheffield of any difficulties encountered in using this manual (including notification of errors).

Typing Conventions

It is hoped that most readers will be sufficiently familiar with the use of a PC and keyboard to be able to deduce from the context when a single key stroke, key combination or sequence of characters is called for. The following typing conventions and phraseology are, however, used throughout this manual when keyboard activities are referred to.

"press"	indicates a single key, thus "press `F2`"
"key"	(as a verb) indicates a key combination, thus "key `[Ctrl+Delete]`" means "while holding down the `Ctrl` key, press the key engraved `Delete`".
"type"	"type" indicates a sequence of characters to be typed, thus "type `??`"

In addition, screen displays and keyboard input are normally shown:

`in this fixed pitch font.`

Typical single keystrokes are:

↵ (the Enter key)
`Delete`
`F1`
`F2` etc

Typical key combinations are:

`[Ctrl+↵]`
`[Shift+Esc]`
`[Shift+→]`

Selections from Windows pull-down menus are shown as, for example: `File>Open` (meaning select the Open option from the File pull-down menu).

Notes:

- The Enter key is the key engraved Enter, Return or ↵, and is represented throughout as ↵
- Where the left arrow, ← key is referred to this means the cursor key (usually available as one of a block of ↓ ↑ → ← keys), not the "backspace" key (which is normally at the top right of the main block of keys).
- "The `Delete` key" is the key engraved `Delete` or `Del` (not the "backspace" key).
- The "escape" key is normally engraved `Esc`.
- "Back Tab" (shown on screen displays as Btab) is the key combination `[Shift+Tab]`
- **!** This symbol is used in the margin to attract your attention to particularly critical items of information.
- **1-2-3** The 1-2-3 combination is used to draw your attention to places where instructions are specified for the user of the package.

ABLE:

Chapter 1 — Introduction and overview

1.1 Introduction

ABLE was originally developed for a specific task faced by many British water companies, although it was always designed to be much more widely applicable. Water companies are required to present Asset Management Plans (AMPs) detailing the assets owned by the company and the projected costs of maintenance. This is a complex task even for those assets that can be seen and inspected, made many times more complicated by the fact that so many of a water company's assets are hidden from view; and the state of repair—and consequent maintenance costs—can be derived only through informed guesswork.

Following extensive experience with version 1 of ABLE, version 2 has now been created, implementing a number of improvements and extensions to the statistical theory of Bayes Linear Estimation (BLE) which underlies the package. Whilst water companies continue to be its main users, ABLE has already been applied by other companies. Further spread of its use may be expected to result from the launch of ABLE version 2.

ABLE is a complex piece of software and is unique. The largest task facing a new user is to appreciate what it is that ABLE does, and what potential uses it has. Whilst ABLE will invariably be acquired by an organisation for a specific task, such as an AMP, it is a versatile package. By understanding its potential, the organisation may see other applications to gain even more value from their investment in ABLE.

The best way to gain this understanding is through an ABLE training course. This manual contains all the necessary information, but it is best seen as amplifying what can be taught on a course, and as a reference for more experienced users.

As a first step in understanding ABLE, an analogy is presented here which relates ABLE to a much more every day piece of software—the familiar spreadsheet. The analogy is developed first through a trivial and whimsical problem, to introduce ideas very simply. It is then extended through a more serious problem which is typical of the tasks for which ABLE is uniquely suited.

1.1.1 A simple analogy

In order to set in context some of the very involved processes with which the ABLE program is concerned, let us consider a very simple analogy without reference to the complexities of asset management or any other real-life ABLE application.

Suppose you have been asked to provide a punchbowl for a party, and to say how much it is going to cost. The punch has three ingredients: rum, wine and orange juice. You know the quantity of each required, and you know the cost of rum and wine precisely. The accuracy of your estimate of the total cost therefore relies entirely on the accuracy of your estimate of the cost of orange juice. You can, however, make a reasonable guess at the price, and you can judge by how much that guess is likely to be wrong.

If you have to guess the price of all three ingredients (and you are not absolutely sure of the amount of each required) the problem of judging how accurate your costing figure is becomes several times more difficult. It is clear, however, that there are two essential factors involved in each guess: the estimate itself and its accuracy.

1.1.2 The punchbowl spreadsheet

You could set up a spreadsheet to compute the total cost (although, unless you deeply distrust your ability to do simple multiplication and addition, it would hardly be worth the effort for such a simple calculation!). Your sheet would have three rows, and you could label these as RUM, WINE and ORANGE. One column called PRICE would give the unit price, per bottle or per centilitre, perhaps, of each ingredient. Another column would give the QUANTITY (in bottles, centilitres or whatever unit was used for the price) required for each ingredient. You would then program the spreadsheet to multiply these two columns together to yield a new column we can call COST, which shows the cost to buy that quantity of each ingredient. Finally, you would program the spreadsheet to total this column to produce the total cost.

Once you have entered figures into the PRICE and QUANTITY columns, setting the spreadsheet to compute will produce figures in the COST column and the total cost. Of course, the spreadsheet treats these figures as accurate, and does not know anything about the fact that they are really just based on your *estimates* of prices and quantities. However, the nice property of a spreadsheet—that you can easily change the original numbers and recompute the consequences—provides a kind of solution. You only have to keep changing the prices and quantities, within the bounds that represent possible estimation errors, and recomputing to build up a picture of how the uncertainty about prices and quantities translates into uncertainty about the total cost.

This would be a laborious process, however. You would need to try many different combinations and permutations of the changes—now increasing the price of wine, decreasing that of rum and holding the orange price unchanged, now holding the wine price unchanged and decreasing both the rum and orange prices, and so on. Fortunately, statisticians have some theory about how the uncertainties combine in a simple example like this, and that theory is built into ABLE. In more complex problems, where statistical theory does not give exact solutions, ABLE does the same process of trying very many permutations of changes, automatically and quickly.

So an ABLE application could be set up for the punchbowl problem. It would be like the spreadsheet but would additionally do these computations to work out the effect on total cost of your uncertainty about prices and quantities. Unlike the spreadsheet, ABLE knows about uncertainties. It expects *all* the raw data, like prices and quantities, to be just estimates, subject to uncertainty. You can, nevertheless, make it clear to ABLE when a number *is* exact, not just an estimate, by specifying zero uncertainty for it.

1.1.3 Variances

Statisticians measure uncertainty most often by a measure called *variance*. An estimate with large variance has large uncertainty, and so could be a long way from the true value. Indeed, a variance is formally an expected squared difference between the estimate and the true value. So if you estimate rum as having a price of 7 (pounds sterling per bottle), and a variance of 4, then you are saying that the difference between the estimate and the true price would be around 2 (the square root of 4). So the price could be as low as 5 or as high as 9. An engineer might express the price as 7±2. This would represent rather a poor estimate, whereas a variance of 1 (price 7±1) would be more accurate.

Note that although we have used expressions like 7±2 and talked of the true price being “around 2” different from the estimate, the real difference could be smaller (even much smaller) than 2, or larger (even much larger). The whole point about uncertainty is that we do not know what the difference will be. But for a variance of 4 a difference of up to ±2 would be quite likely in a statistical sense. In fact, as a rough “rule of thumb”, the true value is about twice as likely to lie within that range 7±2, i.e. from 5 to 9, as to be outside it.

Variances are clearly important in ABLE. ABLE needs to know the variance which it should attach to each raw data entry. There are facilities within ABLE, in its “Elicitation module”, for specifying these variances (where the “rule of thumb” is an intrinsic part of the process). When the user does the ABLE equivalent of asking for the spreadsheet to be recomputed, ABLE not only computes estimates for the programmed quantities, (like the COST column or the total cost) but also variances. It is in this way that it solves the original punchbowl problem of expressing what the uncertainty is in the total cost, as a result of uncertainty about prices and quantities.

1.1.4 The spreadsheet with variances

This gives us the first stage of the spreadsheet analogy for ABLE. ABLE can be thought of as a spreadsheet with variances. Every number in the spreadsheet is, to ABLE, an estimate and has an associated variance. Variances must be specified for the raw data. (ABLE cannot do everything for you!) Then whenever values are computed with the computations programmed into the spreadsheet application, ABLE also computes variances for these computed estimates.

At this point, it is useful to introduce some ABLE terminology. The columns of figures that we have referred to in the spreadsheet are called *lists*. So the punchbowl application would have lists called PRICE, QUANTITY and COST. The TOTAL cost is also formally a list, albeit having only a single value. The rows are called *units*, sometimes also referred to as zones. The names of the rows form the *ID list* for that sheet, or table. ABLE can work with several different sheets, each with its own ID list and with appropriate lists (columns). A powerful operation in ABLE known as *mapping* links lists which have different ID lists, and so enables a computation in one sheet to produce a computed column in another. The punchbowl application would be described in ABLE by means of equations

COST = PRICE * QUANTITY
TOTAL = { COST

The first equation multiplies the PRICE and QUANTITY lists to compute the COST list. The “{” symbol in the second is the ABLE symbol for summation (which is actually a special kind of mapping), so the equation says that the sum of COST values computes the TOTAL cost.

The collection of equations defining ABLE computations (in the same way as a spreadsheet is programmed for a particular application) is called the *model*. The ABLE model for the spreadsheet application comprises just two equations.

1.1.5 Building a motorway

We now consider a second, more serious, application to develop the analogy further. The problem is one of estimating the cost of building a motorway. The motorway is built in a number of sections, and we can define the cost of the whole motorway to be the sum of the costs of the individual sections. The cost for each section is made up of various components, like the cost of ground preparation, the cost of surface laying, the cost of any associated bridges, intersections, etc. The model for this application might begin with equations

TOTAL = { SECTION
SECTION = PREP + LAY + OTHER
WAY = LENGTH * RATE

We have explained the first two equations, and the third expresses the cost for surface laying as a product of the section length (in km) multiplied by the cost per km. The model might go on to break down other lists, like OTHER, into further components. ABLE modelling is very flexible, and models can become highly complex structures built from many equations.

When the motorway building contractor first sets up this model, almost all of the figures in it are estimates. (A possible exception might be the values of LENGTH, the section lengths.) ABLE's ability to reflect the uncertainty in the total cost at this stage will help the would-be contractor in setting their bid at a level which does not expose them to too much risk if they have under-estimated costs.

1.1.6 Updating

Another key feature of ABLE comes into play after the contract has been won and the construction begun. As progress is made with the construction, the contractor will begin to learn about some of the quantities in the model. Fairly early, they may be able to give a more accurate estimate of the cost of ground preparation in the first section. Later, they may know the actual cost of ground preparation in the first section, be able to offer better estimates for the values of this PREP list in some other sections where work has begun, and perhaps have improved estimates of LAY or OTHER in the first section.

As each new improved estimate is obtained, it can be entered into ABLE to replace the old value, and the equations of the model computed to obtain an updated estimate of total cost. This is comparable, so far, to altering one or more numbers in a spreadsheet and recomputing. But ABLE does extra things because it recognises that even the new values are typically also estimates.

First, and obviously, it needs to be told the variances it should attach to new estimates, so that it can take this into account when computing the variance of TOTAL cost. The reduced variances that go with these new estimates will hopefully have reduced the variances of the TOTAL.

However, ABLE incorporates a completely unique extra feature which allows it to extract more value from the new estimates. This feature is a statistical learning facility using Bayesian statistics. (Technically, it is based on a technique known as Bayes Linear Estimation, whose initials were the reason for the name ABLE.) Suppose that the contractor initially estimates the preparation cost in section 1 to be 75 (millions of pounds, perhaps), but that after much of this work has been done a revised estimate of 68 is made. The cost estimate has fallen by almost 10%. Now consider the values of this list PREP in other sections. For instance, PREP for section 2 might have been estimated initially as 46. Having seen a reduction in the estimate in section 1, it is natural to think that 46 might also prove to be an over-estimate. We would probably not go as far as to think that all sections should now have their preparation costs reduced by 10%—the result in section 1 might not be exactly the same as elsewhere—but it might be prudent to reduce the estimate in section 2 from 46 to 44, say.

ABLE does this automatically. Whenever an improved estimate is provided for a quantity in some list, ABLE deduces statistically improved estimates for other quantities in that list. In this way, it makes the maximum use of the information in a new estimate of some quantity, not just looking at what it says about the quantity itself but also deducing what it tells us about other related quantities. The process is called *updating* in ABLE.

There exist commercially available add-ons to spreadsheets to allow them to recognise variances, and to compute variances whenever the programmed computations are invoked, but there is no spreadsheet or add-on which has this powerful updating facility.

1.1.7 The statistical spreadsheet

The twin features of

- recognising variances in all figures, and computing with them whenever recomputation is invoked, and
- ABLE's Bayesian updating, which goes beyond simply replacing old estimates by improved estimates,

make ABLE analogous to a very special kind of *statistical* spreadsheet. The analogy may help the reader to understand and interpret the various ABLE operations as described throughout this manual, but to try to carry the analogy any further here would be to labour the point.

To view ABLE as this unique kind of statistical spreadsheet can be very illuminating, but ABLE is not a spreadsheet. It does not look like a spreadsheet. The machinery needed to take advantage of the above features is complex and not like any operations that one associates with the standard spreadsheet. Analogies are useful, but an analogy is not the real thing, and every analogy has its limitations.

1.1.8 How to use this manual

We said in Section 1.1 that this manual is intended both as support for a training course in ABLE and as a reference. It is organised primarily for reference use. So after this introductory chapter and a chapter dealing with the format of some input and output files, Chapters 3 to 9 deal with each of the “modules” of ABLE in turn, and we end with a chapter on auditing ABLE applications (a task which is specifically required in the UK water industry).

So Chapters 2 to 10 are mainly technical and generally assume that the reader is familiar with what ABLE does, understands the ideas and concepts required to use ABLE, and knows the relevant terminology. The purpose of Chapter 1 is to introduce the reader to what ABLE does, and to some of the principles and terminology. However, the preceding discussion should have indicated how complex ABLE is, and so there is a lot for the new user to learn. To try to explain everything in this first chapter, without reference to the details of actually using ABLE which are explained in subsequent chapters, would make it very indigestible. We therefore postpone discussion of some issues to later chapters.

We have seen how an ABLE application is defined in the first instance by a series of equations, analogous to programming a spreadsheet. This process is explained more fully in Section 1.2, where we work carefully through a complete ABLE model which will be used extensively in this manual to illustrate ABLE's many features. We have also seen that ABLE needs to know about variances of data entered in as estimates of the various quantities in the model, not only the initial estimates entered when the ABLE application is first set up, but also any subsequent estimates which will be used to update the data.

The ABLE module most concerned with defining variances is the Elicitation module, described fully in *Chapter 4 — Elicitation*. We will postpone introducing a range of concepts—levels of uncertainty, stratification, scaling, systematic and random errors—until Chapter 4.

Another important issue concerns what kinds of estimates are most useful. We have seen the analogy of ABLE as a statistical spreadsheet which uses new estimates of quantities to update knowledge about related quantities, and this issue is about how best to make use of this unique capability. The most appropriate place for that discussion is in *Chapter 6 — Designing study programmes*.

1.2 ABLE modelling

1.2.1 The equations

The *Cost Model* is central to the function of the ABLE software. Its purpose is to express quantities which the company wishes to be able to estimate in terms of other, generally simpler, quantities which it finds easier to estimate with some accuracy. For a water company's AMP, for instance, the biggest and most complex thing to estimate is the grand total investment cost. The model equations will steadily break this down in terms first of other quantities which are also of importance to estimate in their own right, and ultimately in terms of simpler building blocks.

ABLE is an extremely flexible piece of software, and one reason is the flexibility to devise model equations which represent widely different applications. We call the collection of equations the "Cost Model" for historical reasons, since the water companies for whom ABLE was originally developed were primarily interested in estimating costs, but equations could readily be written for applications unrelated to costs.

The user will only learn with experience how to use ABLE's modelling power, and it would be out of place in this manual to try to go into too much detail. However, this section will provide some guidance through a small but non-trivial example. The same example is provided with the ABLE software and is developed throughout this manual.

It is worth studying this example not only for modelling guidance but also because it introduces various ABLE concepts which will be important later.

1.2.2 Demonstration model

Our example is intended to represent part of a larger model for a water company. It contains a part of the company's model relating to some of the costs for maintaining and improving water distribution mains. It should be stressed that any actual company might choose to model this sample problem quite differently, because they think of the costs being composed differently, because they have different data available to estimate the simplest quantities, or for many other reasons. The model has been constructed partly to illustrate new modelling features available in ABLE version 2, and partly to make some important general points about modelling, considerations which mean that it may be less than completely realistic!

MODELTOTAL = { DIVCOST

This is the first equation. On the left hand side is the grand total cost, at least for this model, which may be part of some larger model. The equation expresses this MODELTOTAL as a sum of divisional totals. The opening curly bracket symbol in an ABLE equation represents summation.

MODELTOTAL is a single quantity, but the name DIVCOST represents a *list* of quantities, a value for each of the company's divisions. In general, any quantity name in an ABLE equation represents a list, and we sometimes refer to DIVCOST as a *generic* quantity. All the quantities in these equations except MODELTOTAL are generic, representing a list of more than one specific quantity.

DIVCOST = DIVISION { ZONECOST

The next equation expresses the division costs as sums of zone costs. A water company’s water distribution may be divided into hundreds of water supply zones, and in this model we suppose that each division contains many zones. In ABLE, the summation symbol on its own sums *all* the quantities in a list, resulting in a single value, just as in the previous equation. But in order to form the divisional totals we need to add together the zone costs for all those zones which are in each division, producing not a single total but a list of partial totals.

The name to the left of the summation sign is not a quantity name but (*because* it comes to the left of a summation sign) is a different thing altogether called a *mapping*. The mapping DIVISION is simply there to specify which zones are in which divisions. (A summation sign with no name to the left is technically known as a null mapping.) ABLE’s concept of a mapping is actually more flexible than a means of forming partial totals, and is explained more fully in Section .

Note that the first equation could have been expressed as

MODELTOTAL = { ZONECOST

since the sum of division costs is obviously also the sum of all the zone costs. In any system of equations there will always be alternative ways of writing them. It usually does not matter much which of two mathematically equivalent formulations you use. The form of the first equation was chosen in this case to make the exposition clearer, although expressing MODELTOTAL as the sum of ZONECOST will probably lead to slightly faster computation (because to compute MODELTOTAL, ABLE does not need to use the more complex mapping DIVISION to compute DIVCOST first).

ZONECOST = REHABC + PRESSURE + SYSGROW

This expression defines the zone cost to be a sum of costs for three different categories of work—for water main rehabilitation, for relieving problems of low pressure, and for reinforcing the system to meet increased demand. Every one of these generic quantities is a list of individual quantities, one for each zone.

Subsequent equations formulate each of the three different kinds of costs in more detail.

REHABC = LENTH * REHABUC

The star symbol, *, means multiplication. This equation says rehabilitation costs can be expressed as the length of mains needing rehabilitation multiplied by a rehabilitation unit cost. Remember that these are generic quantity symbols, and so the equation says that *in each zone* the rehabilitation cost is represented by such a product. The length LENTH in a particular zone is the length of mains in that zone needing rehabilitation. Similarly, the value of REHABUC in a particular zone is the unit cost in that zone.

It is important to realise that this is not the kind of unit cost that an engineer would use to “cost” a project of rehabilitation. The rehabilitation cost REHABC in the zone is whatever the actual cost *will* turn out to be for that zone, and LENTH is the length of mains that *will* turn out to need rehabilitation. In effect the unit cost REHABUC is *defined* to be the ratio of these two—it is whatever it *will* cost per unit length to rehabilitate *those* mains in *this* zone. So it will vary from zone to zone and REHABUC is a list like REHABC or LENTH.

This idea of expressing a cost as a product of a quantity of work and a unit cost is common in ABLE cost models, and generally requires this idea of an *out-turn* unit cost specific to a zone or asset group.

LENTH = IRON5 + IRON4 + (IRON3 * DECAY)

Water companies classify mains by condition grade from grade 1 (best, “as new”) to grade 5 (worst, “derelict”). This equation expresses our hypothetical company’s intention to rehabilitate all iron mains in grades 4 and 5, plus all those grade 3 pipes which decay to grade 4 or 5 over the planning period. So IRON3, IRON4 and IRON5 are lengths of iron mains in those condition grades, in each zone, and DECAY is the proportion of grade 3 mains which decay to grade 4 or 5 (an “out-turn” proportion—see the discussion of unit costs in the preceding paragraph).

DECAY is the first quantity which we have introduced in the model which will not later be the subject of another equation breaking it down into other quantities. It is therefore the first of the basic building blocks which the grand MODELTOTAL is ultimately composed from. It is called a (generic) *base quantity*, or more accurately a *base list*. Any quantity which appears on the left hand side of an equation is called a (generic) *derived* quantity, or derived list. Everything in the equations which is not a mapping or a derived list is a base list.

IRON5 = exp (logIRON5)
IRON4 = exp (logIRON4)
IRON3 = exp (logIRON3)

These equations may seem strange but serve an important purpose. Note first that this kind of equation was not available in ABLE version 1, which allowed equations to contain only the operations of mapping, addition, subtraction and multiplication. This exponential function is one of a range of new operations introduced in ABLE version 2. Not all possible operations are illustrated in this model (see *Chapter 5 — Creating an AFS (system) file*— for a full list).

The reason for using the exponential function here is that ABLE works best when base quantities have symmetric uncertainty. To explain this point, we first have to recognise that typically *none* of the quantities in the model will have known values. Our water company will not know what length of iron mains are in grade 3 in any zone, and would not find out without digging up every single water main (with consequent unimaginable cost, plus disruption to traffic and to life generally in the zone!).

So the best that can be done is to *estimate* any quantity, and there will inevitably be uncertainty about how far above or below the estimate the true value actually lies. By the phrase “symmetric uncertainty”, we mean the idea that the true value is just as likely (and no more likely) to be equal to the estimate minus *x* as to be equal to the estimate plus *x*, for any *x*. Overestimation and underestimation by any amount (*x*) are equally likely.

Returning to IRON3, on the whole we would not expect uncertainty to be symmetric in this way. Suppose we let *x* equal the estimate itself. Then symmetric uncertainty would require the true value to be equally likely to be zero as to be double the estimate. Whereas (in the light of the poor information which the company would typically have about such a quantity) it may be quite possible for the true value to be double the estimate, it is less likely to be zero. In fact, we may think that the true value is as likely to be half the estimate as double the estimate. Instead of the differences between true value and estimate (positive or negative), being equally likely it is often more plausible that it is ratios (either way up—true/estimate or estimate/true) that are equally likely. This leads to the idea that the logarithm of IRON3 would have symmetric uncertainty. So the purpose of these equations is to make logIRON5, logIRON4 and logIRON3 base lists instead of IRON5, IRON4 and IRON3 (which are made into derived lists).

PRESSURE = KNOWNPRES + OTHERPRES
KNOWNPRES = PRESMAP { SCHEMES
SCHEMES = exp (logSCHEMES)

The first of these equations expresses the cost to deal with problems of low water pressure into two parts. The company will be aware of which properties in a zone are reported as having low pressure, and in many cases will already have identified a scheme of remedial work to deal with a group of properties who receive low pressure due to a particular cause. KNOWNPRES is the cost to carry out all these previously identified schemes. The second equation uses another mapping to sum the costs for all schemes relating to each zone. So SCHEMES is a list, the individual quantities in the list being costs for individual schemes. Some zones may have no such previously identified schemes (in which case the KNOWNPRES for that zone is zero, and is an exception to the rule that quantities in the model do not have known values), and some may have several. The third equation makes logSCHEMES the base list.

OTHERPRES = PRESCOST * (LOWPROPS ** 0.5)
LOWPROPS = exp (logLOWPROPS)

The cost for relieving low pressure in properties for which no specific cause and scheme of remedial work has been identified is represented in terms of a quantity and a unit cost, but with a difference. LOWPROPS is the number of such properties (in each zone), but our hypothetical company's experience is that the cost of dealing with them is unlikely to be directly proportional to their number. The more properties there are receiving low pressure, the more likely it is that just a few schemes will be needed because probably many properties have problems due to the same cause. The company's judgement is that the cost is likely to be roughly proportional to the square root of LOWPROPS, and this is what "** 0.5" in the equation means—the double star represents raising to a power (and square-rooting is mathematically equivalent to raising to the power of 0.5).

PRESCOST and logLOWPROPS are base lists.

SYSGROW = SYSGROWS + SYSGROWL

The cost to accommodate growth in demand is also expressed as a sum of two parts, but in a different way from pressure costs because in this case one of the two terms will always be zero. The idea is that modest growth can be accommodated by relatively little expenditure, and this is represented by SYSGROWS. If on the other hand growth is sufficiently large then major work is likely to be required, and this is SYSGROWL. The subsequent equations, in addition to formulating these costs, determine which of the two costs are appropriate for each zone.

SYSGROWL = BIGINC * (INCDEMAND - CAPACITY) * GROWCL
BIGINC = test(INCDEMAND > CAPACITY)

The second of these equations is the one which decides whether growth is big enough to trigger the larger cost level. The expression test(INCDEMAND > CAPACITY) produces a value 1 if the condition being tested is true and a value 0 if it is false. INCDEMAND is the (generic) quantity defining the amount of growth (to be further expanded in the next equation) and CAPACITY is a (generic) base quantity defining the trigger level, which a notational measure of a zone's spare capacity.

The first equation multiplies the amount by which the growth exceeds CAPACITY by a (large) unit cost GROWCL, but this is also multiplied by BIGINC. If BIGINC is 0, then SYSGROWL is 0, otherwise it is (INCDEMAND - CAPACITY) * GROWCL.

The ability to create a zero-one quantity like BIGINC using ABLE version 2's test(…) function is an important development. It is clear that you are uncertain about whether it is 0 or 1 in any zone, and this uncertainty will generally *not* be what we have called symmetric. An estimate of such a quantity will be a value between 0 and 1, which represents a *probability* that it is 1. The greatest uncertainty you could have about it would be given by an estimate of 0.5, so that you think there are equal chances of it being 0 or 1. In that case your uncertainty is symmetric, since the only two possible *true* values, 0 and 1, are equidistant from the estimate 0.5, and equally likely. But in every other situation your uncertainty is asymmetric. So, although zero-one quantities are very useful in modelling, where these act as switches, you should always try to avoid having them as *base* quantities. Constructing BIGINC from a test(INCDEMAND > CAPACITY) in this way makes it a derived quantity, which is greatly preferable.

To emphasise this point, consider a complication which we could have introduced into this demonstration model (but has been left out because the model is already quite complicated enough for its role in this manual!). Remember that LOWPROPS is the number of properties experiencing low pressure which have not been dealt with through previously devised schemes in KNOWNPRES. In some zones, we may not at present be aware of any such properties, and LOWPROPS could actually be zero, but equally there may genuinely be some which we are currently unaware of. In that case it would be better to include a suitable zero-one quantity, say LPEXIST, and write LOWPROPS = LPEXIST * exp(logLOWPROPS). Now logLOWPROPS is the logarithm of the number of such properties *if* there are any. Now LPEXIST is a base quantity if we do not add an equation to define it in terms of something else, but it is not easy to see how a device like the CAPACITY-threshold could be found for this situation. The answer may be to have a completely artificial quantity, say LPEX, which just has the property that when we add the equation LPEXIST = test(LPEX > 0) then LPEXIST will have the desired probability of being 1. We cannot develop this idea here any more fully because it demands some statistical expertise, but the point is that it can be done. However, it is complicated and leads to particular complications when we need to update such a quantity with new information (as described in *Chapter 7 — Updating*).

INCDEMAND = GROWTH + NEWDEMD - LEAKRED
GROWTH = POPEQUIV * GROWRATE
POPEQUIV = exp(logPOPEQUIV)

The first of these equations expresses the increase in demand for water as the growth due to existing customers using more water, plus the amount arising as new demand from new customers, minus the amount of water saved through leakage reduction. NEWDEMD and LEAKRED are base lists but GROWTH is further expressed as the current population equivalent served times a growth rate.

SYSGROWS = (1 - BIGINC) * INCPOS * GROWCS
INCPOS = max (INCDEMAND, 0)

Finally, we model the cost of growth when it is not large, so SYSGROWS has the factor (1 - BIGINC), which is 0 if growth will exceed CAPACITY, and otherwise is 1. Now INCDEMAND could be not just smaller than CAPACITY but could actually be negative. In that case, we should have no cost at all, and *not* a negative cost. So INCPOS is defined to be zero if INCDEMAND is negative, through the max (INCDEMAND, 0) function. Finally, GROWCS is a unit cost for accommodating moderate growth.

1.3 System overview

1.3.1 The equations

The *Cost Model* is central to the function of the ABLE software. As the example presented in Section 1.2.2 has shown, there are two types of quantity involved in the equations:

- the ***Base Quantities*** that require values to be supplied in order that the equations can function (for example DECAY, logLOWPROPS);
- the ***Derived Quantities*** that are defined, or calculated, by the equations (for example ZONECOST, REHABC).

The equations can be modified and/or extended at any stage should this prove necessary.

1.3.2 The prior data

Once the equations have been defined, the prior data for the base quantities must be collected. As we have already remarked, this information will be available to a greater or lesser degree of accuracy. Indeed, as can be seen in the sample model, some quantities will be known to a high degree of certainty (for example, logPOPEQUIV), while for others an estimate is little more than a guess (for example, DECAY). In general, however, none or almost none of the base quantities in the model will actually be known.

The collection of data must therefore first concentrate on the identification of an estimate for every base quantity, then some thought must be given to the accuracy of these estimates. ABLE itself analyses the input cost model to identify the base quantities about which prior data are required and elicits information about their accuracy. At a later stage, ABLE can be used to identify where studies of base quantities to obtain more accurate estimates will best be rewarded.

In many cases, data for base quantities will already be available in electronic form. It will often be appropriate to draw prior estimates directly from these sources without worrying about refining the data. ABLE can accept data from files which can readily be formatted from a variety of standard packages, spreadsheets etc; and the software can be re-entered at any time to modify the stored information.

1.3.3 Features of ABLE

ABLE makes no permanent record of derived quantities within the software, these are always recalculated from the base quantities as they are required. This means that new data can be entered at any time; and previously entered updating data or prior data can be revised at any time and the consequences for posterior information recomputed.

The implications of currently stored information can be computed and displayed. In particular, current estimates and standard deviations for any derived quantity can be found. Output, to printer and/or to files, can be customised to achieve a variety of presentation styles.

Once a model is established that accurately reflects the operation of the company, it is possible to study the effects of alternative projected courses of action as “what if” scenarios. Snapshots of the model (or variations on the model) and all its information can be stored at any time for later recall.

Every operation with ABLE that influences the final output of estimates is automatically recorded in a form which allows those final estimates to be audited by the company's certifiers.

1.3.4 Typical usage

Figure 1–1 presents a diagrammatic view of the whole of ABLE. Typical usage of ABLE starts at the top of this diagram. Initial work concentrates on establishing the model and prior data. All of this is brought together in the Create/Edit module, but data must first be prepared in several ways:

- Prior estimates and related data must be prepared in files called AIF (ABLE Input Format) files.
- Accuracies of estimates must be specified using the Elicitation module, resulting in one or more AVE (ABLE Variance Elicitation) files.
- Then these files are input through the Create/Edit module under user control from the keyboard (which is also used to type in the cost model equations and other details as needed).
- The end product of all this activity is an AFS (ABLE Full Specification) file, also known as a *system file.*

As can be seen from the diagram, all other ABLE activity revolves around the system file. The other key modules—the Design, Update and Report modules—all require an AFS file as their prime source of data to work on. A company may have several system files. First, they may have separate models for different parts of the business—such as one for water services and another for sewerage and sewage treatment. Also, for a given model different files may be maintained for alternative scenarios, or contain data at different snapshots in time. In particular, the AFS file containing the complete prior data should be kept separate from that which has been updated with more accurate study data.

!

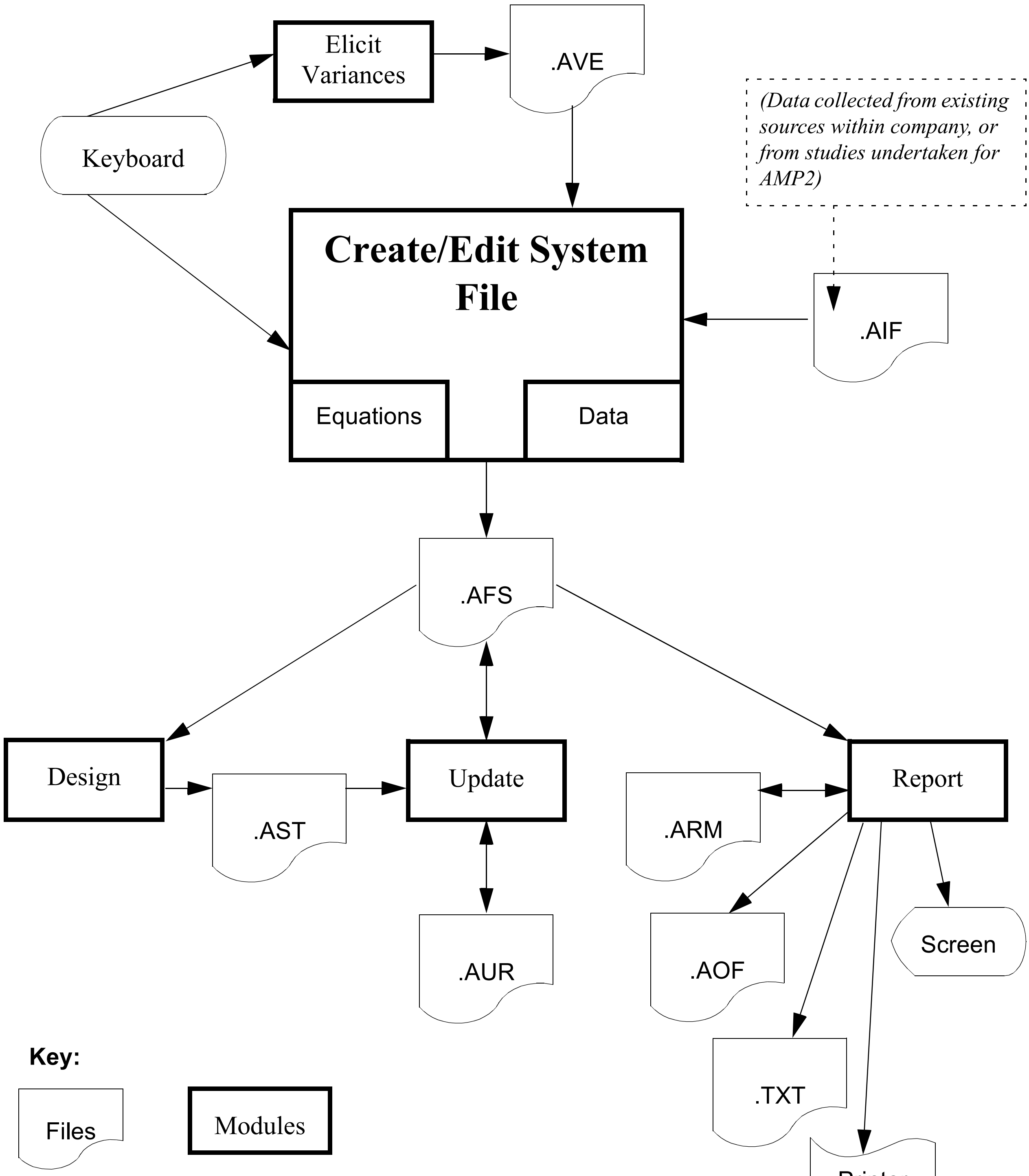
1.3.4.1 Design

Once the prior data are established (or indeed at later stages) the ***Design module*** can be used to devise programmes of detailed study, whose purpose is to obtain improved estimates of base quantities for a sample of assets, and thereby to improve the quality of estimates of key derived quantities. Specifications of different types of study can be saved on an AST (ABLE Study Type) file.

1.3.4.2 Update

The ***Update module*** is used to enter improved estimates of base quantities for individual assets, which may come from studies designed using the Design module, or from other sources. These data may be input from the keyboard, using an AIF file (in the same way as prior data are input), or using another format called an AUR (ABLE Update Record) file. Predefined study types in an AST file can simplify the input of update estimates. A very important feature of the Update module is that it changes the system file. ABLE not only uses the update estimates to replace prior estimates on the file, it also applies the BLE (Bayes Linear Estimator) to deduce statistical implications of the new estimates, and these too are recorded on the system file.

!



File types:
.AFS — ABLE Full Specification
.AIF — ABLE Input File
.AOF — ABLE Output File
.ARM — ABLE Report Macro
.AST — ABLE Study Type
.AUR — ABLE Update Record
.AVE — ABLE Variance Elicitation
.TXT — Text (ASCII)

Figure 1-1: An overview of ABLE

1.3.4.3 Report

The ***Report module*** can be used at any stage to draw out of the AFS file estimates (and accuracies of those estimates) of base and derived quantities. As soon as the AFS file has been established, the Report module can output implied prior estimates and variances for derived quantities of interest. Once the file has been updated with improved study estimates, it can report the implied new estimates of any base and derived quantities. ABLE offers great flexibility of output—to screen, printer, text file or a special form of file called an AOF (ABLE Output Format) file designed to be configurable for further processing. Reporting can be automated by recording macros in an ARM (ABLE Report Macro) file.

1.3.5 Some basic concepts and terms

You will need to be familiar with a number of technical terms and ideas in reading this manual. These and many others are given brief definitions in a glossary at the end of the manual, and terms are generally explained very fully at the most relevant place in the main text. But a general explanation of the most important terms is given here.

(Cost) Model
: The model is a series of equations, expressing derived quantities of interest in terms of base quantities (possibly via intermediate derived quantities).

Derived quantity
: A derived quantity appears on the left hand side of an equation in the model.

Base quantity
: A quantity in the model which is not a derived quantity is a base quantity.

List
: More often than not, a quantity in the model is actually a collection of quantities. For instance, all quantities except MODELTOTAL, DIVCOST, SCHEMES and logSCHEMES in the model of Section 1.2.2 actually take different values for each zone. Technically, we refer to the collection of quantities bearing a single name in the model equations as a *list*. So PRESSURE in the model of Section 1.2.2 is a derived list and PRESCOST is a base list. However, we also refer to these loosely as quantities, in the sense that PRESCOST for instance is a generic base quantity.

ID List
: The individual base quantities covered by the list PRESCOST are pressure unit costs in each of the zones. The list of names of zones constitutes the ID list for PRESCOST. Every quantity has an ID list that identifies for what assets, groups of assets or “zones” that quantity takes different values. The quantity DIVCOST has as its ID list the list of division names. Only MODELTOTAL takes just one value, for the whole company. Its ID list has just one name in it (which is actually blank, because when only one value is involved we do not need names to identify the different values).

Units
: The assets, groups of assets or “zones” named by the ID list are called the units. A list takes a different value, therefore, for each unit.

Group
: Quantities are formed into groups on the basis that quantities in different

groups are uncorrelated. Technically this means that ABLE treats different groups as completely separate and independent. Quantities in the same group, however, are statistically correlated, so ABLE must treat them together. The quantities in a list are *always* assumed to be correlated, so a group is made up of one or more lists.

Variance

Variance is a technical term in statistics. It is a measure of uncertainty, or possible magnitude of error, attaching to an estimate. Its square root is the standard deviation, which is more useful for less technical purposes. As a rough rule of thumb, the true value of a quantity is about twice as likely to be within one standard deviation of its estimate as it is to be outside those bounds. It is crucial in ABLE to ascribe accurate variances to estimates.

Stratification, Scaling,

Systematic and Random Errors

The importance of variance structures has led to development of several important statistical constructs to handle the most common situations. These terms are used, particularly in the Elicitation and Create/Edit modules, when it is necessary to discuss variances. This is generally a technical topic where expert statistical advice is needed, but a general introduction is given in *Chapter 4 — Elicitation.*

Levels of Uncertainty

Most quantities will be capable of having at least two kinds of estimate. Every quantity must be given a prior estimate, and many may be capable of some kind of detailed study to yield an improved estimate. ABLE allows the quantities in a group to have as many different “levels of uncertainty” as appropriate, each denoting a distinctly different kind of estimate which the company could obtain (at least in principle). It may, for instance, be possible to undertake studies at different depths and degrees of detail; or it may be that prior estimates are obtained in different ways. Levels of uncertainty are ordered from most accurate (the best possible study) to least accurate (the crudest prior estimate).

1.3.6 Some useful keys

Each ABLE module has its own purpose and its own character. Special keys are used in the most appropriate way in each case to access the various functions and facilities of that module. However, you will find a generally consistent interface to ABLE facilities is offered, with certain keys always having broadly the same effect throughout the system. (Please see also *Typing Conventions at* the front of this manual.)

F1 **Help**

The F1 key provides brief help information throughout ABLE. Notice also that similar information, of particular relevance to the current screen, is often given on the bottom screen line.

[Ctrl+↵] **Select/Inspect**

(While holding down the Ctrl key, press the “Enter” key.) This key combination is used to “Zoom in” on a particular task. It is often used to select from a menu, where “Zooming in” means starting the activity associated with that menu item. It is also used to “Zoom in” to inspect some data or a file.

[Shift+↵] **Conclude**
(While holding down the Shift key, press the "Enter" key.) This key combination is used generally to conclude an activity, accepting any changes or selections which have been made. The result will be to "Zoom back out" or to move on to the next logical activity.

[Shift+Esc] **Abort**
This key combination also concludes an activity, but any changes or selections which have been made are ignored. The activity is therefore aborted.

F2 **Calculate**
The F2 key causes ABLE to perform some calculation, based on the changes or selections currently made.

F3 **Equations**
The F3 key generally opens a window to view the equations of the cost model. This is used when you are required to choose one or more quantities (base or derived) in the model. Pressing F3 shows the equations, whereupon you can move around selecting quantities before pressing [Shift+↵] to conclude (or [Shift+Esc] to abort).

Space The space bar can be used to mark items for selection, for example when moving through the equations of the cost model.

Tab/Btab The Tab key (or [Shift+Tab] for back-tab) is used to move through menu fields displayed on the screen.

1.4 The ABLE modules

1.4.1 Starting ABLE

ABLE version 2 runs in Microsoft Windows 3.1 or Microsoft Windows NT. The installation process is simple and instructions are supplied with the ABLE software. To run ABLE, simply double-click on its icon. After displaying the ABLE greeting, a small window opens for you to enter a password.

Your ABLE system will be configured with three passwords. Depending on which password you enter, ABLE will allow you access to more or fewer of the ABLE features.

Password level—

Basic: If you type the password for a Basic level user, you will be able to use all features of ABLE which do not involve creating new files (except output files) or editing existing files.

Standard: If you type the password for a Standard level user, you will have access to all but a few advanced features of ABLE. You will be able to create and edit all the various types of file used by ABLE.

Expert: If you type the password for an Expert level user, you will have access to all the ABLE features.

Note that having typed a valid password, clicking on the Change button will allow you to change that password before entering ABLE. You should obviously only use this feature with the approval of your system administrator.

On clicking the OK button, you will see the main ABLE window. The elements of this window are fully described in Section 1.5. Before studying those details it is helpful to have an idea of what the main *modules* of ABLE do.

If you click on the View menu item, a menu drops down with the following six items at the top:

Equation Editor
Group Data
Mappings
Reports
Update Module
Design Module

The first three items relate to the building and editing of ABLE applications, which are stored in files called AFS (ABLE Full Specification) files. This is the concern of the Create/Edit module. The other View items provide entry to three more of ABLE's main modules, the Report, Update and Design modules. The fifth main module, the Elicitation module, is selected automatically when you load a file of the appropriate type.

The actions of, and interactions between these modules, and the files which facilitate those actions/interactions, are shown in Figure 1-1. We now discuss the role of each briefly. Each module has its own chapter in this manual for detailed exposition of its features.

1.4.2 Elicit Variances

In order to combine estimates of Base Quantities correctly, ABLE needs to know how accurate each is, and any statistical relationships that exist between estimates. Formally, these are expressed as variances and covariances. Informally, covariances are concerned with correlations between estimates, which arise when two or more estimates are affected by a common estimating error. Such an error is called a systematic error. Variances describe the possible magnitude of errors in individual estimates, due to both systematic and random errors. The elicitation module allows you to specify the nature of systematic errors (primarily through grouping and stratifying quantities), and to specify the likely magnitude of each systematic and random error.

Some discussion of the statistical methodology that lies behind this module can be found in *Chapter 4 — Elicitation*.

The details collated by the Elicit Variances option are stored in an ABLE Variance Elicitation (AVE) file.

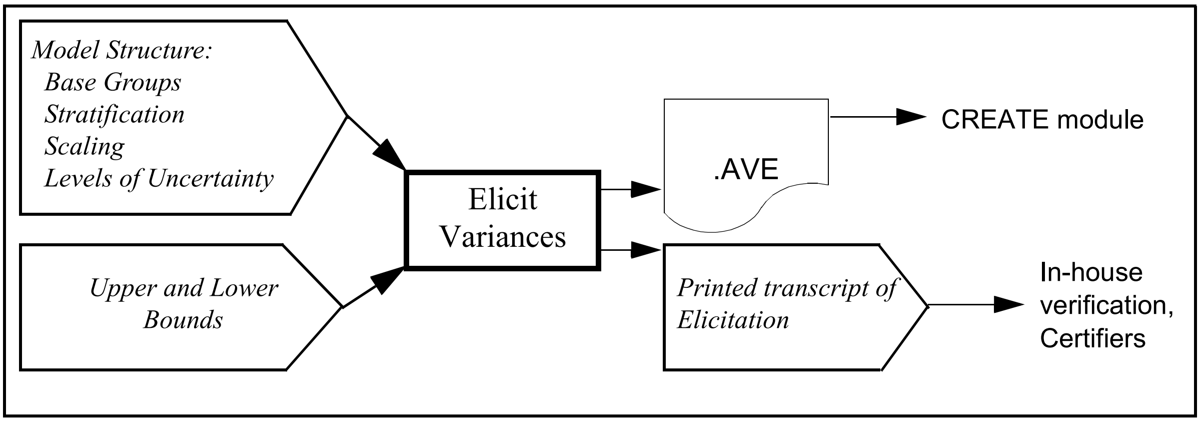


Figure 1-2: Elicit Variances

1.4.3 Create/Edit system file

ABLE maintains a full specification of all the data it needs for its main calculation functions (Design, Update and Report, see below) in a single file known as an ABLE Full Specification (AFS) file. The Create/Edit System File selection enables you to create a new AFS file or edit an existing file.

An AFS file contains a complex system of information built up from a variety of sources typically:

- keyboard input
- variance information from the AVE file set up in the elicitation module)
- data in an externally created ABLE Input Format (AIF) file (used to import data from spreadsheets, databases and editors; see *Chapter 2 — Input and output formats*)

and less typically

- data for nonstandard cases can be set up by advanced users by accessing the underlying APL language.

Data in an AFS file includes

- definition of the basic model: *Equations*

- specification of which quantities in the model are correlated (i.e. their estimates are affected by common systematic errors): *Groups*
- names for the basic units or assets on which the quantities are defined: *ID Lists*
- information about characteristics of units that allow them to be grouped or stratified
- available types of estimate for each quantity: *Levels of Uncertainty*
- current estimates and their types
- variances and covariances for all types of estimate.

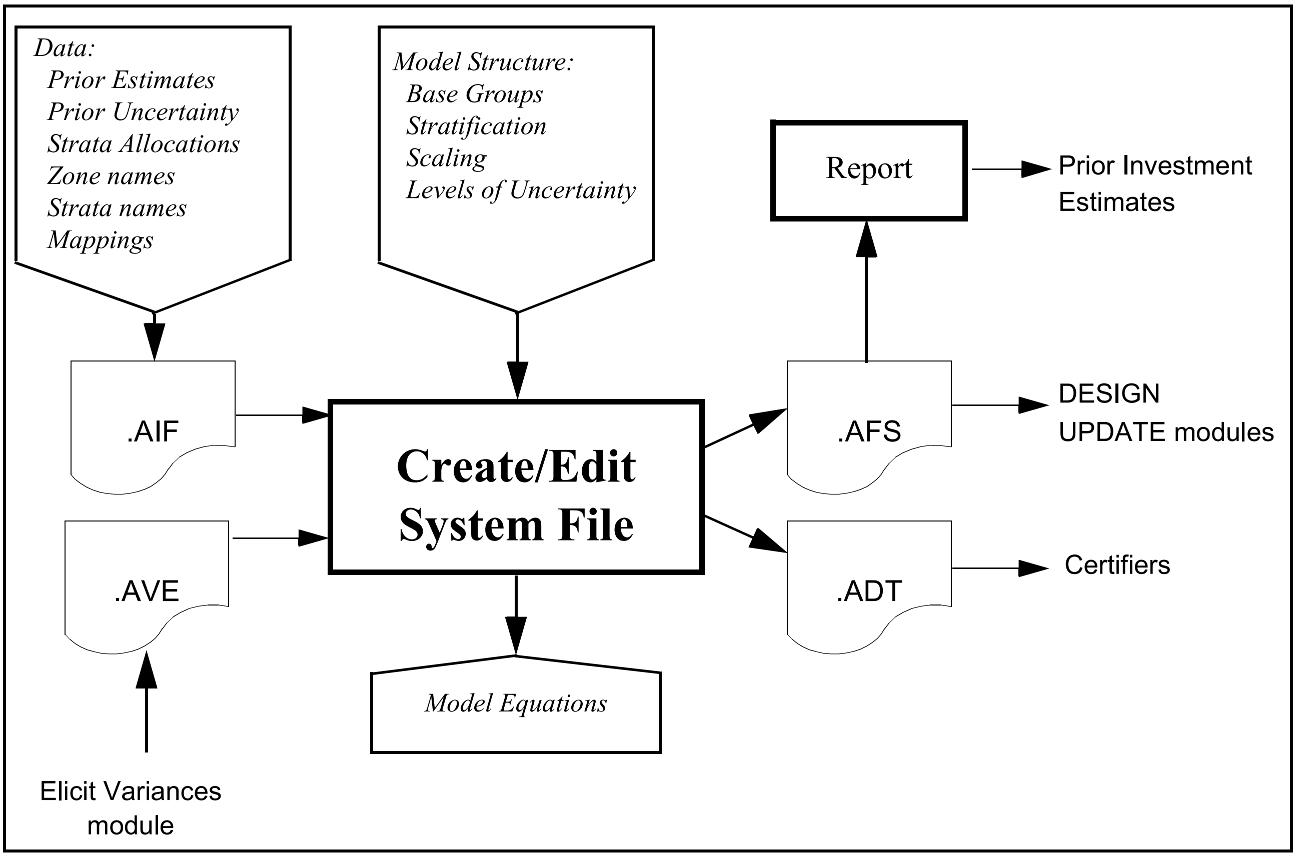


Figure 1-3: Create/Edit System File

1.4.4 Design studies

This option is used to manipulate and combine the data collected so far in order to identify what new information to gather to improve the accuracy of key estimates. The result is a programme of studies to be performed. A particular kind of study is defined as obtaining new estimates, at specified levels of accuracy, on some or all of the base quantities which are defined on a list of units. A simple study programme might then be set up by saying how many units to study, and ABLE will select the sample of that number of units which will be most informative about a given quantity of interest. More complex programmes can be designed forcing particular studies (units) to be included or excluded.

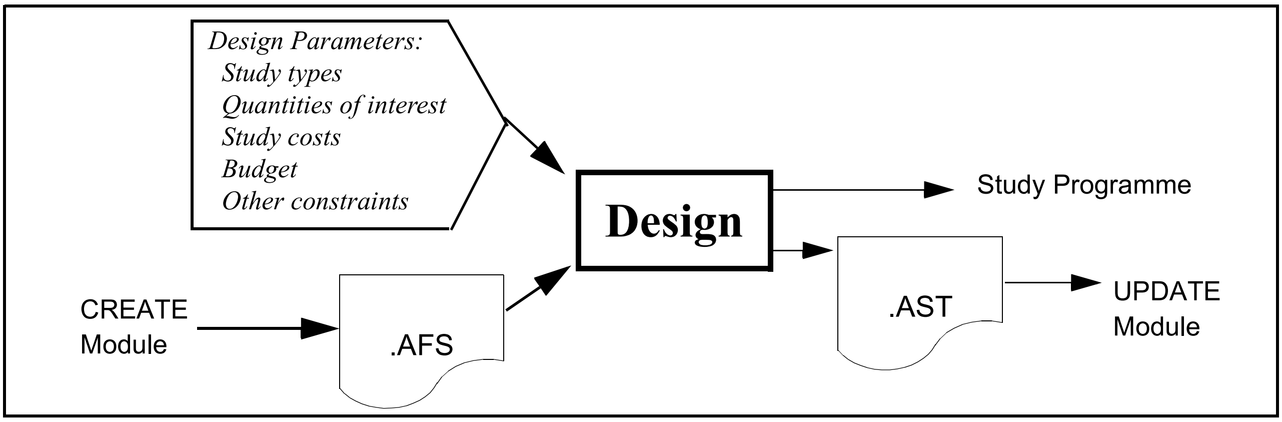


Figure 1-4: Design Studies

1.4.5 Update

Designed programmes of studies and other sources will from time to time generate new data in the form of more accurate estimates of base quantities. A primary function of ABLE is to combine this new data with existing information and update the base of knowledge in the AFS file. The mechanism used is the Bayes Linear Estimator (BLE, from which the name ABLE derives).

A feature of ABLE is the ability to backtrack and replay updates after changing some data on the original AFS file or on the update file. This is achieved by saving an ABLE Update Record (AUR) file, which gives details of individual updates or a series of updates.

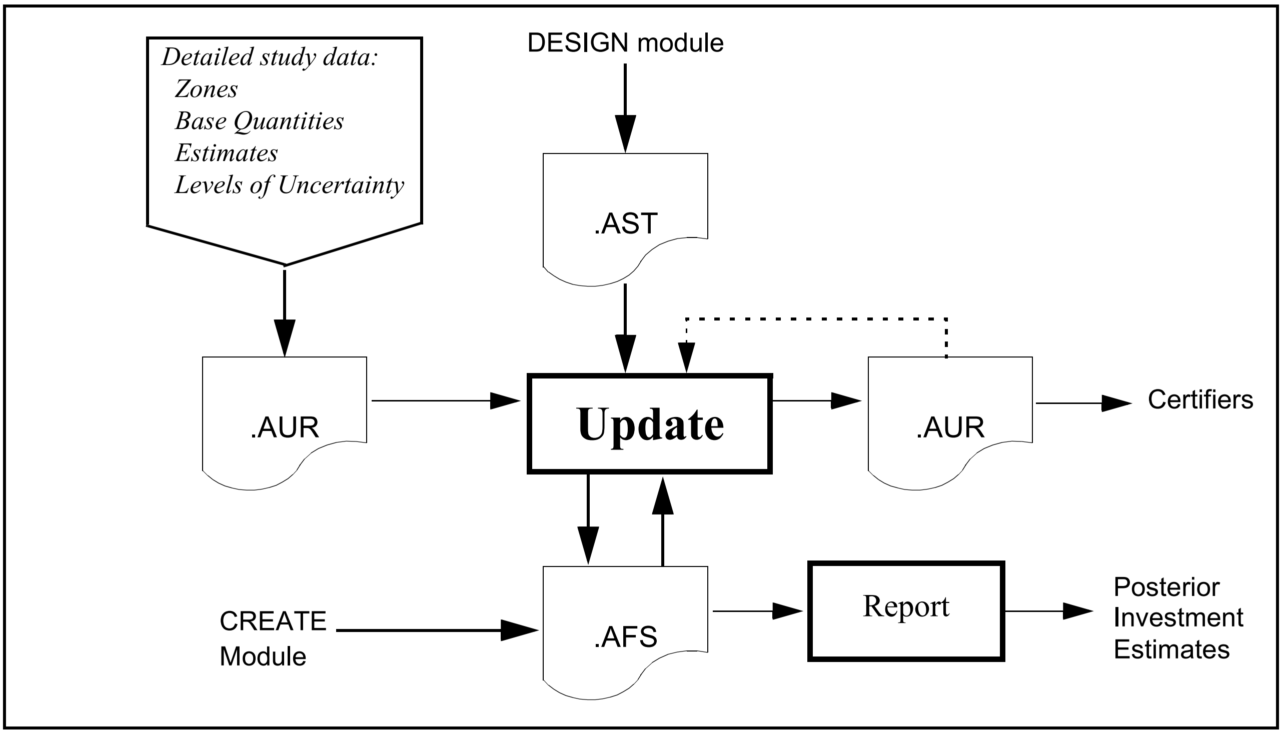


Figure 1-5: Update

1.4.6 Report derived estimates

This option gives you access to an AFS file to report information contained or implied within it. The latter feature—implied information—is most important. The AFS file records information on the base quantities which lie at the lowest level of the system of equations defining the model. Interest more often centres on the higher quantities, called Derived Quantities. The purpose of the model is to structure the derived quantities of interest to you in terms of more elementary base quantities which can be estimated individually. So the Report Derived Estimates option allows calculation of estimates and variances for any quantities in the model, Derived or Base. These may be reported in a variety of forms: on screen, to a file or in hard copy. A macro definition facility allows a predefined series of reports to be set up and saved, which can then quickly be called up and used whenever required.

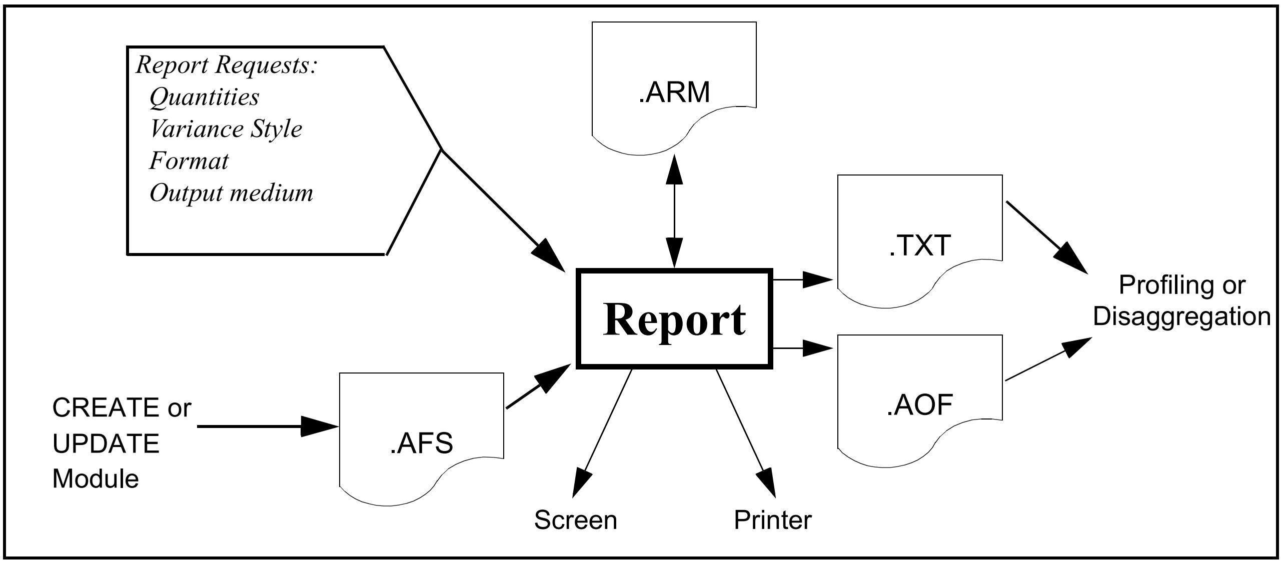


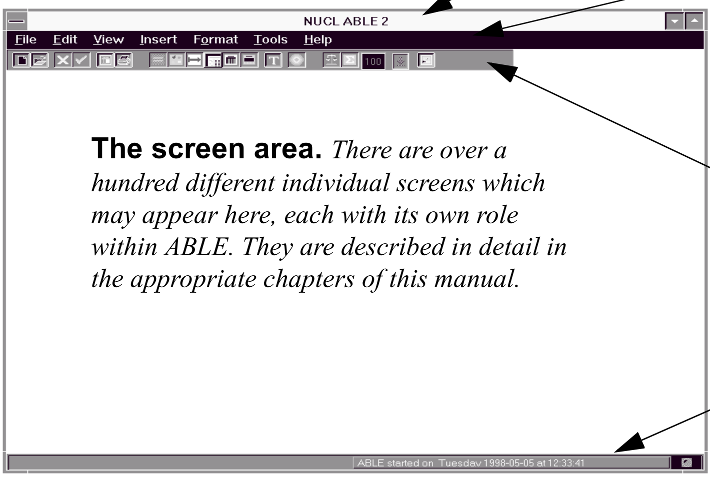
Figure 1-6: Report Derived Estimates

1.5 The main ABLE window

After entering your password as described in Section 1.4.1, ABLE presents the main window. It has the following elements.

Title Bar. *This shows the ABLE icon and name. In addition, when a file is open for use its name and path will be displayed here.*

Menu bar. *The ABLE menus—File, Edit, View, Insert, Format, Tools and Help—are opened in the usual way by clicking on the name or using keyboard short-cuts (e.g.* [Alt+F] *to open the File menu). The contents of each menu are described below.*



The screen area. *There are over a hundred different individual screens which may appear here, each with its own role within ABLE. They are described in detail in the appropriate chapters of this manual.*

Toolbar. *The buttons on the toolbar perform various functions, allowing alternative ways to access various items in the menus.*

The status bar. *This bottom line is used to provide helpful information. Two items are constantly viewed: the start time and the* ***level indicator****. The latter is a coloured circle which is green if your status is that of an Expert level user, amber if you are a Standard level user, and red for a Basic level user.*

1.5.1 The File menu

The file menu offers the following items.

Item	Alternative method of selection	Description
New	[Ctrl+N] or click the New button on the left of the toolbar.	This creates a new file. There are two kinds of file which can be open for use in ABLE, AFS files and AVE files. When you select New you will be presented with a standard file selection dialogue. In the bottom left, the area Save file as type: allows you to select AFS or AVE file. If you select an AVE file, ABLE will automatically open the Elicitation module, which is the only module relevant for operating on AVE files. Otherwise, ABLE will expect you to begin operating on the new AFS file with the Equation Editor, since defining the equations necessarily precedes any other activity.

Item	Alternative method of selection	Description
Open	[Ctrl+O] or click the Open button, second from the left on the toolbar.	This opens an existing AFS or AVE file. Opening an AVE file will cause ABLE to begin the Elicitation module to operate on that file. If you open an AFS file ABLE will begin whichever module is currently selected from the top six items in the View menu (see Section 1.5.3).
Notepad		The Windows standard Notepad program is a convenient tool to view various kinds of ABLE files that are basically text files. Selecting this menu item opens a file dialogue again, but this time the option List files of type: offers selection of AIF, AOF, AUR, ADT or TXT files. When you select a file, Notepad is opened to view that file.
Close/ Abort	[Shift+Esc] or click the Close/Abort button on the toolbar, which appears as a red cross.	This action is used throughout ABLE to end some operation *without* saving the results. It may close the active file, or may terminate some more specific operation.
Save/OK	[Shift+↵] or click the Save/OK button on the toolbar, which appears as a green tick.	This action is used throughout ABLE to end some operation *and* to save the results. It may save and close the active file, or may successfully terminate some more specific operation.
Manage	click the File Management button, between the Save/OK and Print buttons on the toolbar.	This opens the ABLE File Management module, whose purpose is to help you to manage the many kinds of files produced by ABLE. This option is available only when no file is currently active in ABLE. This operation is described fully in *Chapter 9 — File Management*.
Printer Setup		This opens the standard Windows printer setup dialogue for your default printer.
Print	F8 or click the Print button on the toolbar.	This sends the current file view to your default printer. Printing is implemented in only a few of the ABLE screens.
Recently used files		The five most recently used files can be reopened quickly by selecting from this list.
Exit		This closes ABLE.

1.5.2 The Edit menu

Item	Alternative method of selection	Description
Copy	F5	This has a specific role in ABLE, typically to copy the current column of data. It is not like the usual Windows Copy operation, and does not use the Windows Clipboard.
Delete item	[Ctrl+Delete]	In appropriate places in ABLE, these have specified actions.
Select item	Space	
Modify item	F7	
Change initialisation		Configuration actions when ABLE is started are primarily controlled by the ABLE.INI file. Selecting this menu item opens ABLE.INI in the Windows Notepad program, so that you may edit it.
Paste		Enables equations to be pasted from the clipboard into the equations editor of the Create/Edit module. It makes building large models easier because these can be constructed in an ordinary text editor, using all the cut/paste/copy features, and then copied to the clipboard and pasted into ABLE.
Back		When building or editing the information about a group in the Create/Edit module, you are expected to progress through the various screens in strict order. However, the Back feature allows you to step back through the screens. This is useful when you wish to correct something in an earlier screen without aborting the group and starting again.
Change keyboard mapping		Throughout this manual, there are many places where certain keypresses or key combinations are described as having given actions. It is actually possible to redefine keys to have different effects, by editing the keyboard translate table. Selecting this menu item opens the relevant file for editing. This should not be done lightly! It may have unpredictable consequences and can certainly confuse other users of your ABLE system.

!

1.5.3 The View menu

Item	Alternative method of selection	Description
Equation Editor, Group Data, Mappings, Reports, Update module, Design Module	A row of six buttons on the toolbar, starting with the Equation Editor button which is an equals sign, provide alternative ways to select the AFS file view.	The first six items of the View menu determine which of six “views” is used when the currently active file is an AFS file. The first three—Equation Editor, Group Data and Mappings—are different views in the Create/Edit module, concerned with editing each of the three main constituents for an AFS file. The other three provide views into the other three main modules which operate on AFS files. When an AFS file is active, selecting any of these items changes the view, and you will see a warning that any changes may be lost unless you have saved them. When no AFS file is active, selecting one of these items changes the view which will be used the next time an AFS file is opened.
Totals	The Totals button on the toolbar is a letter T.	This affects whether the Report module includes totals in reports. Selecting this item toggles totals on or off.
List/Next	F6	This item has a specific role in certain ABLE modules.
Zoom in	[Ctrl+↵] or the Zoom button on the toolbar.	There are many places in ABLE where Zoom will present you with a more detailed screen.

1.5.4 The Insert menu

This menu allows insertion of items of various kinds.

Item	Alternative method of selection	Description
Equation Quantity	F3	This opens a view of the model equations to allow a base or derived quantity to be selected for insertion.
Insert Item	[Ctrl+Insert]	This opens a space to insert a new item where appropriate.
ID List	F4	This opens a view of the ID lists already defined in the AFS file, to select one for insertion/use elsewhere.
Simple Default		These items reset the random or systematic error screens to default values.
Prior Setting		
Update Record	F10	This imports an ABLE AUR file for use in the Update module.

Item	Alternative method of selection	Description
Import AVE	F9	These menu items open views of the contents of an AVE or AIF file, respectively, to select items for insertion.
Import AIF	F11	

1.5.5 The Format menu

Item	Alternative method of selection	Description
Font		This opens a standard font selection dialogue. Changing the default font here affects the font used in certain parts of ABLE. It does not affect the font used in most ABLE screens, but does have an effect in the newer fully-Windows style screens.
Toggle strata		In the Group Data section of the Create/Edit module, this item toggles the table in the Random Error Variances screen, between having a single row (implying having the same variances for all units) or one row for each stratum (implying variances dependent on which stratum a unit is in).
Autoexpand grid		In some of the new fully-Windows style screens, the data are shown in a grid of numbers. Typing in the cells of the grid can produce numbers which do not fit the cell. Selecting this item will expand the cells to fit the numbers.

1.5.6 The Tools menu

Item	Alternative method of selection	Description
Calculate/ Execute	F2 or the Calculate button on the toolbar.	This menu item initiates a calculation by ABLE. Important calculations initiated in this way are the incorporation of update information using Bayes Linear Estimation, the generation of reports, computation of programmes of studies, analysis of model equations, and analysis of elicitation bounds.
Calculation Mode	The Calculation Mode button on the toolbar, which is a Greek letter Σ, toggles the calculation mode between exact and simulation, and when simulation is selected the sample size may be set in the adjoining edit field.	When generating reports, the Report module may use Exact or Simulation-based computations. Simulation-based computation also requires the specification of sample size. Selecting this item from the Tools menu opens a small window to specify these options.
Pocket Calculator		This opens the standard Windows Calculator program, so that you can do calculations while running ABLE.
Configure	The Configure button on the far right of the toolbar.	This item opens the ABLE Configuration module, where certain options can be set. In particular, this module allows you to change user level. See *Chapter 3 — Configuration*. This option is available only when no AFS or AVE file is active.
APL Session	F12	Expert users who are running ABLE in the full version of Dyalog APL (as opposed to the run-time version supplied as standard with ABLE) may use this option to open the underlying APL session.
Diagnostic Log		Any errors which occur while ABLE is running are trapped and logged. This menu item allows you to view and edit the diagnostic log. If errors occur which you suspect are caused by bugs in ABLE, sending the diagnostic log file to NUCL together with a full explanation of the circumstances under which the error occurred, may assist in identifying and correcting bugs.
Macro	[Ctrl+↵]. Macro is equivalent to Zoom, so the Zoom button on the toolbar is another alternative.	In the Report module, this selection opens a macro file, to edit or to use to generate a predetermined set of reports.

1.5.7 The Help menu

Item	Alternative method of selection	Description
Context Help	F1	Help is available through this option, particularly in the Create/Edit and Elicitation modules. The help is primarily concerned with specifying which key presses and combinations are available, and their meanings. In other modules, this information is less bulky and is often given on the bottom of the screen.
Greeting		Selecting this option merely displays the ABLE greeting.
About NUCL ABLE 2		This produces a standard Windows Help>About… window. Additionally, it gives information about the available memory and Windows resources.

1.5.8 Working in ABLE screens

ABLE version 1 was a DOS program. In converting it to run in Windows, no attempt has been made to perform the comprehensive rewriting necessary to exploit the full Graphical User Interface capabilities of Windows. So, although ABLE version 2 has a menu bar and toolbar offering a range of standard Windows features, working within the ABLE screens is primarily in the version 1, keyboard-based way.

! You should not expect that the mouse will do any important operations within the screen area of the ABLE window. Screens are navigated by the keyboard cursor, tab and back-tab (i.e. [Shift+Tab]) keys. Many of these keys have menu items or toolbar buttons as alternatives, such as—

- [Ctrl+↵] = Zoom, or the Zoom button,
- [Shift+↵] = Save/OK, or the Save/OK (green tick) button,
- [Shift+Esc] = Close/Abort or the Close/Abort (red cross) button.

However, the regular user will probably find that ABLE is easy to use predominantly with the keyboard.

! Users familiar with ABLE version 1 will find that one of the most used keys has changed. Version 1 used the Escape key, [Esc], for Save/OK. This was thought to be confusing and quite different from the usual meaning of Escape in other software. This has been changed so that Save/OK equates to [Shift+↵] in version 2. To avoid version 1 users inadvertently destroying data, Escape has not been given its more familiar role of Close/Abort, but instead does nothing. (Anyone wedded to the version 1 use of Escape can remap the keyboard via the Edit>Change keyboard mapping menu item.)

ABLE:

Chapter 2 — Input and output formats

2.1 Introduction

This chapter concerns the three ABLE file structures that you will need to understand. These are the AIF, AUR and AOF file types. ABLE operates with several other kinds of files, notably the AFS, AVE, AST, ARM, ACO and AFD types, but only for its own internal purposes. You need to know what kind of information ABLE is using these files for, and that is explained at relevant points in the manual, but you do not need to know details of how that information is stored in those file types. (In fact, all those files are “APL Component Files”, and cannot be read without using appropriate commands in the Dyalog APL language.) One other file type is a readable ASCII file—the ADT file. This is intended for use by the company certifiers, and is explained in *Chapter 10 — Auditing*.

AIF files are the principal way of inputting quantities of data to ABLE. They are particularly useful when setting up or modifying a system file with the Create/Edit module. The prior estimates will typically be provided on an AIF file, as will information about stratifications and various other parts of the model specification. They may also be used to supply data on improved estimates from detailed studies for the Update module, although such data will not necessarily be so voluminous.

AUR files are another way of supplying detailed study data for the Update module. That module is designed to read and process AUR files in an automated way, so minimising the need for an operator to set up the updates. The Update module also creates an AUR file as a record of updates that have been carried out. The output file can also be used as an input file, and is also part of the auditing process, so even if you do not use AUR files to prepare your detailed study data for input to ABLE, it is useful to know about the AUR format.

Output from ABLE, in the form of estimates and variances for quantities, can be obtained in a format designed to be suitable for input to other computer systems for further processing (for example profiling, tabulation or other kinds of analysis). The relevant file type is the AOF file, which is designed to be flexible enough for you to obtain output in the form acceptable to your further processing software.

2.2 ABLE Input Format (AIF) files

This Section sets out the format for the primary kind of datafiles to be input to the ABLE program. It is designed to be very flexible and allow a variety of data to be transferred directly to ABLE from some other computer system, or to be output from some other software package. It is particularly intended for input of prior estimates of base quantities, which constitute large quantities of data that it would be inefficient and error-prone to type into ABLE by hand. These data will be collected by water companies or their agents in their own computer systems, typically using standard databases or spreadsheets. It is generally straightforward to require such programs to output selected data in the form of a simple ASCII file. If necessary, such a file can readily be edited using any standard text editor to produce a file of the format described here. Then the data can be picked up directly by ABLE on the same computer, or transferred to ABLE on another computer (for example, via floppy disk).

The file format described here is called ABLE Input Format or AIF. Names of AIF files appear with the extension `.AIF` at the end. AIF files are ASCII files, which we can think of as text consisting of lines on a page, each line terminating with a newline character. Sections 2.2.1 and 2.2.2 provide a formal specification of this format. Sections 2.2.3–2.2.5 explain how AIF files should be used in ABLE.

2.2.1 Data items and separators

An AIF file consists of a series of ***items*** delimited by ***separators****.*

Items may be numbers or alphanumeric ***words***.

A separator is a space, a comma, a newline character, a colon or a > character. Two or more separators together (for example, two or more spaces, a comma and a space, a space and a newline) are treated as a single separator but *no such sequence may contain both a colon and a > character.* There are four kinds of separator:

- A ***Blank*** separator is a space or a comma or any sequence of these.
- A ***Newline*** separator is a single newline character or any sequence of spaces, commas or newlines containing at least one newline character.
- A ***Colon*** separator is a single colon (:) or any sequence of separators containing at least one colon.
- A ***Block*** separator is a single > character or any sequence of separators containing at least one >.

Separators at the beginning or end of the file are ignored.

! Obviously no item can contain a separator, so in particular *numbers must not contain commas and words must not contain spaces*.

2.2.2 Blocks of data

Block separators divide the file into one or more blocks of data. (Since separators at the end are ignored, it is not necessary to end the whole file with a Block separator.) A block may be either a ***Vector block*** or a ***Matrix block****,* and blocks of both types may appear in a single AIF file.

2.2.2.1 Vector blocks

The first item in a Vector block is the ***identifier*** for this vector, which is followed by a Colon separator. The remaining items of the block make up the data for the vector, and may be separated by any combination of Blank separators or Newline separators. The identifier is the name by which this vector of data is referred to when inputting it to the ABLE program. When seen as text on a page, a Vector block just looks like a name followed by a colon followed by a string of items separated by spaces and commas, typically taking up several lines in all (and ending with a > if another block follows).

2.2.2.2 Matrix blocks

A Matrix block begins with a series of ***identifiers*** separated by Blank separators and ending with a Colon separator. The number of identifiers is the number of vectors being defined and will be called the ***Count***. Following the Colon separator come a series of ***lines,*** each line having Count items separated by Blank separators and ending with a Newline separator. (The last line, however, ends instead with the Block separator if another block follows, or may end with no separator if this is the last block in the file.) When seen as text on a page, a Matrix block appears simply as a table, in which each column is a vector, with the identifiers for the vectors appearing at the tops of the columns.

For clarity, when an AIF file is read as text on a page (for example, when editing with a standard text editor), it is recommended that a Block separator includes a newline character both before and after the > character, so that the latter is set out clearly on a line of its own. It is also recommended that in the Colon separator in a Matrix block the colon is followed by a newline character, to ensure that the identifiers appear at the tops of the columns.

According to the above specification, a Matrix block with a Count of one is indistinguishable from a Vector block, since it begins with just a single identifier. There is no difficulty with this because both define just a single vector. The only difference is that the Vector block form allows items to be separated with Blank or Newline separators, whereas in a Matrix block with a Count of one all items would have to be separated by Newline separators. ABLE will always read a block with a single identifier as a Vector block, and so will correctly input the data in either case.

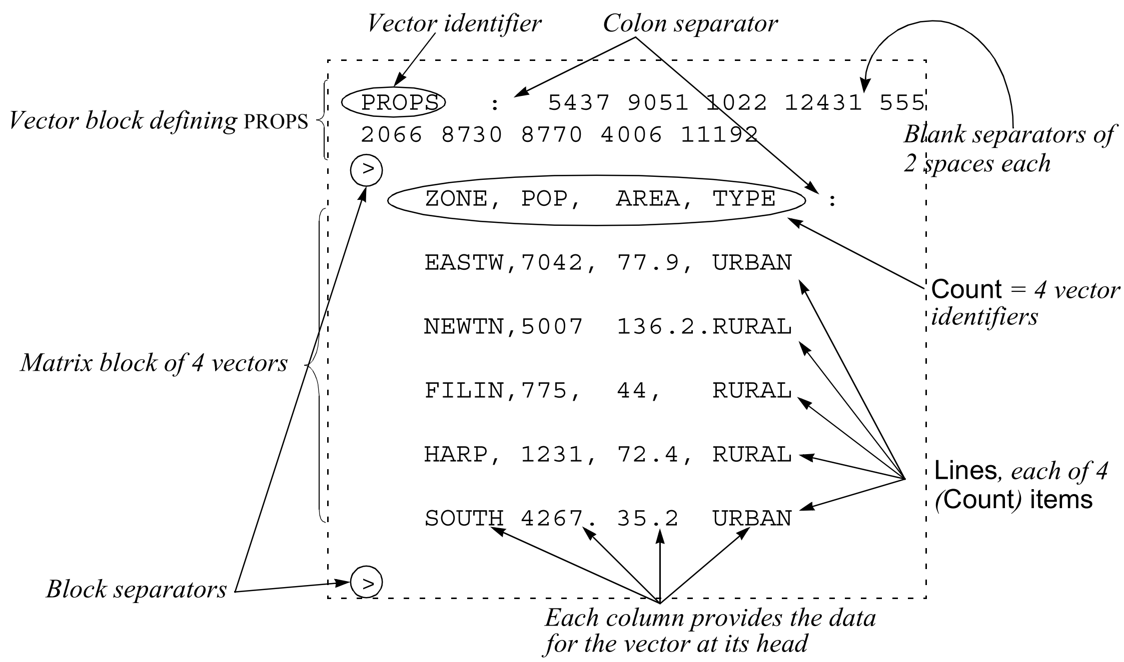
! **Note**: All identifiers in a file must be distinct.

2.2.3 Examples

2.2.3.1 Data structure

In the following file a Vector block defines the vector to be known as `PROPS`, then a Matrix block defines the vectors called `ZONE`, `POP`, `AREA` and `TYPE`. The `PROPS` vector has ten items whereas the other vectors have five items each. (Obviously the numbers of items in a real-life example would be much larger.) The > is shown at the end of the second block, but is not needed if this is the end of the file. The same dataset is shown twice, the second time with annotations.

```
PROPS:   5437 9051 1022 12431
    555
2066 8730 8770 4006 11192
>
    ZONE, POP,  AREA, TYPE:
    EASTW,7042, 77.9, URBAN
    NEWTN,5007  136.2.RURAL
    FILIN,775,  44,   RURAL
    HARP, 1231, 72.4, RURAL
    SOUTH 4267. 35.2  URBAN
```



2.2.3.2 Data interpretation

Figure 2-1 provides a different sample of data with an accompanying explanation giving an indication of how the data have been derived, or from where they might have been obtained. Comments are included delineated by // characters (see Section 2.2.5.2)

Figure 2-1: Data with accompanying interpretation

`// SOME LEVELS OF UNCERTAINTY //`
 	
`M_LENGTH_LEV: CALIB MAP RECORDS>`
`BURST_LEV: STUDY PRIOR  >`
	The mains length in a zone can be estimated in three different ways. RECORDS *denotes just looking up old records claiming to say the length of mains.* MAP *denotes an estimate from a digital map.* CALIB *denotes an estimate in which the map is also checked by a calibrated hydraulic model. The last two are used for prior estimates, and the first two may be used for subsequent improved estiamies. The vector* FROM_MAP *in the first block will be used by the operator to assign the prior level of uncertainty for the estimates in the vector* MAINS_LENGTH. *It is not necessary that the things in the* FROM_MAP *vector are the actual names* MAP *and* RECORDS*—*Y *and* N *can be made to correspond to these by the operator when inputting the data.*

`// SECOND BLOCK OF ZONE DATA //`
 	
`ZONE_A PUMP_STNS SERV_RES :`
`N1     5         1`
`// The block goes on and ends with`
 	
`W19    0         2`
`>`
	This is another block of zone data. Note that the first column is not labelled ZONE *but* ZONE_A. *Identifiers in a file must all be different.* ZONE_A *could be the same as* ZONE, *but has to have a new name. It is nevertheless worth having this column repeated for two reasons. First, it is easier to check the file later. Second, ABLE can check for rows being out of sequence in one block compared with another.*

	Some final remarks. *1. This is a rather bitty file, with big matrix blocks of data, which in practice are likely to be drawn from a database or spreadsheet, together with little vector and matrix blocks of more structural information. It may be sensible to have two files, one with the hefty data, created directly by downloading from a database or spreadsheet, the other with the little things set up by hand.* *2. It is sensible to have files that can fit on a page for printing out like this one or for viewing with a text editor. That means not having too many columns in your matrix blocks. Even if you have 100 items of data to set out for each zone, and could do that in a big 100-column matrix, don't. It is generally clearer to split it into smaller blocks, with 6 to 10 columns.*

Figure 2-1 (cont): Data with accompanying interpretation

```
// FIRST BLOCK OF ZONE DATA //
ZONE MAINS_LENGTH MAINS_CONDIT FROM_MAP LOW_PRESSURE DENSITY
      BURSTS:
NI    132.4        GOOD          Y         15         URBAN
      LOW
N2    17.5         FAIR          Y         0          SUBURB
      LOW
N3    26.0         GOOD          N         0          RURAL
      LOW
// Then a lot more lines for other zones ...
W17   2.6          POOR          N         153        SUBURB
      HIGH
WI8   24.3         GOOD          Y         77         URBAN
      MED
W19   0.2          GOOD          N         0          RURAL
      MED
>
// Note the > at the end of the block.
```

This block is being used to carry a variety of data. MAINS_LENGTH *is obviously a length of mains in each zone.* MAINS_CONDIT *is grading the overall condition of mains in the zone by* GOOD, FAIR *or* POOR. *(The company has no better prior information about condition than this subjective grading.)* FROM_MAP *is saying whether* MAINS_LENGTH *has been obtained from digital maps or not. That will affect the quality of the prior estimate and will correspond to assigning different levels of uncertainty to the prior* MAINS_LENGTH *estimates.* LOW_PRESSURE *is numbers of properties with low pressure, prior estimates again.* DENSITY *classifies the zone as* URBAN, SUBURB *or* RURAL. BURSTS *is another rough grading of whether the bursts record in the zone is* LOW, MED *or* HIGH.

```
//FIRST SET OF STRATIFICATION VECTORS //

M_COND: GOOD, FAIR, POOR >
    // This is a simple vector block
U_S_R: URBAN, SUBURBAN, RURAL >
    // Note comma separators are same as spaces
```

The identifiers for these are M_COND, *and* U_S_R. *All identifiers in the file must be different so we could not use* MAINS_CONDIT *or* DENSITY *as identifiers again. Those are identifiers for the vectors saying which zones fall in those strata. We need other identifiers for the lists of stratum names. When extracting the data from the file, we can use these for two purposes. The first is to check that the vectors* MAINS_CONDIT *etc in the first block are correct. Otherwise we might have typed* GOOF *instead of* GOOD, *and ABLE would not know that there was not a fourth stratum called* GOOF. *The other purpose is to tell ABLE in which order to arrange the strata internally and in reporting: mains conditions are in the order* GOOD, FAIR, POOR, *not* POOR, FAIR, GOOD *or any other way.*

Such small amounts of information need not actually be on the file. They can be typed in by the operator directly to ABLE, but it is generally good practice to put them in the file for later checking and auditing purposes.

```
BURST STRATA AND PRIOR ESTIMATES //

BURST BURST_EST  :
LOW   .01
MED   .025
```

This little block is giving the list of names of the strata for BURSTS, *and also assigning prior estimates to those strata. Thus, there is an equation in the cost model saying that the length of mains to be replaced because of poor burst record is the total length of mains in a zone (*MAINS_LENGTH*) times a proportion. The* BURST_EST *vector then gives prior estimates of that*

2.2.4 Advice to users

AIF files are used primarily to provide prior estimates of quantities in a Bayes model, plus information on stratification and other relevant characteristics of zones or assets. Data giving many different quantities for each unit (i.e. each of a collection of zones or each of a class of assets) may most conveniently be arranged as a single matrix block as in the second block of the example. In this case it is useful for the first column to be names, as in the example, since that makes it easier to check the data later, and possibly to correct for data elsewhere coming in different orders.

! **But remember** that all identifiers in a file must be different. So if there are two or more blocks of data referring to the same collection of zones or assets, then their first columns should be headed by different identifiers. Then the ordering of the rows in the two blocks could be different and ABLE can detect this, using the name columns to match up the data (by *calibration,* see Chapter 5, Section 5.5.3).

Data giving prior estimates for a single base quantity should make up a vector whose name is the name of that base quantity in this model. The items in such a vector will of course be numbers (prior estimates of that quantity for each unit) but, as the example shows, the items in vectors can be words. The zone names vector is one example, and the vector giving stratification by `URBAN` or `RURAL` is another. These could of course be numbers instead, if you choose to identify zones by number rather than name, or to use numbers like 1 for Urban and 2 for Rural. Or you could shorten Urban to U and Rural to R. But in general it is easier to check data if you spell things out, letting ABLE convert codes to numbers if it needs to.

Another kind of vector will enable assets to be related to zones or things to be grouped. Thus in a table of data for treatment works one column (vector) might identify in which zone the work lies; or in a table of data for water quality zones we might have a column saying which resource zone each quality zone is fed from or falls within. Another kind of vector might identify where some prior estimates are at a different ***level of uncertainty*** than others, for example which zones have been electronically mapped.

2.2.5 Extra options

2.2.5.1 Alternative separators

An open parenthesis character, (, is equivalent to a colon, and a close parenthesis character,), is equivalent to a >. These are alternative separators and therefore cannot appear in any item.

2.2.5.2 Comments

A double slash (i.e. two consecutive / characters) appearing anywhere in the file marks the start of a comment. The double slash and everything after it, up to but not including the next Newline separator, is ignored.

2.3 ABLE Update Record (AUR) files

After gathering prior information, a company will carry out a programme of detailed studies (possibly with the aid of the ABLE Design module) to obtain more accurate data. The Update module is where you input these improved estimates, which can be supplied in AUR format.

An AUR file is also created by ABLE for output from the Update module; such a file can be used to replay all the updates that have been applied to date, by itself becoming an input file to a saved earlier copy of the model. Thus an AUR file provides an essential record of the transformations an AFS file has undergone.

The use of such files is explained in detail in *Chapter 7 — Updating*— and in *Chapter 10 — Auditing*. This section is concerned only with the format of AUR data.

2.3.1 AUR format

An AUR file is an ASCII file. Comments are allowed, introduced by two slash symbols: //. Everything from (and including) the // to the end of that line is ignored. Blank lines are also ignored.

Comments are written to the AUR file created by ABLE to track the course of updates and to remark where data has been ignored (because, for example, a level of uncertainty has been specified that is the same or worse than the current level).

Each item of data has four components:

1. The estimate itself
2. The base quantity being estimated
3. The ID of the unit studied
4. The level of uncertainty of the estimates.

Data items are supplied one item per line. All four components are supplied on a line, separated by one or more spaces. The estimate is, of course, a number but the other three components are words.

2.3.2 Example of an AUR Input File

The file `NEWDEMO1.AUR` supplied with ABLE shows a simple AUR file intended for input of data to update the `NEWDEMO.AFS` file.

! **Notice that** AUR input files should not have the same base name as any AFS file, because ABLE uses *AFS-filename*`.AUR` as the output file to record updates.

Hence the use of the name `NEWDEMO1.AUR` instead of `NEWDEMO.AUR` for this file. (`NEWDEMO.AUR` will be created the first time that `NEWDEMO.AFS` is updated.) Figure 2-2 provides a listing of the file, and shows seven update items. These are the results of four detailed pressure studies, although the study in zone `W7` is incomplete because the estimate of `PRESCOST` is not yet available. The order of items on an AUR input file is not significant. The items have been grouped by base quantity, but could have been grouped by zone or in a random order. Spelling of words must agree with the spellings in the AFS file (which were established using the Create/Edit module), including capital letters. The names of base quantities in this example are all in capitals (which is how the equations were originally typed), but names for levels of uncertainty use both upper and lower case.

```
// NEWDEMO1.AUR Demonstration AUR input
      file

// First results of Detailed Pressure
      studies

6.746 logLOWPROPS  N2  Detail
6.205 logLOWPROPS  MS2  Detail
6.492 logLOWPROPS  W7  Detail
4.745 logLOWPROPS  A5  Detail

1.27  PRESCOST N2  Study
2.05  PRESCOST MS2  Study
4.52  PRESCOST A5  Study
```

Figure 2-2: Demonstration AUR input file

2.4 ABLE Output Format (AOF) files

This Section specifies the variety of formats available for files produced in ABLE Output Format (AOF). This format is particularly suitable if results are to be transferred from ABLE to another computer system, possibly for further processing by the user.

Use of the Report module is described in Chapter 8, including how to opt for the different formats of output described here. This Section is concerned principally with the syntax, not the semantics, of the output; nevertheless, some appreciation of the function of the Report module is necessary in order to understand the various aspects of the output produced. The following sections therefore make reference to this function as necessary in the context of this Chapter—but if details are required you must refer to Chapter 8.

Section 2.4.1 outlines briefly the options open to you in producing output from the Report module. Section 2.4.2 goes into more detail of the implications of your choices for different output media. Section 2.4.3 provides some examples of ABLE Output Format.

2.4.1 Reports

An AOF file is an ASCII file consisting of one or more *reports* produced by the Report module. Reports can have different formats when output to screen or printer, or to other kinds of files, and are fully described in *Chapter 8 — Reporting.* This section briefly considers those aspects that are relevant to AOF files.

A report presents estimates for one or more derived or base quantities in the model, as computed from a chosen AFS file. It will typically also produce descriptions of the accuracy of those estimates, which may be expressed in one of a number of different ways. The different accuracy specifications are defined in Chapter 8, but it should be noted that specification of accuracy as an *interval* means that accuracy is shown as a pair of numbers (lower and upper bounds of the interval), whereas otherwise it is a single number.

The main content of a report consists of one or two tables of numbers. If there are two tables, the first contains estimates of the reported quantities and the second contains the corresponding accuracy specifications. If there is only one table it either contains only estimates, or estimates and accuracy specifications in alternate columns.

A table has one column for each quantity being reported, or two for each quantity if there is just one table containing both estimates and accuracy specifications. Each row of the table gives estimates and/or accuracy specifications for a unit (i.e. one asset, group of assets or “zone”) on which that quantity is defined. (All quantities being reported in a single report must be defined on the same ID list of units.)

In other forms of output from the Report module, tables really appear as tables and may have headings and lines drawn round them; but in ABLE Output Format a table is thought of as simply a series of numbers, read row by row. That is, the first number is the number in the first column and first row of the table, the second is the number in the second column and first row, and so on across the first row. Those numbers are followed by numbers in the second row and so on. What makes this format so flexible is that you can specify exactly what is placed on the file *between* the numbers: to separate one number from another, to mark the ends of rows, ends of tables, ends of reports and so on. These formatting options are described in the next section.

2.4.2 Formatting

You may format *each report* separately by specifying the various features set out below.

2.4.2.1 Tables

You determine the number and form of the tables in a report via the following options:

Quantities
The number of quantities specified for a report determines the number and order of columns in a table. (See also Combined/Separate and ID List below.)

Accuracy
Various forms of specification are possible, including “none”, in which case only a table of estimates will be given. Specification as an interval results in a pair of numbers, otherwise a single number.

Combined/Separate
If Separate is specified (and accuracy specification is not “none”), two tables will be produced. Otherwise one table is produced; and if accuracy specification is not “none” there will be two columns for each quantity being reported.

ID List
If ID List is to be shown, it is given as a first column in each table (so as to identity the rows).

2.4.2.2 Titles and separators

Text before, between and after the numbers (and ID list names if requested) in a report are specified as follows. All these items of text may include newline characters, or any other ASCII characters as required.

Title
Text to be placed at the beginning of the report, before the first number in the table.

Interval separator
Text to be placed between the two numbers comprising an accuracy specification when it has been requested as an interval. Note that then the two numbers and separating text form one data item for the purposes of the field separator.

Field separator
Text to be placed between the data items in each row of each table. A data item corresponds to a single column, and so to an estimate or an accuracy specification. (An accuracy specification as an interval is a single item for this purpose.) The field separator is *not* placed at the beginning or end of a row, only *between* items.

Record separator
Text to be placed between rows of each table. It is not placed before the first row or after the last row of a table, only *between* rows.

Table separator
Text to be placed between tables if the report has two tables.

Report terminator

Text to be placed at the end of the report, after the last number in the last table.

2.4.3 Examples

Two examples are given here to show the flexibility of AOF files. In practice, you should be able to instruct the Report module to format AOF files to input easily to spreadsheets or other software for further processing.

2.4.3.1 Example 1

This is a very simple example using data presented in other output formats in the examples of Chapter 8. The following have been specified for the first report:

Quantities:	DivQual
Accuracy:	90% interval
Combined/Separate:	Combined
ID List:	Shown
Title:	`"Divisional quality costs (thousands of pounds)"`, newline, newline
Field separator:	space
Record separator:	asterisk, newline
Interval separator:	space
Table separator:	newline, colon, newline
Report terminator:	newline, colon, colon, newline

The second report is set up the same except for

Quantities:	TotQual
ID List:	Not shown
Title:	`"Company total"`, newline, newline

Notice that the field and interval separators have been made the same, because the user wishes in this instance to read the two bounds of a 90% interval as separate fields. Both reports have a single table, so the table separator is not used. The output is as follows.

```
Divisional quality costs (thousands of
      pounds)

North 17653 15578 19894*
South 2009 1667 2392*
East 10130 8800 11580*
West 9822 8254 11566
::

Company total

39614 36608 42711
::
```

2.4.3.2 Example 2

The NEWDEMO.AOF file supplied with ABLE is output from the Report module (as a report from the NEWDEMO.AOF file) using the following formatting:

Quantities:	REHABC, PRESSURE, SYSGROW
Accuracy:	Standard deviation
Combined/Separate:	Separate
ID List:	Shown
Title:	The default title showing file name and dates
Field separator:	space, space
Record separator:	newline
Interval separator:	space
Table separator:	newline, newline
Report terminator:	newline, newline, newline
Totals:	On
Calculation mode:	Simulation, 400 samples

The separators are the default ones (and the interval separator was not used). The resulting NEWDEMO.AOF file is listed below.

```
File -- C:\MATHS\APL\ABLE214\ABC\NEWDEMO.AFS
Last edited -- 1998-02-08 13:11:37
Report date -- 1998-02-09
Simulation with Sample Size 400
C1  1003  10  596
C2  1210  78  295
C3  416  16  88
C4  803  16  97
MS1  0  138  83
MS2  902  29  633
MS3  326  9  169
MS4  197  11  1648
MS5  488  26  460
MS6  201  86  240
MS7  555  10  402
MS8  410  120  286
W1  548  0  546
W2  363  0  302
W3a  249  0  166
W3b  239  9  178
W4  2600  15  26
W5  946  9  110
W6  54  124  99
W7  287  12  145
W8  2099  0  203
W9  1447  20  333
W10  1956  11  17
W11  2740  13  445
W12  3466  9  51
```

```
W13  196  8  47
W14  1909  21  100
W15  531  0  118
W16  380  15  118
W17  338  0  34
A1  1748  15  112
A2  169  11  92
A3  245  0  97
A4  227  0  3
A5  457  43  143
A6  172  14  132
A7  1113  8  410
A8  61  20  75
A9  3220  19  239
A10  1172  14  222
A12  995  11  367
SS1  538  18  132
SS2  519  7  122
SS3  221  112  44
SS4  0  0  99
SS5  19  0  22
SS6  1022  11  140
SS7  26  0  130
SS8  238  15  215
SS9  2967  193  364
E1  2090  7  280
E2  1243  0  121
E3  46  0  78
E4  3672  16  340
E5  117  82  76
E6  788  5  318
E7  44  7  89
E8  1008  7  192
N1  2831  110  90
N2  1988  51  252
N3  673  44  314
N4  1380  12  517
N5  1498  21  275
N6  1007  111  512
N7  742  16  326
Total  61117  1816  14976

C1     706  5  511
C2     509  20  202
C3     638  6  69
C4     875  9  147
MS1     0  18  118
MS2     570  17  511
MS3     203  5  130
MS4     884  4  1204
MS5     1119  14  326
MS6     110  30  219
MS7     650  9  347
MS8     451  18  214
W1     662  0  718
W2     420  0  276
W3a     289  0  141
W3b     280  8  135
```

```
W4    1173  8  76
W5    699  8  89
W6    189  25  140
W7    418  11  125
W8    1451  0  140
W9    2347  7  251
W10    1151  4  36
W11    874  7  323
W12    1752  5  139
W13    850  8  84
W14    1908  8  160
W15    232  0  88
W16    979  6  90
W17    398  0  58
A1    625  6  81
A2    239  6  116
A3    307  0  133
A4    483  0  8
A5    1416  15  134
A6    622  8  105
A7    1565  5  302
A8    212  7  53
A9    795  7  186
A10    2942  8  173
A12    416  4  273
SS1    225  6  93
SS2    553  2  94
SS3    145  17  42
SS4    0  0  140
SS5    62  0  55
SS6    1441  6  124
SS7    98  0  106
SS8    761  5  160
SS9    1389  25  330
E1    1042  3  245
E2    597  0  107
E3    151  0  123
E4    2086  5  254
E5    118  22  89
E6    1691  2  236
E7    137  2  65
E8    952  2  139
N1    1603  23  110
N2    1934  19  243
N3    927  14  224
N4    1103  7  401
N5    616  8  221
N6    1303  33  391
N7    1100  5  236
Total    42479  286  7750
```

2.4.3.3 Example 3

The listing below shows the same sample data as in Example 2 output with the formatting changed as follows::

Accuracy:	90% deviation
Combined/Separate:	Combined
ID List:	Not shown
Field separator:	comma, space
Title:	`"<REHABC, PRESSURE, SYSGROW>"`, newline

```
<REHABC,PRESSURE,SYSGROW>
Simulation with Sample Size 400
1003, 1161, 10, 9, 596, 840
1210, 837, 78, 33, 295, 332
416, 1050, 16, 10, 88, 113
803, 1440, 16, 14, 97, 241
0, 0, 138, 30, 83, 194
902, 938, 29, 28, 633, 840
326, 334, 9, 9, 169, 214
197, 1455, 11, 6, 1648, 1981
488, 1841, 26, 24, 460, 536
201, 181, 86, 49, 240, 361
555, 1069, 10, 15, 402, 571
410, 742, 120, 29, 286, 352
548, 1089, 0, 0, 546, 1180
363, 691, 0, 0, 302, 454
249, 476, 0, 0, 166, 232
239, 460, 9, 14, 178, 223
2600, 1930, 15, 14, 26, 125
946, 1149, 9, 13, 110, 146
54, 310, 124, 40, 99, 230
287, 687, 12, 18, 145, 206
2099, 2387, 0, 0, 203, 231
1447, 3861, 20, 12, 333, 412
1956, 1894, 11, 6, 17, 59
2740, 1437, 13, 12, 445, 531
3466, 2883, 9, 8, 51, 229
196, 1398, 8, 13, 47, 138
1909, 3138, 21, 13, 100, 264
531, 382, 0, 0, 118, 145
380, 1610, 15, 9, 118, 149
338, 655, 0, 0, 34, 95
1748, 1028, 15, 9, 112, 134
169, 394, 11, 10, 92, 191
245, 505, 0, 0, 97, 218
227, 794, 0, 0, 3, 14
457, 2329, 43, 25, 143, 221
172, 1024, 14, 13, 132, 172
1113, 2575, 8, 8, 410, 497
61, 348, 20, 11, 75, 88
3220, 1307, 19, 12, 239, 306
1172, 4840, 14, 14, 222, 284
```

```
995, 684, 11, 7, 367, 450
538, 369, 18, 10, 132, 153
519, 909, 7, 4, 122, 154
221, 239, 112, 29, 44, 69
0, 0, 0, 0, 99, 230
19, 102, 0, 0, 22, 91
1022, 2370, 11, 11, 140, 204
26, 162, 0, 0, 130, 175
238, 1252, 15, 9, 215, 263
2967, 2285, 193, 42, 364, 543
2090, 1714, 7, 4, 280, 403
1243, 982, 0, 0, 121, 176
46, 248, 0, 0, 78, 203
3672, 3432, 16, 9, 340, 418
117, 194, 82, 36, 76, 147
788, 2781, 5, 3, 318, 389
44, 226, 7, 4, 89, 107
1008, 1566, 7, 4, 192, 228
2831, 2637, 110, 37, 90, 181
1988, 3181, 51, 31, 252, 400
673, 1525, 44, 23, 314, 369
1380, 1814, 12, 11, 517, 660
1498, 1013, 21, 13, 275, 364
1007, 2144, 111, 54, 512, 643
742, 1810, 16, 9, 326, 388
61117, 69877, 1816, 470, 14976, 12748
```

ABLE:

Chapter 3 — Configuration

3.1 ABLE configuration

The Configuration module is provided to enable you to change certain default conditions under which ABLE operates. The module is entered from the Tools menu or the Configuration module button on the toolbar and you are then presented with a screen displaying the options that can be set. This is achieved by Tabbing to the required field then cycling through the available settings by keying `[Shift/↑]` and/or `[Shift/↓]`.

```
                    The ABLE Configuration Module

              Printer type      default printer

         Standard    level.                Set    Full   colours.

Disk saving of derived    On , base    On    and expressions   Off

─────────────────────── Directory Path ───────────────────────
 C:\ABLE

Tab/Btab,Shift =Select,F1=Help,CtrlDel=Del Line,Shift-Enter=Exit,ShiftEsc=Quit
```

Printer type

This field shows the current default printer set in Windows. ABLE uses the default printer. This field cannot be edited, so to use another printer it may be necessary to change the default in the Windows Control Panel.

(User) Level

Three levels of user are catered for: Basic, Standard and Expert. This setting affects ABLE facilities available to you.

!

In particular, the Basic level user is very restricted, and cannot change data or write files.

The Standard level user can perform all but a few activities that are designed for the statistical Expert level users.

An indication is given within the text of this manual when activities are denied to Basic level users, or are restricted to Expert level users.

Your user level on entering ABLE is determined by your password. If you are a Standard level user, you will be able to enter the Configuration module, but will not be allowed to change to Expert level without typing an Expert level password.

The coloured spot at the right of the status bar at the bottom of the screen is always a reminder of your current user level (green for Expert, amber for Standard and red for Basic).

Colour settings

You can request Full or Simple colour schemes to suit your taste.

Disk saving

This line of options controls the extent to which ABLE stores intermediate calculations on the hard disk. In general, time will be saved by having these settings On—but you may switch them off if disk space is at a premium, or if you know you have little disk space free.

Directory Path

Directories are important in controlling which files are immediately accessible in some parts of ABLE.

You can type into this area the names of directories to be searched for ABLE files by the File Management module. Directory names should be typed one per line, and will be searched in the order in which you give them.

ABLE:

Chapter 4 — Elicitation

4.1 Introduction

The Elicitation module is used to construct an ABLE Variance Elicitation (AVE) file, which forms part of the input to the ABLE Full Specification (AFS), or System file. The module is entered whenever you load an AVE file as the active file in ABLE.

! Users at the Basic level of expertise (see *Chapter 3 — Configuration*) can use this module only to *view* the data in an elicitation (AVE file) that has already been carried out, and to produce a print-out if required.

Elicitation is a complex operation and can be difficult for someone without training in Bayesian statistics. Before using the Elicitation module—and, indeed, before using ABLE at all—you should have carried out the following preparatory steps with the aid of a statistician.

1. You should have a Cost Model, i.e. a set of equations defining derived quantities of interest in terms of base quantities.
2. You should have formed base quantities into correlated groups.
3. For each group, you should have decided about stratification and scaling.
4. For each group, you should have identified levels of uncertainty, i.e. the different possible kinds of estimate that could be obtained. The Elicitation Module will “conduct a dialogue” with you to find out various aspects of how accurate those estimates will be if actually obtained. You will therefore need to have given some thought to the estimation processes themselves, and their potential inaccuracies.

Some general discussion of these matters may be found in Section 4.2.

4.1.1 File navigation and help

The Help screen can be invoked at any time by pressing the `F1` key.

Keying `[Shift+↵]` takes you straight back to the menu which controls the collection of forms you are working on. `[Shift+↵]` again takes you back to the previous menu if appropriate.

Key(s)	Action
`Ctrl+↵`	*Select highlighted item from menu.*
`Ctrl+Insert`†	*Insert new menu item.*
`Ctrl+Delete`†	*Delete menu item.*
`Ctrl+PgDn`	*Move to (same page in) next form.*
`Ctrl+PgUp`	*Move to (same page in) previous form.*
`PgDn`	*Move to next page in current form.*
`PgUp`	*Move to previous page in current form.*
`Tab`	*Move to next (white) field in page.*
`↵ (Enter)`	*Move to next row in field.*
`F2`†	*Calculate.*
`F8`	*Print.*

† These functions cannot be invoked by Basic level users.

4.1.1.1 Colours

Text and numbers shown in ***white*** can be changed by typing over them.

Items in ***blue*** are changed automatically when you hit the Calculate key (`F2`, see later).

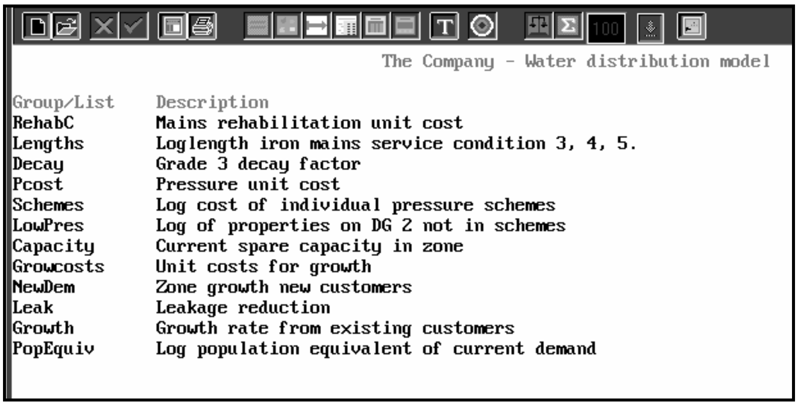
Items in ***green*** are fixed.

The procedure by which equations are colour-coded to distinguish visually between base quantities, derived quantities, operations and mappings can take a long time for large models. You are therefore offered the choice of not colouring the equations.

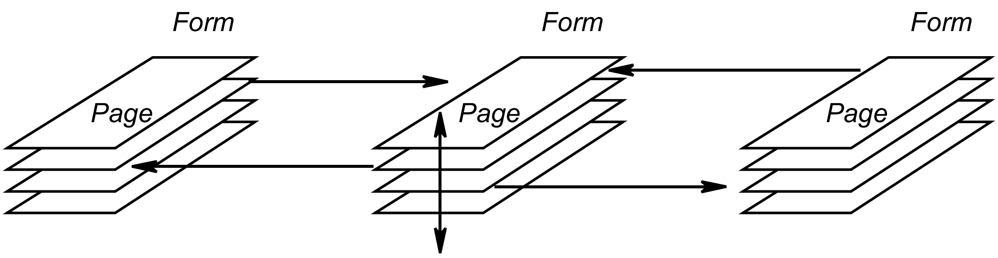
4.1.2 The menus and file organisation

On entry to the `Elicit Variances` option, the first menu consists of a list of files available for use.

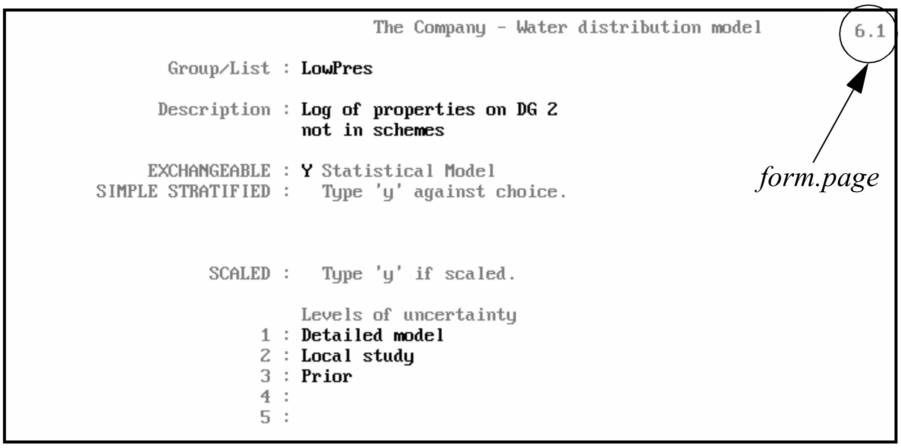
Selecting a file from the first menu brings up the second menu, which is the menu of base quantities or groups. For example:



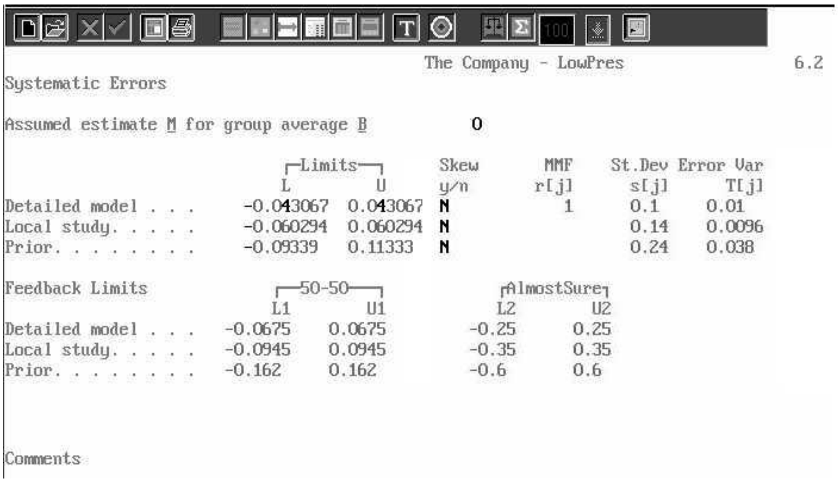
The file is organised in forms, each form being made up of several pages. Each base quantity or group has at least one form. If it is unstratified there is just one form, and if it is stratified there is an initial form plus separate forms for each stratum. The control and cursor keys enable you to move from page to page— `[PgUp]` and `[PgDn]` —and form to form — `[Ctrl+PgDn]` and `[Ctrl+PgUp]`. The pages of a form can be thought of as laid out one above the other in a column, so that moving up or down skips back or forward through the pages. Moving from form to form moves to the same page in the previous or next form in a collection of forms.



The menu of base quantities or groups basically consists of choosing from a collection of forms, which is the collection of first (or only) forms for all the base quantities or groups. Thus, moving the cursor to the sixth item, `LowPres` in the menu above and keying `[Ctrl+↵]` delivers the screen below:



While you are working on these forms, keying `[Ctrl+PgUp]` or `[Ctrl+PgDn]` takes you to the previous or next base quantity. Keying `[PgDn]` on the above form delivers the page giving details of systematic error:



In the initial form for a stratified quantity or group, the third page is a menu, for selecting the subsequent stratum forms. These stratum forms make up a separate collection, so when you are working on these keying `[Ctrl+PgDn]` or `[Ctrl+PgUp]` takes you to the next or previous stratum.

4.2 Group structures

The purpose of elicitation is to provide ABLE with details of variances, which express the accuracies of the individual estimates which you will enter for all the quantities in the model. This is a very important process. In order that ABLE can report correctly the variances for derived quantities, and in order that ABLE's unique method of learning from new estimates should operate correctly, variances must be specified carefully in the Elicitation module. New ideas and new terminology will be needed, and these are introduced step by step in this section.

4.2.1 Groups

We have already introduced the concept of a list, which corresponds to a column of data in our statistical spreadsheet analogy for ABLE. A list is represented by a single name in the model equations, such as `REHABUC` in the `NEWDEMO` model. We have also described ABLE's updating process which adjusts the estimates of other quantities in a list when we obtain a new estimate for any individual quantity. For instance, if the prior estimate of `REHABUC` is 35 in zone `N1`, and if we then obtained a more accurate “study” estimate of 39.4 for `REHABUC` in zone `N1`, ABLE's updating will adjust estimates of `REHABUC` in other zones, raising them slightly. This is reasonable because the new estimate in `N1` suggests that the prior estimation process may have tended to under-estimate the unit cost of rehabilitation in other zones.

Statistically, we say that the estimation errors for the various quantities in the list `REHABUC` are *correlated*. Whilst in general we do not expect quantities in *different* lists to be correlated in this way, it can sometimes occur. To understand correlation across lists, it is helpful to look again at the NEWDEMO model. An increased estimate of 39.4 for `REHABUC` in zone `N1`, compared with a prior estimate of 35, would *not* cause you to change your beliefs about the value of `CAPACITY` in any zone, so `REHABUC` and `CAPACITY` are *not* correlated. Indeed, it is hard to see how this information would cause you to adjust estimates for *any* other list, so `REHABUC` is not correlated with any other list.

On the other hand, consider the lists `logIRON5` and `logIRON4`. If you learnt that a particular zone, `A2` say, had more mains in condition grade 5 than you had initially estimated, then you might increase your estimate of the length of grade 4 mains. This would be appropriate if the increased estimate of grade 5 length suggests that in general the mains in zone `A2` are in worse condition than you had originally thought. In fact, we might consider that the *three* lists `logIRON5`, `logIRON4` and `logIRON3` should be correlated together.

This introduces the fundamental concept of a *group*. In ABLE, all the quantities in a group are considered to be correlated with each other, while quantities in different groups are treated as uncorrelated. A group comprises one or more lists. Often, a group contains just one list, as in the case of `REHABUC`. However, in the NEWDEMO model the lists `logIRON5`, `logIRON4` and `logIRON3` are formed together into a single group.

ABLE stores and updates data at the group level. You will elicit variances separately for each group. You will enter data separately for each group using the Create/Edit module as described in Chapter 5, and ABLE will update each group separately when you enter new estimates as described in *Chapter 7 — Updating*.

The definition of groups is therefore a key step in building an ABLE application.

4.2.2 Levels of uncertainty

ABLE needs to know about variances for all the different kinds of estimates that can be obtained for the various quantities in a group. Often, there are two kinds of estimate. One is a “prior” estimate, which is entered initially for all quantities in the group when you first build the application. The other is a “study” estimate, which can be obtained for individual quantities in the group by means of a detailed engineering study of some kind. Sometimes there is only one possible estimate, which will be the initial estimate, and this is the best that can be done. Other groups may have three or more different kinds of estimates available, of various stages of accuracy.

ABLE defines the different estimation processes as having different *levels of uncertainty*. The worse, least accurate, form of estimate has the *highest* level of uncertainty. The best available estimation method corresponds to the lowest level of uncertainty. Actually, there is always an implicit, unstated, kind of “estimate”, which is the “true” value of the quantity. Obviously this would have the lowest level of uncertainty of all, but since it equally obviously has zero variance we do not need to mention it explicitly or elicit variances for it.

For each group, you must identify all the possible kinds of estimate, and place them in order of accuracy so as to define their levels of uncertainty.

4.2.3 Systematic and random errors

Suppose that you have estimated the value of `REHABUC` as 35 in every zone. This does not mean that you believe that the rehabilitation unit cost will be the same in every zone. On the contrary, costs will certainly vary from zone to zone because of the varying conditions which contractors will experience from one rehabilitation job to the next. But if you have no information to suggest that costs will be higher or lower in any *particular* zone compared with any other, then it is clear that you should give them all the same estimate.

Now consider the possible errors in such estimates. There are two ways in which the estimate of 35 in zone `N1`, say, might be wrong. First, the figure of 35 might be wrong *on average*. It was estimated as an average value, and could be wrong in that sense. We call such an error a *systematic error*. Second, we know that zones will differ, and consequently the figure of 35 in zone `N1` might be wrong even if 35 is exactly right on average. After any systematic error has been taken into account, estimates for individual zones will still be in error, and we call such errors *random errors*.

Formally, a random error affects the estimates on a *single* zone or unit in the group, whereas a systematic error affects *more than one* estimate. In the above example, we considered a single systematic error affecting every zone alike. This is not the only kind of systematic error which ABLE recognises. The ABLE structures of scaling and stratification, dealt with below, allow you to say for any given group which kinds of systematic error are to be expected, and how they affect the various estimates.

It is necessary to elicit variances separately for systematic and random errors. This is extremely important when we consider ABLE's updating process to learn from new estimates.

Returning to the `REHABUC` example, suppose that you now obtain an improved estimate of 27 for zone `N1`. How should this affect your beliefs/estimates for other zones? The answer depends on the relative magnitudes (as expressed by variances) of the random and systematic errors. Suppose first that there is no random error. This means that all zones must have the same `REHABUC`, and so your estimate for every other zone should also change to 27. Now suppose instead that there is no systematic error. This means that 35 is exactly right for the average zone, and that any deviations from it are due entirely to individual random errors. So the new 27 differs from 35 only by a random error affecting just this zone, and you have no reason to change estimates in other zones at all.

The truth invariably lies *between* these extremes. Both kinds or error are to be expected. The new estimate of 27 differs from the old estimate of 35 partly because of systematic error (affecting all zones) and partly because of random error (affecting only zone `N1`). We should therefore be inclined to reduce our estimate of `REHABUC` in other zones, but not as far as 27. The extent to which ABLE adjusts estimates in other zones is directly related to the relative magnitudes (variances) elicited for systematic and random errors.

4.2.4 Exchangeability

In principle, we need to think about the random errors affecting each individual quantity in the group, and might assign a different variance to each. (Plus thinking about variances of systematic errors.) Since there can be hundreds, or even thousands, of quantities in a group, the task of eliciting all those variances could easily become prodigious. Fortunately, this is almost never needed in practice, because we can assume, at least approximately, one of a small number of standard variance structures.

The simplest of these is *exchangeability*, which simply says that all zones or units are subject to random errors of the *same* magnitude. This does not mean all random errors are the same, since then they would amount to a systematic error. But their probabilistic magnitudes, as expressed in their variances, will all be the same. No *specific* estimate is expected to be subject to a larger random error than any other. This is often true, at least approximately. Assuming exchangeability cuts the elicitation effort dramatically—there is only one random error variance to specify (for each kind of estimate, i.e. for each level of uncertainty).

Exchangeability also has implications for systematic error. It specifies that there is a single systematic error, affecting all zones alike, as in the `REHABUC` example. So there is just one systematic error variance to elicit (for each level of uncertainty).

4.2.5 Scaling

Now although exchangeability might be appropriate for a list or group like `REHABUC`, it is not clear that it is appropriate for other lists in the demonstration model NEWDEMO.

Consider for instance the group called LEAKRED. This contains the single base list of the same name, and represents the amount of leakage reduction in each zone over the AMP period. The prior estimate in zone MS1 is zero, showing no expected improvement (but no worsening). In MS2 the estimate is 0.03 expressing a belief that overall there will be a small improvement in this zone. But in zone MS3 the prior estimate is –0.60, perhaps because no leakage control scheme is planned for this zone, and therefore leakage may be expected to increase. Clearly our beliefs about the values of LEAKRED in the different zones are not the same, which seems to make this case different from REHABUC.

Nevertheless, exchangeability may still be an appropriate assumption for LEAKRED. That is, it may be reasonable to suppose that the magnitudes of random errors are the same for all zones. MS3 is not expected to have larger or smaller errors that MS2, even thought their estimates are quite different. Also, the other part of the exchangeability assumption, that there is a single common systematic error, may also be acceptable. This would represent an overall misjudgement of the scale of leakage or of the effectiveness of leakage control.

Now consider the group NEWDEMD, containing the list of the same name and representing the amount of new demand in each zone. Prior estimates vary greatly, from 0.06 in zone W6 to 8.49 in zone MS4. New demand cannot be negative. (This is by definition. New demand in the water industry refers to new customers. Change in demand from existing customers is called growth, and *can* be negative.) The possible error in W6 is therefore small, whereas the estimate for MS4 can and probably will have an error larger than 0.06. Exchangeability is clearly not appropriate for random errors, nor is it sensible to assume a single systematic error affecting all units by the *same* amount. It is far more realistic in a case like this to think of errors affecting estimates by the same *proportion* rather than the same amount. That is, the random errors in W6 and MS4 might be of the order of ±30%, say, meaning ±0.018 for the estimate of 0.06 in zone W6, but ±2.55 for the 8.49 estimate in MS4. And we could consider the systematic error also having a proportional effect in the sense that we might have systematically under- or over-estimated by 20%, say.

ABLE offers two devices for such a situation both of which are illustrated in the NEWDEMO model. One is *scaling*. To specify a group to be scaled, one simply specifies a number for each individual quantity in the group, which is its scaling constant. Then both prior and systematic errors are supposed to be proportional to the scaling constants. It is possible in ABLE to specify arbitrary scaling constants (when operating as an Expert level user), but the default scaling constants are the initial estimates of the quantities themselves. This has exactly the effect required, of making errors proportional to the estimates themselves.

The drawback of scaling is that if we subsequently obtain an improved estimate, the errors in that estimate will still be set as proportional to the *initial* estimate of that quantity, *not* to the new estimate itself. In the NEWDEMO model, the NEWDEMD group is defined as having only one level of uncertainty, so the initial estimates are all that will be available. No improved estimates are possible and so this difficulty does not arise.

To avoid such problems when improved estimates are possible, ABLE version 2 allows the use of logarithmic quantities. An example in the NEWDEMO model is the group logPOPEQUIV, containing the single list of that name. In the model equations, the population equivalent served by a zone is represented by the list `POPEQUIV`. Zones are of very different sizes, and `POPEQUIV` can have very widely differing estimates in different zones. Proportional errors are reasonable for estimates of `POPEQUIV`, just as in the case of `LEAKRED`. Now notice that in this model `POPEQUIV` is a *derived* quantity, defined by the equation

```
POPEQUIV = exp (logPOPEQUIV)
```

It is `logPOPEQUIV` that is the base quantity.

An error of +0.5 or –0.5 in an estimate of `logPOPEQUIV` corresponds to *multiplying* the estimate of `POPEQUIV` by exp(+0.5) = 1.65 or exp(–0.5) = 0.61. So the error in `POPEQUIV` is proportional to the estimate itself, no matter what the value of that estimate is. This property will hold true even after improved estimates are obtained: the magnitudes of errors are always proportional to the current estimates.

4.2.6 Stratification

Exchangeability is the simplest variance structure, but we often need something a little more complex. In particular, we may consider the quantities in the group as divided into two or more subgroups, called *strata*, such that we may be prepared to say that quantities in the same stratum are subject to the same magnitude of errors (possibly after scaling, or expressed in terms of a logarithmic quantity), but different error variances should apply to different strata.

Consider, for example, the `PRESCOST` group, containing the single list of the same name. This represents the unit cost of dealing with problems of low pressure. The company divides zones into three strata, labelled High, Average and Low, meaning that the unit cost is likely to be relatively high for a zone in the High stratum, etc. Random errors are likely to be correspondingly larger for zones in this stratum, and this is one reason for stratifying.

There is, however, a more important reason for stratification. Whenever you specify a group to be stratified, ABLE defines an extra systematic error for each stratum, known as a *stratum-specific* systematic error. In the case of `PRESCOST`, a stratum-specific systematic error is allocated to the High stratum, affecting all the zones in the High stratum alike, but not affecting zones in the other two strata. There are stratum-specific systematic errors defined also for the Medium and Low strata, *as well as* an overall systematic error which affects every quantity in the group. For `PRESCOST`, the existence of separate systematic errors allows us to learn, for instance, that we have systematically under-estimated costs in the High stratum but over-estimated them in the Medium and Low strata.

Each estimate now has *three* kinds of error—random, overall systematic and stratum-specific systematic. If we observe, for instance, a detailed study estimate of 77 replacing an initial estimate of 62 in one zone, then the discrepancy will effectively be partitioned by ABLE into three corresponding parts, based on the variances elicited for each kind of error. Because some part of the discrepancy is attributed to overall systematic error, ABLE will increase estimates of *all* quantities in the group, in all strata. Because a further part is attributed to stratum-specific systematic error, ABLE makes a further increase in estimates for other quantities in the same stratum as the quantity receiving the new detailed study estimate, over and above the general adjustment already described. Finally, since a part of the discrepancy is attributed to random error, even quantities in the same zone will not be increased by the full observed difference of 15.

4.2.7 Other structures

ABLE is actually capable of representing much more complex structures than simple stratification. However, in general to define such structures properly requires statistical expertise, and can only be done in ABLE by an Expert level user. ABLE's user interface is designed to allow the simple structures of exchangeability and stratification (which will suffice in all but a very few situations) to be defined as easily as possible.

Section 4.4.5 discusses cross-stratification and provides a way to implement this structure without needing the skills and powers of an Expert user.

The remainder of this Chapter is concerned with entering estimates and limits, and requesting recalculations. These operations cannot be performed by Basic level users, who may therefore wish to skip these sections.

4.3 Estimates and limits

The key things you will type into the file are assumed estimates for quantities and upper and lower limits around the assumed estimates.

4.3.1 Estimates

Each page that asks you to enter this kind of information begins with an assumed estimate. For a scaled quantity or group this estimate *must* be 1. This will often be preset in the file. (The program allows other values to be entered, for technical reasons, but you should never use any value but 1 for assumed estimates of scaled quantities.) For scaled quantities we are interested in *percentage* errors, so the assumed value of 1 represents 100%.

For an unscaled quantity or group, enter a *realistic* estimate as described in the appropriate section of this Guide. The estimate does *not* have to agree with any prior or other estimates that you supply for the AMP exercise or anywhere else. It is a hypothetical value, but it must be realistic, so use real estimates you have readily to hand.

! **Remember**: what *units* this quantity is defined in, and make your assumed estimate in those units.

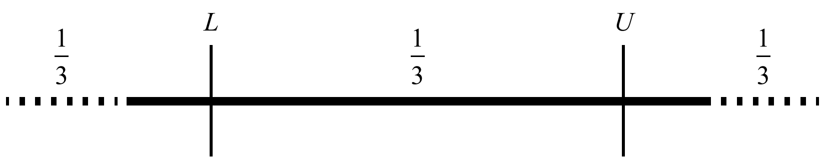
4.3.2 Limits

Limits consists of a lower limit (L) and an upper limit (U). At any stage, there are three sets of limits: 50-50 limits, Thirds limits and Almost-sure limits. Where the program just says `Limits`, these are Thirds limits which you must type in. You *only* ever enter Thirds limits. The others are always calculated by the program.

50-50 limits:	the quantity concerned has an equal chance of lying either inside or outside these limits.
Thirds limits:	the quantity concerned is twice as likely to be outside these limits as inside.
Almost-sure limits:	the quantity concerned is almost certain to be inside these limits.

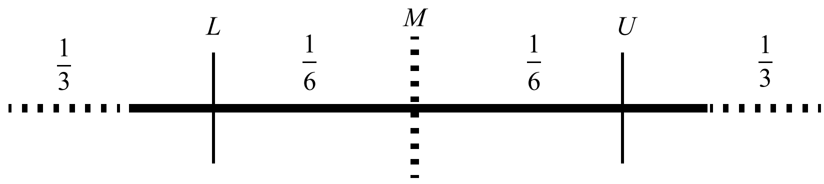
The upper and lower limits are also placed so that if the quantity is outside then it is equally likely to be above the upper limit or below the lower limit.

So when you enter (Thirds) limits, you must choose them so that (*in your judgement*) there is a chance of one-third that the quantity concerned will be below L, one-third that it will be above U, and one-third that it will be between U and L.



Thirds limits: a quantity is just as likely to be too high, too low, or to fall between the limits.

But it is *very important* that these limits are always related to the assumed estimate at the top of the page. You are assuming that this estimate is an actual unbiased estimate in the sense that the quantity concerned is equally likely to be above or below this estimate. (Technically, it is a *median.*) So if the assumed estimate is M we actually have three values L, M and U in increasing order. The chance should be one-third that the quantity is below L, one-sixth that it is between L and M, one-sixth that it is between M and U and one-third that it is above U.



Always place your L and U limits with reference to M.

Remember: if this is a scaled quantity, you are working in percentages. So if you set L to 0.8, you are saying there is a one-third chance that the quantity concerned is more than 20% below the actual estimate.

Note that versions of ABLE2 prior to version 2.5 used different limits. Then the user was required to specify Two-to-one limits instead of Thirds limits. Care should be taken to ensure that limits are specified correctly.

An important consequence is that if an AVE file created under an earlier version of ABLE2 is read using version 2.5 or later then its limits will be interpreted wrongly. The limits already on such a file should be corrected as described in Section 4.6

4.3.3 Feedback

Limits shown in blue are for feedback, to show you the implication of the limits you have given. You always get immediate feedback in the form of 50-50 and Almost-sure limits. Sometimes you get more feedback in the form of all three limits for a different quantity or scenario. Feedback may come on the same page or later. You must check all feedback limits to see that these do genuinely accord with your judgements. If not, you must go back and change one of the limits you have entered. Then check the feedback limits again, and go on adjusting until you are happy with *all* the numbers.

In particular, you must make changes if you think there is a real chance of something lying outside Almost-sure limits, or if 50-50 limits do not have a roughly equal chance of including or not including the quantity. But do remember that it is the *Thirds* limits you enter that you must ultimately feel really happy with. (We ask for those because we believe people are more reliable in judging these than either 50-50 or Almost-sure.)

4.3.4 Skewness and calculated limits

Feedback limits are calculated whenever you change pages or whenever you hit the `F2` (calculate) key. But here is another *important* thing to remember.

Whenever you enter or change limits, you *must* make sure that there is a question mark in the *Skew* column beside the limits you have entered or changed.

Calculated limits may not be correct if you do not do this before hitting `F2` or changing the page. The question mark will always disappear if you hit `F2` or leave a page and come back to it, turning to `Y` or `N`. But you *must* change it back to `?` whenever you make changes. (The program lets you leave `Y` or `N`, or even to type in `Y` or `N`, but this is for technical reasons only—you must always put the `?` back whenever you make changes.)

4.3.5 Levels of uncertainty

All limits must always be given for every possible level of uncertainty. You are saying how accurate you believe estimates of a given level of uncertainty *will be*. So, if there is a level of uncertainty called `Prior`, you are assessing the accuracy of prior estimates, individually and on average. If there is one called `Drainage Area Study`, you are assessing the accuracy that will be achieved for estimates from drainage area studies, both individually and on average. You must always use the *same assumed estimates.* So if you are dealing with `Drainage Area Study` estimates you are placing limits around the same hypothetical assumed estimate as if you are dealing with a different level of uncertainty on the same page.

! Levels of uncertainty higher up the table are always assumed to be more accurate, so limits must *never get narrower* as you move *down* the table.

4.3.6 Thinking about systematic errors

When assessing these bounds, you will be required to do so separately for random and systematic errors, and in the case of a stratified group the systematic errors must be assessed separately as overall systematic error and stratum-specific systematic errors. In each case, you do this by imagining that the other kinds of error are absent. So, for instance, when assessing random errors you put bounds on how far the true value might lie away from the estimate *assuming* that the systematic error is zero (or that both kinds of systematic error are zero for a stratified group). So you assume that the estimates are accurate on average (averaged over the stratum or over the whole group as necessary), and think to what extent an individual estimate might still be wrong.

Systematic errors are often hard to think about. Consider the case of an exchangeable group, where there is just one systematic error affecting all the quantities. If we imagine that there is no random error, then all quantities will be in error by exactly the same amount, which is the amount of systematic error. So you place bounds by thinking how far any one of the estimates might be from its true value. Equally, you could think how far the *average* of the estimates might be from the true average value. In the Elicitation module you are invited to think about how much error there might be in the average. Strictly, you must still remember that your bounds should be assuming zero random errors. But the advantage of using an average is that random errors in individual quantities tend to cancel out. If there are many quantities in a group, the average random error is almost certain to be very small. (This is why, when we have several measurements of something, it is good to average them.) So in this case, you can forget the injunction to assume zero random error when eliciting bounds for systematic error.

Now consider a stratified group. When thinking about a stratum-specific error, you have to assume that there is no overall systematic error and no random error. But again we think about bounds on the average, this time on the average of quantities *in that stratum*. If there are many quantities in the stratum we need not worry about assuming no random error, since random errors can be taken as cancelling out, but we have still to assume no overall systematic error, i.e. that over all the quantities in all the strata the estimates are right on average.

When eliciting bounds for the overall systematic error we think about bounds on the overall average, averaged over all strata. We should assume, but can ignore the assumption, that there is no random error. We also assume no stratum-specific systematic errors.

4.3.7 Difficult cases

The limits always assume that the values a quantity can take follow the sorts of probability distributions which statisticians generally deal with. That means, we assume that the quantity is more likely to be close to M than further away, and the further we go from M the less likely it is. There will be cases that do not fit this picture.

1. Sometimes a quantity must really be 0 or 1, because it is switching something on or off in the model. Then an estimate is a probability and it is not easy for a layperson to think about sensible limits.
2. Sometimes a quantity, like a measure of a problem, may very well be zero in a significant proportion of zones but cannot be negative. Then when calculating lower limits for individual zones the program either comes with negative values or refuses to include zero.
3. Sometimes a quantity is likely to equal (or be close to) one or other of two very different values but is less likely to be in the middle.

The first case was discussed in Section 1.2.2, where it was pointed out how the new `test(…)` operation allowed in ABLE version 2 equations will often make it possible to avoid having such “zero-one” quantities as base quantities. In general, problems such as these should be avoided by modelling where possible. When it is necessary to have base quantities like this, so that elicitation is needed, advice should be sought from a statistician.

4.3.8 Logarithmic quantities

A different, potentially difficult, case is that of quantities defined as logarithms. The demonstration model contains several such quantities. They are used, as explained in Section 1.2.2 and Section 4.2.5, to avoid difficulties of skewness and scaling.

! It is important to observe the following guidance when dealing with logarithmic quantities.

1. Logarithmic quantities should *not* be scaled. If *X* is the logarithm of *Y* (so that the model will include an equation `Y=exp(X)`) then adding ±0.1 to X corresponds to *multiplying* *Y* by exp(±0.1). So if *X* is an ordinary unscaled quantity *Y* is automatically scaled, its uncertainty being expressed naturally in percentage error terms.

2. Similarly, since we would use a standard value of 1 for the assumed value if doing the elicitation for *Y*, it is often convenient to assume a value of 0 always when eliciting variances for *X*. This is done for logarithmic quantities in the demonstration model file `NEWDEMO.AVE`. However, it is then important *not* to input skew limits. The elicitation module is not designed to handle skewness properly when the lower limits are negative. If ABLE

places a `Y` in the `Skew y/n` column, you should replace it with `N` and press `F2` again. Otherwise erroneous variances will be computed.

Note that any skewness in *X* would typically correspond to very skew limits on *Y*, and so any skewness in logarithmic quantities should usually be very slight.

3. In setting the limits, the following table may be a useful guide

Y	*X*	*Y*	*X*
±5%	±0.05	±30%	±0.26
±10%	±0.095	±40%	±0.335
±15%	±0.14	±50%	±0.405
±20%	±0.18	±75%	±0.56
±25%	±0.22	±100%	±0.693

Note that the interpretation of ±*y*% limits for *Y* is of an upper limit $1+\frac{y}{100}$ and a lower limit of $1/\left(1+\frac{y}{100}\right)$ For instance ±100% corresponds to limits of 0.5 and 2 for *Y*.

4. Finally, another feature of the relationship between *X* and *Y* is important to mention here, even though it is more concerned with *Chapter 5 — Creating an AFS (system) file*. There are several ways in which the idea of an *estimate* can be formally represented statistically. In ABLE, estimates are always *expected* values. The expected value of a random variable is the *average* value you would see if you could observe many independent instances of it. Two other interpretations of an estimate are the mode or median values. The mode of a random variable is its most likely value and the median is such that in many independent instances half would lie above and half below the median value.

For a symmetric quantity the three measures are the same, but for a skew quantity they will all be different. As mentioned above, you should generally think of using symmetric limits for a logarithmic quantity X, and this corresponds to skew limits for Y.

When building the system file, as described in Chapter 5, you will need to supply prior estimates of all the base quantities. In order to provide an estimate for a logarithmic quantity *X*, you will naturally think first of estimating *Y*. So, for instance, to estimate the `logLOWPROPS` quantity in the demonstration model, in zone `C1` you might estimate the number of properties (`LOWPROPS`) to be 61. Then an obvious estimate for `logLOWPROPS` is log(61)=4.111. However, it now matters what kind of estimate 61 is for `LOWPROPS`. If it is a median, i.e. you believe the number of properties is as likely to be above 61 as below it, then it is correct to set the estimate of `logLOWPROPS` to log(61). However, if you now use ABLE to report estimates of `LOWPROPS` it will report the expected value, which will be higher than 61.

If 61 really were your expected value for `LOWPROPS`, then you need to provide a smaller estimate for `logLOWPROPS`. How much smaller depends on the *total error* variances that you elicit for `logLOWPROPS`. The relevant group in the `NEWDEMO.AVE` file is called `LowPres`. The final screen of feedback is entitled Total Error. There is a column labelled `R[j]` giving the standard deviations of total error. For an estimate y of Y at a given level of uncertainty, if that estimate is an *expected* value, then the estimate you should supply to ABLE for *X*=log(*Y*) is

$$\log(y) - \frac{(R[j])^2}{2}$$

For instance, if the estimate 61 of `LOWPROPS` is at the Prior level of uncertainty, for which R[j]=0.3538, you should specify

$$\log(y) - \frac{0.3538^2}{2} = 4.048$$

as the estimate of `logLOWPROPS`. Then 61 will genuinely be the expected value of `LOWPROPS`.

4.4 Filling in the forms

Detailed page by page instructions follow on how to fill in each of the three different kinds of form: *exchangeable* forms and the two *stratified* forms. It is assumed here that the basic structural information on stratification and/or scaling will already have been entered (and can thus be regarded as "preset").

4.4.1 Exchangeable base quantities

There is a single form, of 4 pages, for exchangeable base quantities.

Page 1: This page gives basic information about the quantity, most of which will be preset. A short description (up to two lines) may be entered or changed.

Page 2: This page concerns systematic errors, i.e. possible errors in the average of a complete set of estimates. So the assumed value is an assumed average estimate for the whole region. (And if you are working with a level of uncertainty like `Detailed Study` you are assuming that a detailed study has been done on every zone or asset, and their average turns out to be the assumed value.) For a scaled quantity, of course, the average is 1 (i.e. 100%).

Page 3: This page concerns random error. Now you *must* assume that there is *no systematic error.* We are interested not in the average but in individual estimates, i.e. for an individual zone or asset. You are working with a typical estimate, so your assumed value (if unscaled) should be a typical value. It is therefore sensible for it to be the same as your assumed average value on page 1.

You are asked for limits on how far an individual zone or asset's true value might be from its estimated value. Random errors affect individual estimates differently (unlike systematic errors) and are generally due to the estimating process being unable to take full account of all the details and local circumstances of every case.

Page 4: This page is for feedback only. It shows you the total error that comes from putting together systematic and random error. It says how far an *individual* estimate might be from its true value without assuming the absence of systematic error. So the limits are wider than on page 3.

4.4.2 Stratified quantity or group

A stratified quantity has an initial form, followed by a stratum form for each stratum.

! If the group has two or more different stratifications—for example by Urban/Suburban/Rural and also by Inland/Coastal—then you should read Section 4.4.5 on Cross-Stratification.

4.4.2.1 Initial form

The initial form has 3 pages. Notice that if base quantities have been put together to form one correlated group, that group is (almost) certain to be stratified. There will be one stratum for each of the quantities that have been brought together into this group. There may also be other strata.

Page 1: This page gives basic information about the quantity, most of which will be preset. A short description (up to two lines) may be entered or changed.

Page 2: This page concerns systematic errors, i.e. possible errors in the average of a complete set of estimates. So the assumed value is an assumed average estimate for the whole region. (And if you are working with a level of uncertainty like `Detailed Study` you are assuming that a detailed study has been done on every zone or asset, and their average turns out to be the assumed value.) For a scaled quantity, of course, the average is 1 (i.e. 100%).

! This page is the same as for an exchangeable quantity but **remember** that for stratified variables there are two kinds of systematic error. This page is concerned with *overall* systematic error, i.e. errors in the grand average. (If this is a group with more than one base quantity the grand average is an average over all zones/assets *and* over all the base quantities.) As before, overall systematic error can be thought of as error intrinsic to the whole estimation process.

Page 3: This is a menu of strata which controls the various stratum forms for this quantity or group.

4.4.2.2 Stratum form

For each stratum there is a form rather like the exchangeable quantity form. There are 4 pages.

Page 1: This page concerns *stratum-specific* systematic error, i.e. errors in the assumed average estimate for the stratum. We consider the average of estimates over all the zones or assets in this stratum, assuming that there is *no overall systematic error.* We are looking for errors that affect all estimates in this stratum but not in other strata.

If the estimation procedure takes no account of stratum differences then the stratum-specific systematic error is likely to be large, reflecting the differences that might arise between the true stratum averages, which the estimation procedure cannot reflect. This might happen, for instance, in prior unit cost estimates where, because of lack of data, the same prior estimate is made for each stratum even though true stratum averages are expected to differ from one stratum to another.

A detailed study estimate, on the other hand, is likely to use information about which stratum a zone or asset lies in, and so stratum-specific systematic error should be low.

This page has space only to enter your (Thirds) limits. Feedback comes on the next page.

Page 2: This page is feedback only. First come the 50-50 and Almost-sure limits for the stratum average, assuming no overall systematic error. Then come all three limits combining both sources of systematic error. So they are limits for the true stratum average, but because they no longer assume the absence of overall systematic error they will be wider than those at the top of this page and on page 1.

Page 3: This page concerns random error. So the assumed estimate is an individual estimate for a typical member of this stratum. (Because it is a *typical* individual estimate, it could be the same as the assumed average estimate on page 1.) We assume *no systematic error* of any kind. Setting limits here is just like setting random error limits for an exchangeable quantity.

Page 4: This page is feedback only. It shows total error, comprising random error and both kinds of systematic error. So it shows how far the true value for an individual zone or asset might be from its assumed estimates without assuming the absence of systematic errors. Limits will therefore be wider than on page 3.

4.4.3 Other columns of numbers

There are other columns, with headings like `MMF`, `St Dev` and `Error Var` which you cannot type in and may ignore. They are for the benefit of the statistician when checking the file after you have completed your elicitation, and for auditing (see Chapter 10).

4.4.4 Comments

Finally, there is a small space for comments on almost every page. You should use it as an *aide-memoire* for the thoughts that guided you in entering numbers on that page or in viewing the feedback limits.

! But this is *not* a substitute for documenting your ideas fully. For reasons of quality assurance and auditability, you should keep a separate record of any relevant thoughts and reasons that guided your elicitation. This need not be highly structured or even very detailed. Do not needlessly repeat the limits that appear on the form—a full printout of all the forms will be available as the primary documentation of your elicitation. The purpose of the separate record is to amplify and explain the forms.

4.4.5 Cross-stratification

It is possible for a group to be stratified in two different ways. For instance, costs per property for dealing with low water pressure might be stratified by whether a zone is Urban, Suburban or Rural (since costs for setting up and carrying out underground work might tend to be higher in urban areas), and whether it is Flat or Hilly (since work to deal with a small number of properties with low pressure might need to be more substantial in a hilly area). In technical terms, there are several different ways in which ABLE could model the systematic and random errors associated with multiple stratifications. The Expert level user has the necessary tools available to set up arbitrarily complex statistical structures, but to do so requires not only statistical expertise but possibly intricate work with the system file. ABLE was designed so that it would be easy to establish the simplest and most frequently used structures. One example is scaling: ABLE deals implicitly, both in the Elicitation and Create/Edit modules with “scaling” as the standard scaling using the prior estimates as the scaling vector. An Expert user can define different scalings in the Create/Edit module, but then a special elicitation would be needed to define variances.

Multiple stratification is probably the most significant area where ABLE has not been designed to allow appropriate structures to be set up easily. The Elicitation module offers the choices of “Exchangeable” (i.e. no stratification) or

“Simple Stratified” (i.e. one stratification). Consequently, AVE files do not have suitable information for cases of multiple stratification, and so cannot be used to draw variances in automatically in the Create/Edit module. Some advice is given here on one way to handle multiple stratifications, in a form known as cross-stratification (or more technically cross-stratification without interaction). You should consult a statistician to confirm that this is appropriate for a given situation (and for further clarification of the method if needed).

We first describe how to elicit variances in the Elicitation module for a cross-stratified group. Then (although it refers to material more properly found in Chapter 5) we describe the corresponding procedure to incorporate these variances via the Create/Edit module.

4.4.5.1 Elicitation module

You need to think about systematic and random errors in different ways, and the best approach is to have the elicitation done in two different groups. For systematic error, you should define the group as stratified and on the page for naming the strata you should give one list of stratum names, containing the names of all the strata in all stratifications. In the example, this means naming five strata—Urban, Suburban, Rural, Flat and Hilly. Now elicit systematic errors and skip the elicitation of random errors. The overall systematic error is, as usual, concerned with error in the overall average. When thinking about stratum specific errors, assume no overall systematic errors *and* no stratum specific errors in the other stratification(s). For example, when eliciting the stratum specific error for the Urban stratum, you are concerned with how far wrong your average estimate for Urban zones could be under the assumption not only that the overall average is right but also that averages for Flat and Hilly strata are also exactly right.

Random errors are potentially much more complex because now you must, in principle, consider combinations of strata. For instance, how far wrong could your estimates be in the group of zones that are both Urban and Flat, assuming that the averages for Urban zones and for Flat zones are right? Our procedure assumes that you believe that the random error will depend only on one stratification. In the example, for instance, you may believe that random errors will be of roughly the same size in the Urban-Flat, Suburban-Flat and Rural-Flat combinations, and that they will also be roughly similar in the Urban-Hilly, Suburban-Hilly and Rural-Hilly combinations. Then you would elicit random error variances using only the Flat/Hilly stratification.

You do this by setting up another group in which the list of stratum names comprises just Flat and Hilly. Now elicit random errors, skipping the pages concerned with overall and stratum specific systematic errors.

4.4.5.2 Create/Edit module

You should create the two stratifications for the group as defined in Section 5.6.2, but ensure that the stratification used to define random error variances is the *first* column in the stratification screen.

On the random error variances screen, press `F9` and select the group on the AVE file where you elicited random errors. On the systematic error variances screen, key `[Shift+F9]` and select the group where you elicited systematic errors.

4.5 Examples

Examples in this section are based on the NEWDEMO demonstration file (see also *Appendix B — The Sample Data Set*). You should select the NEWDEMO.AVE file in the Elicitation module and examine the data in the groups discussed in these examples. The examples intend to indicate briefly some of the thinking behind the elicitation actually given in that file. You should also try changing numbers and pressing F2 to see the consequences of your changes (remembering to enter a ? in the Skew column each time).

! Note that the names of groups, levels of uncertainty and strata do not generally match those used for the *same* groups/levels of uncertainty/strata in the NEWDEMO.AFS file. This has been done in order to demonstrate (see Section 5.8) that ABLE can cope with such inconsistencies, but it is *bad practice*. Inconsistent naming increases the risk of subsequent confusion and human error.

4.5.1 DECAY group

The quantity DECAY is the "Grade 3 decay factor". It has only one level of uncertainty (Prior) since the company cannot contemplate any kind of study to realistically improve their estimates of DECAY. It is exchangeable and unscaled.

The assumed estimate is 0.1 (corresponding to an estimated 10% of Grade 3 mains decaying over the AMP period). This estimate could be seriously wrong, as an average for the company, so bounds of 0.05778 to 0.15576 are set for the systematic error. Feedback gives a 50-50 range from 0.04597 to 0.2175 and an "almost sure" range from 0.005623 to 1.778. The last figure is clearly silly because DECAY cannot exceed 1 (i.e. 100%), but ABLE does not know that (and you certainly should be "almost sure" that average DECAY is less than 1.778!). The feedback bounds are accepted and we move to the next page in the form.

! Note that the elicited values of 0.05778 and 0.15576 have been set here to keep the elicitation consistent with that shown in the ABLE manual for earlier versions of the software. This approach has been followed for all the elicited bounds in the NEWDEMO.AVE file. In practice, one would not specify these numbers to anything like this accuracy.

Now assuming that the company average DECAY is exactly 0.1, how far from this figure might DECAY be for the Grade 3 mains in an individual zone? The decay rate might not be expected to vary wildly between zones, and bounds of 0.067394 to 0.14838 are set. The feedback 50-50 limits are 0.05388 to 0.1856 and the "almost sure" limits are 0.01012 to 0.9882.

On the next page, feedback gives limits for an individual zone, taking account of both systematic and random errors. Although the "almost sure" bounds have an even sillier upper bound of 5.947, the Thirds range of 0.04946 to 0.2022 seems acceptable (giving a single zone a one in three chance of a DECAY rate exceeding one-fifth).

4.5.2 LOWPRES group

The LOWPRES quantity is defined as "Log of properties on DG2 not in schemes". It differs from the last example in two respects. First it has three levels of uncertainty: Detailed model, Local study and Prior. Second it is a logarithmic quantity as discussed in Section 4.3.8. Prior estimates of the number of properties with low pressure are judged to have a systematic error having a one in three chance of being within ±12% , which according to the analysis in Section 4.3.8 implies an error for the logarithms of ± 0.11333. In fact, we judge that errors on the low side are somewhat less likely, so limits are set of -0.09339 (a figure which, again, is calculated to produce the same results as in previous versions of ABLE2) to +0.11333. The bounds are tighter, ±15%, for “Local Study”, and reduce further to approximately ±10% for “Detailed model”.

4.5.3 PCOST group

The PCOST quantity is the “Pressure unit cost”. It is unscaled, but is stratified into three strata: High, Average and Low. There are two levels of uncertainty. Page 2 of the main form considers the overall systematic error. An assumed estimate of 1.2 (thousands of pounds per property receiving low pressure) has been entered. It is judged that if the prior estimate were 1.2, averaged over all strata, then the true average is likely to be in the range 1.02773 to 1.37227 (with probability about one third); whereas if studies were carried out in all zones, and yielded an average (over all strata) estimate of 1.2 then the true average would now be likely to lie in the range 1.11387 to 1.281613. Although the range is narrower, it is still quite wide (± 7% almost), reflecting the fact that pressure problems are hard to cost, even after detailed study, without developing complete engineering schemes.

If you now select the High stratum on page 3, you will see stratum-specific systematic errors being estimated. The assumed estimate is now 4.7, which is appropriate for zones in the High shortfall stratum. If there were no systematic error, so that the overall average actually were 1.2, how far from 4.7 might the average in the High stratum be? For the prior estimates, bounds of 4.1832 to 5.2168 have been entered, resulting in feedback “almost sure” bounds of 1.7 to 7.7. Notice that the person doing this elicitation really is almost sure that the High stratum average will be above the overall average of 1.2.

The next page shows feedback. In particular, allowing for uncertainty on the overall average, the 50-50 bounds for the prior estimates give equal probabilities to the High stratum average lying within or outside the range 3.954 to 5.587. The corresponding “almost sure” bounds are from 2.477 to 8.917.

The next page deals with random errors. If the High stratum average is exactly 4.7, how far from this figure might the unit cost be for pressure problems in a single zone? The final feedback amalgamates all sources of uncertainty, giving bounds for a single zone allowing for possible error in the stratum mean.

4.6 Revising an old elicitation file

As discussed in Section 4.3.2, an AVE file created using a version of ABLE earlier than version 2.5 will have the wrong elicitation limits if then used with a more recent version of ABLE. The limits on the file should be amended as follows:

1. Make a copy of the old AVE file using Windows Explorer or similar, and give it a suitable name to indicate that it relates to an old version of ABLE2.
2. Start ABLE2 and open the AVE file.
3. Change limits shown as not skew (i.e. with N showing in the column labelled Skew) so that their distance from the assumed central value M is reduced by a factor of 0.43. (Actually, 0.43067 is a more accurate figure, and will reproduce very precisely the variances that are implied by the original file, whereas 0.43 is an approximation that is adequate in practice.) For example, if the assumed value is 1 (as would be the case with a scaled quantity), and the old elicited bounds are 0.8 to 1.2, then the new bounds should be 0.914 to 1.086, since these are just 43% of the original distance from the assumed M=1.
4. The method to change limits shown as skew (i.e. with Y showing in the column labelled Skew) is unfortunately a little more complex. If the original limits are L (lower) and U (upper), first let G be the square root of (L times U). Next let R be (U divided by L) raised to the power 0.21533. Then the new limits are G divided by R (lower) and G times R (upper).
5. Be careful to do this for all elicited values on every form in the file. Do not change anything else, and in particular do NOT change the N or Y in the Skew column to “?”. Save this file. It is now ready for use with ABLE version 2.5 or later.

ABLE:

Chapter 5 — Creating an AFS (system) file

5.1 Introduction

There are 3 key components in an ABLE Full Specification file (AFS or System file):

- the equations of the model
- the definition of the base groups or quantities
- the definition of the mappings.

The Create module is actually in 3 parts, one dealing with each of these components. Each has its own button on the toolbar.

- The Equation Editor button is an equals sign.
- The Group Data button is the next button on the toolbar, showing a “group” of coloured blobs.
- The Mappings button is the next button, showing an arrow.

Pressing any of these buttons takes you into the appropriate section of the Create module, to work with the currently active AFS file. Alternatively, there are 3 corresponding options (Equation Editor, Group Data and Mappings) in the View menu.

The Equation Editor allows you to view, define and edit the model equations. Section 1.2 contains an explanation of ABLE modelling through defining the model equations. It also includes a detailed description of the equations for the demonstration model which is supplied with ABLE version 2. The equations are the key to any ABLE application. They define what the user can obtain reports about, and how those are to be constructed from current information about *base quantities*.

That information is defined and edited in the Group Data section. This is the most complex section of the module and the most complex part of ABLE. It deals with the formation of base lists into groups, and with the various items of information which together define the data for each group.

The idea for a mapping was introduced in Section 1.2.2. It is a means of relating lists of quantities and is explained fully in Section 5.7. The Mappings section of the Create module is where mappings which appear in the model equation are defined and edited.

5.1.1 Help

You can inspect the help screen at any time by hitting the `F1` key. A line of additional, context sensitive, help is usually displayed at the bottom of the screen indicating how key tasks are performed.

Selecting from a menu is always done by using the ↓ and ↑ cursor keys to place the cursor on the required item, then keying `[Ctrl+↵]`.

5.2 Viewing, building or editing

The 3 parts of the Create module may be used for different purposes:

- simply to view the information in an AFS file,
- to build a new AFS file, specifying its equations, group data and mappings, or
- to edit the data in an existing AFS file.

These activities are also affected by the type of user and by whether the AFS file has been updated, as follows.

As described in Section 1.4.1, you can log into ABLE as a Basic level, Standard level or Expert level user. This user type may also be changed while you are using ABLE through the Configuration module (see Chapter 3).

! Basic level users may only *view* data in an AFS file, and may not build new files or edit existing files. Standard level users can create or edit all aspects of the file, but do not have access to special features used to handle nonstandard situations. These features are available only to Expert level users.

When a file is first created, and built up by specifying all the 3 components in full, it is said to contain *prior* information. A vital feature of ABLE is the way in which that information is added to and modified subsequently by inputting improved estimates of base quantities, an activity which is handled by the Update module. Updating affects the information stored on individual base groups, and as soon as the information on any group has been updated it is said to contain posterior information.

When, in the Group Data section of the Create module, you select a particular group to view or edit, ABLE displays on the Status bar at the bottom of the window whether the information on that group is prior or posterior information. This affects what information can be viewed or edited for that group.

5.2.1 Building a new file

The process of building an AFS file is essentially that of creating the ABLE application. It requires planning beforehand, to develop the equations (which define what base lists and mappings will be needed), to define appropriate base groups and structures for the information on each group, and to assemble the data to supply the Group Data and Mappings parts of the module.

Defining equations, groups and information structures for the group requires a full understanding of how ABLE works, and you will learn better what is involved and what tricks and ideas can be employed through experience with actually using ABLE. It is beyond the scope of this manual to try to develop that expertise except through some simple illustrative examples.

Assembling the prior data can be a substantial task, and it would be quite impractical to type all the data into ABLE by hand. ABLE provides a very flexible file format for inputting such data, the AIF file format described in Chapter 2: *Input and Output Formats*, Section 2.2. It is possible to export data from standard spreadsheets and databases to conform to the AIF format (perhaps with minimal editing by hand) and thereby import them to ABLE.

A key kind of data needed in group specification concerns the accuracies of different kinds of estimates. ABLE provides a separate module for *eliciting* this information (See Chapter 4: *Elicitation*). The necessary data are then stored on a special file called an AVE file.

At many points in the Create module, it is a simple matter to import data from either an AIF file, by pressing `F11`, or from an AVE file by pressing `F9`.

Please note

Section 5.8 provides a detailed worked example of building an AFS file, using the demonstration model provided with the software. **You are strongly recommended** *to work through with the small sample data sets, AIF and AVE files, to create the demonstration model in order to gain a working understanding of the steps involved in building an AFS file.*

5.3 Equations

On entering the Equation Editor you are presented with the equations editing screen. For an existing file the existing equations are displayed. For a new file the screen is blank except for the name of the file shown as a comment at the top.

You can type the equations directly onto the screen, and move around using the cursor keys → ← ↓ ↑. Each equation should be on a new line, but blank lines are also allowed. The double slash symbol (`//`) denotes a comment, so any text following it on the same line is ignored.

Every equation must of course begin with the name of a derived quantity, followed by an equals sign (`=`). The rest of the equation can include any of the operations defined in Section 5.3.1 below.

Names of quantities or mappings must be single words of up to fifteen characters. Spaces are not allowed, but the underscore character (_) may be used instead.

Pressing the `F2` key causes ABLE to analyse the equations and redisplay them with colour coding. The colours are

White:	derived quantities
Blue:	base quantities and constants
Yellow:	mapping names
Green:	comments
Purple:	symbols like `=`, `{`, `(`, `)`, `+`, or `test`.

! In addition to “refreshing” the display by recomputing the colour coding, ABLE then offers the opportunity to check the syntax of equations. It is *highly recommended* that you accept this option, at least until you are completely familiar with equation syntax; ABLE may crash if it later tries to work with invalid equations.

Any equations with incorrect syntax will be highlighted in red.

Corrections can be made by deleting (`Delete` or `Backspace` key) or typing over existing text.

Keying `[Shift+↵]` or clicking on the tick button saves the equation in the file. ABLE then asks whether you are finished with this file or wish to continue. Keying `[Shift+Esc]` aborts, i.e. closes the current AFS file *without* saving any amendments which you have made to the equations.

! The procedure by which equations are colour-coded to distinguish visually between base quantities, derived quantities, operations and mappings can take a long time for large models. You are therefore offered the choice of not colouring the equations.

5.3.1 Equation syntax

As already mentioned, every operation must begin with a name, which becomes the name of the derived quantity defined by the equation, followed by an equals sign and then some *expression* which forms the definition of the derived quantity. This section defines what kinds of expression are permitted on the right hand side of the equals sign.

First, the expression cannot be just a single name or constant. Whilst this may be useful in principle, to provide an alias for some other quantity, or even for a constant, ABLE is not programmed to accept it. (If this feature is desired, it can be achieved by equations like `A = B + 0` or `A = 1 + (0 * B)`. The latter defines `A` as the constant 1, and gives it the ID list of the quantity `B`.)

So every expression must include at least one operation, such as adding quantities or performing a mapping. When more than one operation is to be included in the expression, parentheses (i.e. round brackets, "(" and ")") can be used to instruct ABLE in which sequence to perform the operations.

! You are strongly advised to use parentheses to specify exactly how an expression should be evaluated, since ABLE is not guaranteed to perform them in a particular order or to acknowledge rules of algebra such as performing multiplications before additions.

5.3.2 Binary operations

We now specify all the permissible operations, beginning in this Section with operations which combine two things, such as the addition operation. In the following specifications for each operation, *A* and *B* represent any quantity names (derived or base) or constants. A constant may be any number, and may include a decimal point but negative numbers are not allowed. The minus sign in ABLE operations always denotes the operation of subtraction and requires a quantity or constant to the left as well as to the right. So negative numbers and negation are not permitted, but the same effect can always be obtained by subtraction, if necessary by subtraction from the constant zero.

Note that if *A* and *B* are both constants ABLE may have difficulty with later computations because the ID list of the result is not defined. To explain this, remember that any named quantity (base or derived) in the equation is a generic quantity representing a list. ABLE needs to know what *units* a list is defined on. For instance, most of the base lists in the demonstration model of Section 1.2.2 are defined on water distribution zones, so each zone is a unit, but the `logSCHEMES` base list is defined on individual schemes, so that each scheme is a unit for this list. The collection of unit names (e.g. zone names or scheme names) for a list is called its *ID list*. The ID lists for base lists are specified in the group data, and ABLE needs to be able to deduce the ID list for every derived quantity.

! In all of the binary operations below—addition, subtraction, multiplication, division, power, maximum, minimum and test—if *A* and *B* are both lists then they must have the same ID list. The operations are then understood to be applied to each unit, and the result has the same ID list. For instance, the equation

```
PRESSURE = KNOWNPRES + OTHERPRES
```

in the demonstration model combines `KNOWNPRES` and `OTHERPRES` which are defined to have values in each zone. So they both have the list of zone names as their ID list. They represent costs for remedying different low pressure problems in each zone. The result `PRESSURE` is therefore also defined to have a value for each zone, which is the sum of the `KNOWNPRES` and `OTHERPRES` values in that zone.

If one of *A* or *B* is a constant, the result is to combine the constant with the value of the list in each unit separately. So `PRESSURE + 1` would simply add 1 to the value of `PRESSURE` in each zone. The result therefore has the same ID list as the list *A* or *B*.

Operation	**Example**	**Description**
Addition	`A + B`	adds the values of *A* and *B* on each unit.
Subtraction	`A - B`	subtracts the value of *B* from that of *A* on each unit
Multiplication	`A * B`	multiplies the values of *A* and *B* on each unit
Division	`A / B`	divides the value of *A* by that of *B* on each unit
Power	`A ** B`	raises the value of *A* to the power of the value of *B* on each unit
Maximum	`max(A, B)`	results in whichever is the larger of the values of *A* and *B* on each unit
Minimum	`min(A, B)`	results in whichever is the smaller of the values of *A* and *B* on each unit
Test equal	`test(A = B)`	results in a value of 1 for a unit if the values of *A* and *B* are equal on that unit, otherwise a value of 0
Test greater	`test(A > B)`	results in a value of 1 for a unit if the value of *A* on that unit is greater than that of *B*, otherwise a value of 0
Test less	`test(A < B)`	results in a value of 1 for a unit if the value of *A* on that unit is less than that of *B*, otherwise a value of 0

5.3.3 Other operations

ABLE allows three other operations. Two, logarithm and exponential, are easily explained. The third, mapping, is only defined here in terms of its syntax: further explanation is given in Section 5.7.

Operation	**Example**	**Description**
Logarithm	`log(A)`	where *A* is a list: results in the natural logarithm of the value of *A* on each unit, and the result has the same ID list as *A*. If *A* is constant, log(*A*) is also a constant, the natural logarithm of *A*.
Exponential	`exp(A)`	where *A* is a list: results in the exponential of the value of *A* on each unit, and the result has the same ID list as *A*. If *A* is a constant, exp(*A*) is also a constant, the exponential of *A*.

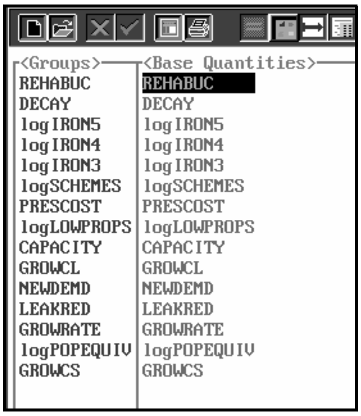
Operation	Example	Description
Mapping	`A { B`	Here *B* represents any list and *A* is the name of the mapping. The definition of *A* specifies what the ID list of *B* must be and what the ID list of the result is. Mappings are explained in Section 5.7. If a mapping sign is present it should be the first operation following the equals sign, either immediately after the equals sign or with the name of the mapping coming between the equals sign and the mapping sign.

5.4 Grouping Base Quantities

On selecting the `Group Data` section, you are presented with the groups formation screen. On the left hand side is a column of group names, and on the right hand side the base quantities in each group are shown. (If there are more groups than will fit on the screen, it will scroll up and down as necessary.)

5.4.1 Grouping in a new file

For a new AFS file, for which groups are not yet defined, the screen will initially show every base quantity in a group of its own, with the same name. For example:

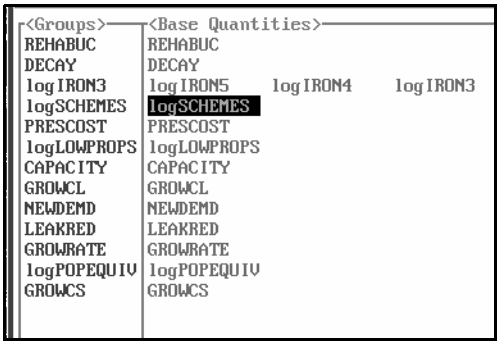


If quantities are correlated, i.e. they are subject to common systematic error(s), they should be formed into groups.

A quantity is marked for inclusion in a group by moving the cursor to its entry under Base Quantities and pressing the space bar. Once all the appropriate quantities have been marked, they are formed into a group by keying `[Ctrl+Insert]`.

The group then has the same name as the last quantity selected, this should be altered by moving the cursor to the Group entry and typing a new name.

For example, the quantities `IRON5`, `IRON4` and `IRON3` above form a correlated group; after marking them and keying `[Ctrl+insert]` they appear as follows:



The entry `logIRON3` in the first column could be altered to read `logIRONLEN`.

5.4.2 Changing groups

Groups can be split, merged and reformed to change or correct the groupings shown on screen. Pressing the space bar when the cursor is on a base quantity name selects that name, and pressing it when the cursor is on a group name selects all the base quantity names in the group. Pressing the space bar again cancels a selection. Keying `[Ctrl+Insert]` adds all selected base quantities to the group where the cursor is.

Pressing `Delete` with the cursor on a base quantity name moves that name to the end of the group. (The order of names in a group is not important but this allows you to arrange them in a logical way where appropriate.)

Keying `[Ctrl+Delete]` with the cursor on a base quantity name takes that quantity out of its current group and places it in a new group on its own. (This obviously has no effect if there is only one quantity in the group to begin with.)

You can change the name of any group by typing over the old name.

Pressing `[Shift+↵]` takes you back to the Equations/Groups/Mappings menu, saving the groupings on the AFS file. `[Shift+Esc]` aborts (exiting without saving).

5.4.3 The Groups menu

Once groups have been formed, the next thing to do with a new AFS file is to enter prior data for every group. The same screen also acts as a menu. Keying `[Ctrl+↵]` selects the group where the cursor is, and begins a series of screens for entering group data. These screens are described in Section 5.6. Important general principles for entering group data are covered in Section 5.5.

! It is important to finish forming groups before starting to enter group data, because if you subsequently decide to change any groups (split, merge or reform groups) then ABLE may have to throw away any data on the AFS file concerning affected groups.

Entering group data is a substantial and complex task, and must be done for every group. Notice, however, that it does not need to be done at one sitting. The Create/Edit module can be exited at any time, saving all changes and data entered up to that point, by pressing `[Shift+↵]` continually until you are returned to the main ABLE menu. You can enter more data (or view or change existing data) by re-entering the Create/Edit module at a later time and selecting that AFS file again.

5.5 Entering group data

Entering data for a group entails working through a prescribed series of 7 screens (more for Expert users), as explained in Section 5.6. A number of general principles applying to using these screens are considered here.

5.5.1 Navigating

When entering data for a group, the `[Shift+↵]` key sequence accepts data entered on that screen and moves onto the next. Keying `[Shift+Esc]` at any time on one of these screens aborts all the data entry so far for that group and returns to the groups menu. If, however, you work through all the screens, then pressing `[Shift+↵]` on the last screen accepts and saves the entire group data and returns you to the groups menu. Then you can select another group to enter or edit.

Some of the screens have subscreens for more detailed work. Keying `[Shift+Esc]` then just aborts that subscreen and takes you back to its parent screen.

5.5.2 Keyboard entry

Each screen is organised as one or more columns of data. The data may be numbers or words (names of things). You can always type on these screens (and use the `Delete` key) to change or edit any data showing on them. Sometimes the number of items in a column is fixed, but otherwise you can insert or delete items:

- pressing `[Insert]` inserts a new blank item before the cursor in that column
- pressing `[Delete]` deletes the item under the cursor.

Where there is more than one column, you can insert and delete whole rows:

- keying `[Ctrl+Insert]` adds a new row before the cursor
- keying `[Ctrl+Delete]` deletes the row under the cursor.

In some cases the entries in a column must be words, and those words must belong to some list that you have already specified. Then instead of having to type those words (and possibly making spelling mistakes), you select or change a word by cycling around the various possible words. Keying `[Shift+→]` or `[Shift+←]` (where → and ← are the cursor-right and cursor-left keys) cycles forwards or backwards through the list.

In addition to typing or cycling in this way, all screens allow the data in a whole column (or even many columns) to be input together from a suitable data source. The three possible sources—AIF files, AVE files, and previously defined ID lists—are described in the next three subsections.

5.5.3 AIF file input

AIF files are described in Chapter 2, Section 2.2. Data on AIF files are arranged as strings of numbers or words, called vectors. At various places in the group data entry screens you can read a specified vector from an AIF file, to fill (or replace) a column on the screen. This option is selected by pressing `F11` with the cursor in the column to be filled.

If there is more than one AIF file in the current directory, then the first time you press `F11` you will be offered a menu of AIF files to select from. (Pressing `F7` allows you to change directory by typing in a new directory path.) Otherwise you will automatically be given the only, or most recently selected, AIF file (see *Change file* below for information on changing the file). This appears as a menu of vector names. Each row represents one block on the AIF file (so a vector block appears as a row with one vector name, while a matrix block appears as a row of more than one name). The following operations are then possible.

Select a vector `[Shift+↵]`
selects and reads in the vector under the cursor at that point.

Abort `[Shift+Esc]`
aborts, and returns you to the data screen without changing the current column.

View `[Ctrl+↵]`
presents a view of the data in the data block on the AIF file corresponding to the row that the cursor is on. You can then view the block to make sure you choose the right vector of data. Pressing `[Shift+↵]` takes you back to the menu of vectors.

Change file `F11`
gives you the menu of AIF files, from which you can select a new file.

Calibrate `space`
When selecting a vector from an AIF file you can usually choose another vector first, using the space bar, to calibrate with.

Calibration is a process of checking, and if appropriate rearranging the order of, the selected data. The first group data screen is the unit names screen (see Section 5.6.1) where the ID lists for the base quantities in the group are specified. Thereafter, wherever in some other screen a column of data is required for a base quantity, those data need to be in the order corresponding to that quantity's ID list. Data in a vector on the AIF file could possibly be in some different order, but if there is another vector specifying the order that those data are really in, it can be chosen as a calibration vector, and ABLE will do any necessary rearrangement. Therefore the calibration vector must be a vector of names containing all the names in the quantity's ID list. If the calibration vector gives the names in the same order as in the ID list, then the selected vector is read in unchanged. Otherwise it will be rearranged into the required order. An example will help to explain this process.

If the base quantity's ID list begins `ZONEA, ZONEB, ZONEC,…,`

the calibration vector begins `ZONEA, ZONEC, ZONEB,…,`

and the selected vector begins `1.1, 2.2, 3.3, …`

then that vector will be read in and reordered to 1. 1, 3. 3, 2. 2 … before being used to fill the requested column. This is because the calibration vector is saying that the number 2.2 is for `ZONEC` and the number 3.3 is for `ZONEB`.

5.5.4 AVE file input

On the two group data screens concerned with variances, the usual way to input those variances is from an AVE file. (The AVE file is produced by the Elicitation module, whose purpose is to determine such variances.) Input from an AVE file is initiated by pressing `F9` on the first of these screens (the random variance screen—see Section 5.6.7). If there is more than one AVE file in the current directory then the first time you press `F9` you will be offered a menu of AVE files to select from. Otherwise you will automatically be given the only, or most recently selected, AVE file (but keying `[Shift+F9]` instead of `F9` always allows you to select an AVE file). This appears as a menu of names of groups on the AVE file whose structure matches that of the group you are entering data for (i.e. whether stratified, and if so how many strata).

Now keying `[Ctrl+↵]` with the cursor on a group name reads the random error variances from that group on the AVE file. However, if the group is stratified you will first be asked to set up a correspondence between the names of strata for that group on the AVE file and the names of strata you have declared for this group. This is done by cycling.

Notice that it is possible to define a group to have more than one stratification (see Section 5.6.2), but the Elicitation module only allows variances to be elicited according to a single stratification. If you read variances from an AVE file when the group has more than one stratification, ABLE ignores all but the *first* stratification. So it will offer you groups on the AVE file which have the same number of strata as the first stratification and on selecting one of these you will be asked to match stratum names with those of the first stratification. This feature is exploited in the procedure for handling cross-stratification described in Section 4.4.5, but in general you should seek expert statistical advice when considering multiple stratification.

To read systematic error variances from the same group on the same AVE file, press `F9` on the next group data screen (the parameter variances screen—see Section 5.6.8. If you should for some reason wish to read these from a different file from that used for random error variances, key `[Shift+F9]` instead of `F9`.

Reading from an AVE file reads a whole screenful of data, not just one column.

5.5.5 Copying and editing ID lists

Any column of words is called an ID list within the Create/Edit module. The most important kind of ID list is the list of unit names for a base quantity, which we call the ID list for that quantity. When a column of words is required, then if the same column has been used before it can often be obtained quickly by copying. Pressing `F4` allows you to copy any of the ID lists already stored for this AFS file. The resulting screen shows all those previously declared lists as columns, and moving the cursor to the required column then pressing `[Shift+↵]` will copy that column. (The columns will scroll left and right if there are too many to fit the screen.) `[Shift+Esc]` aborts as usual.

! It is particularly important that you *should* use this method when setting the ID lists of unit names for base quantities. Several operations in ABLE are valid only for quantities defined on the same ID list, and it is therefore necessary for ABLE to know whenever quantities have the same ID list. Copying (from previously entered ID lists or as described in Section 5.5.6 below) is the way of ensuring that ABLE stores this information properly.

It is also possible to edit previously defined ID lists in this screen. Keying `[Ctrl+↵]` makes the entries in all the lists editable. Keying `[Ctrl+↵]` a second time makes the *names* of the lists, which appear as headers at the top of the columns, editable. Key `[Shift+↵]` to accept or `[Shift+Esc]` to abort and undo all edits.

5.5.6 On-screen copying

Another form of copying can be useful for groups having more than one base quantity. Then many group data screens have a column for each base quantity. One column can be copied from another as follows. With the cursor in the column to be copied key `[Ctrl+Home]` to mark that column. Then move the cursor to the column you wish to copy into and key `[Ctrl+End]`. The marked column, initially the first, is indicated by a heading in yellow, rather than green.

5.6 Group data screens

This Section considers each group data entry screen in turn. For important information on entering data in these screens, see the previous Section (5.5).

! Note that the sequence of screens is affected by whether this group has been updated or not. If the group has not been updated, the status line at the bottom of the window will say `PRIOR INFORMATION`. The group is then ready for definition of prior group data when first building the AFS file, or for subsequent editing of this data. Sections 5.6.1 to 5.6.9 assume that you are working with `PRIOR INFORMATION`. If the group has been updated, the status line will instead say `POSTERIOR INFORMATION`. Section 5.6.10 explains the differences which apply when viewing or editing an updated group.

5.6.1 Unit names

Enter the ID list of names of the units on which each base quantity is defined. These will almost certainly be read in from an AIF file, but if typed in, note that each name must be one word (of up to 15 characters, spaces are not allowed). When two or more quantities have the same ID list, these will in the first instance be typed or read from an AIF file (`F11`) and thereafter will be copied from the ID list (`F4`)—i.e. the list as defined in the first instance.

5.6.2 Stratifications

On this screen each column represents a stratification of the group, and contains the names of the strata for that stratification. Extra stratifications can be created by keying `[Shift+Insert]` to open a new column. Keying `[Shift+Delete]` will delete the stratification under the cursor.

Names of strata must be single words, i.e. containing no spaces. There are three ways to enter the names for a given stratification:

From AIF file: Using `F11`, the complete column of strata names can be read from an AIF file.

From ID list: Using `F4`, the complete column of strata names can be read in as a copy of an already defined ID list. Note that lists of strata names are stored by ABLE as ID lists, so this is a quick way of repeating a stratification whose strata names have already been defined for another group.

By direct entry: You can also simply type strata names in. First create enough lines for typing in by keying `[Insert]` the required number of times. (N.B. Do not use `[Ctrl+Insert]` if you have more than one column/stratification, because this will change the other stratifications and may cause ABLE errors.) Then simply type in the names of strata. Note that the previous two methods will also give the stratification a name, which appears at the top of the column. You cannot type in a name in this screen, but since lists of strata names are ID lists, you can create or edit their names using `F4`—see Section 5.5.5).

A stratification requires more than just names of strata. You must specify which units for the base quantity belong in each stratum. To specify each stratification in this way, key `[Ctrl+↵]` with the cursor in that column. This opens a new subscreen to set up the stratum allocations. These can be read from an AIF file, or entered or edited from the keyboard by cycling. Press `[Shift+↵]` to accept the allocations and return to the stratifications screen (or `[Shift+Esc]` to abort).

A special way of setting up a stratification is possible for groups with more than one base quantity. Keying `[Ctrl+↵]` in an empty stratification column (i.e. with no stratum names—note that if there are no other stratifications the cursor is always in an empty first column) creates a stratification in which the stratum names are the base quantity names and all units for a base quantity are allocated to the stratum of the same name (so both stratum names and allocation are created automatically).

Pressing `[Shift+↵]` in the stratifications screen only allows you to continue to the next screen if all stratifications have had stratum allocations defined.

! If you define two or more stratifications, it will not be straightforward to elicit or to specify suitable variances. Section 4.4.5 gives details on one way of handling two stratifications. In general, structures with multiple stratification should be used only with statistical advice.

5.6.3 Current estimates

In the current estimates screen you enter the prior estimates of each base quantity, for each unit In its ID list. These may be typed in or read in from an AIF file. There is also a special way to fill a column using a stratification, which is initiated by keying `[Ctrl+↵]` in that column. If there is more than one stratification for this group, you must next select a stratification to use. Then you must enter (typing in or from AIF file) an estimate for each stratum. The effect is to give to each unit the estimate you have specified for the stratum to which that unit belongs.

5.6.4 Uncertainty levels

The next screen has a single column to enter the names of the uncertainty levels for this group. These can be read from an AIF file or copied with `F4` (because the Create/Edit module defines it as an ID list). If typed in, note that each level of uncertainty name must be one word; spaces are not allowed. Having named the levels of uncertainty you need to allocate a level of uncertainty to each unit for each base quantity in the group. Keying `[Ctrl+↵]` produces the screen for assigning levels of uncertainty. The level of uncertainty given to each unit must be the level of uncertainty for the estimate given to that unit in the previous screen. By default, all are set to the highest level of uncertainty. This will often be the correct allocation (when all estimates are prior estimates and all at the same level of uncertainty), but can be changed by cycling, by reading from an AIF file, or by using a stratification (in the same way as defined for estimates in Section 5.6.3).

5.6.5 Scaling

Select either `Yes` or `No` and key `[Ctrl+↵]`. If `Yes`, Expert users only will see an extra screen, where they can change the default scaling.

5.6.6 Design matrix

This screen appears only for Expert users (who need to understand the underlying statistical theory).

5.6.7 Random error variance estimates

Variances should always (unless you are an Expert user) be imported from the AVE file (`F9`). See Section 5.5.4 above.

Keying `[Ctrl+↵]` opens a screen where you can view the allocation of random error variances to individual units. By amending the values in this screen it is possible to express a view that the random error variances in individual units might be higher or lower than the typical values elicited for all units (or for all units in a given stratum). Keying `[Shift+Esc]` will cancel any amendments and return you to the main random error variances screen, while `[Shift+↵]` accepts any amendments and moves on to the next screen.

Two items on the `Insert` menu allow for resetting this screen. Selecting `Insert>Simple Default` resets each variance to the corresponding variance from the main random error variances screen (according to the defined stratification, as appropriate). If you are editing an existing AFS file and have made some changes, you can select `Insert>Prior Setting` to reset this screen to how it was when you entered, thereby undoing any changes.

5.6.8 Systematic error matrices

This screen appears only for Expert users. It allows the definition of nonstandard systematic error variance structures, but the user needs to understand the underlying statistical theory.

5.6.9 Systematic error variance estimates

The variances for this screen should also be imported from the AVE file by keying `F9` (unless you are an Expert user).

Keying `[Ctrl+↵]` allows you to view how the estimates here are combined with the systematic error matrices (see Section 5.6.8 above) to produce the parameter variance matrix. This option is primarily for the benefit of Expert users. Keying `[Shift+Esc]` returns you to the main systematic error variance screen, while keying `[Shift+↵]` either here or in the main screen accepts all the amendments to the group data, saves the revised data for this group on disk and returns you to the groups menu.

5.6.10 Posterior information

! It is primarily in order to *view* the updated data on a group in detail that the user might wish to see these screens when they show posterior information. Standard and Expert users are permitted to edit the group data, but this should only be undertaken with a full appreciation of the statistical consequences.

The sequence of screens when working with posterior information is somewhat modified. The main random error and systematic error variance estimates screens are no longer shown, and the Expert user does not see the systematic error matrices screen either. Instead, the view goes straight into the detailed screens which are only seen after keying `[Ctrl+↵]` when working with prior information. The Expert user finally sees an additional screen showing systematic error parameter estimates.

Any modifications to posterior information do not affect the prior information on a group. This is important when ABLE carries out further updates, since when that updating is done by variance learning ABLE refers to the prior data and *not* to the posterior data from previous updates. In this case, then, any editing of posterior information will be lost. On the other hand, updating without variance learning for a group which has been updated previously ignores the original prior information, and ABLE refers instead to the current posterior information. Changes to posterior information therefore *will* affect any subsequent updates done without variance learning.

5.6.11 ABLE version 1 files

ABLE version 2 stores more group data on the AFS file than ABLE version 1. Nevertheless, ABLE version 1 files can be handled by the Group Data part of the Create/Edit module. ABLE decides whether version 1 AFS files contain prior or posterior information depending on whether there is an associated AUR file (a process which relies upon the user not having divorced an updated file from its AUR file, thereby breaking its audit trail).

A group whose data is stored on the AFS file in version 1 format cannot be updated by variance learning. However, passing through all screens to view or edit a group will, if it is deemed to contain prior information, cause it to be converted to version 2 format so that it may then be updated by variance learning.

5.7 Mappings

5.7.1 Mappings and ID lists

A mapping is a conversion from one ID list to another. To understand this notion, consider the demonstration model in the NEWDEMO.AFS file.The equations are as follows (more details are presented in Table 5-1):

```
// NEW ABLE DEMONSTRATION MODEL

// PART OF A WATER DISTRIBUTION COST MODEL

MODELTOTAL = {DIVCOST
DIVCOST = DIVISION{ZONECOST       // Accumulate divisional totals
ZONECOST = REHABC+PRESSURE+SYSGROW

REHABC = LENTH×REHABUC
LENTH = IRON5+IRON4+(IRON3*DECAY)
IRON5 = exp(logIRON5)
IRON4 = exp(logIRON4)
IRON3 = exp(logIRON3)

PRESSURE = KNOWNPRES+OTHERPRES
KNOWNPRES = PRESMAP{SCHEMES
SCHEMES = exp(logSCHEMES)
OTHERPRES = PRESCOST*(LOWPROPS**0.5)
LOWPROPS = exp(logLOWPROPS)

SYSGROW = SYSGROWS+SYSGROWL
SYSGROWL = BIGINC*(INCDEMAND-CAPACITY)*GROWCL
BIGINC = test(INCDEMAND>CAPACITY)
INCDEMAND = GROWTH+NEWDEMD-LEAKRED
GROWTH = POPEQUIV*GROWRATE
POPEQUIV = exp(logPOPEQUIV)
SYSGROWS = (1-BIGINC)*INCPOS*GROWCS
INCPOS = max(INCDEMAND,0)
```

When specifying the data for groups, an ID list of unit names is given to each base quantity as explained in Section 5.6.1 above. In the demonstration model all base quantities are defined on the same ID list, which is a list of zone names. So each base quantity actually has a value for every zone. In the equation

    GROWTH = POPEQUIV*GROWRATE

POPEQUIV and GROWRATE are multiplied. This means that the value of POPEQUIV in each zone is multiplied by the value of GROWRATE in that zone, and the result is defined to be the value of GROWTH In that zone. Therefore GROWTH also has a value for every zone. It too has an ID list, which is also the Zones ID list.

In fact, every derived quantity has an ID list. An operation of addition (+), subtraction (-) or multiplication (*) in the model equations must combine two quantities defined on the same ID list.

The equation

    DIVCOST = DIVISION { ZONECOST

expresses DIVCOST In terms of ZONECOST through the mapping DIVISION. As stated at the start of this Section, a mapping is a conversion from one ID list to another. DIVISION is defined in the demonstration model as a mapping

From the Zones ID list *Into* an ID list of division names. (The nature of this "conversion" from Zones to Divisions is explained in the next section.) As a result, ABLE deduces that DIVCOST is defined on the Divisions ID list.

The mapping PRESMAP in the equation

```
KNOWNPRES = PRESMAP { SCHEMES
```

is defined in the model as a mapping *from* the ID list of individual schemes *to* the Zones ID list. The base list logSCHEMES is defined on the Schemes ID list, and therefore so is the derived list SCHEMES. Through this mapping, KNOWNPRES is defined on the Zones ID list, as it must be in order to be added to OTHERPRES in the previous equation.

The equation

```
MODELTOTAL = { DIVCOST
```

defines MODELTOTAL through an *unnamed* mapping. This is a special null mapping whose effect is to total the values of DIVCOST over all divisions, so producing a single value. MODELTOTAL therefore takes just one value for the whole company. Its ID list is a null list of one name which is actually blank, since we do not need an ID list to identify the different values of a quantity that only takes one value. The null mapping is always a summation. Its *From* ID list can be any ID list, and its *Into* ID list is the null ID list of one blank name. The null mapping is always available, but you must define each named mapping in the model equations, and the Mappings selection from the Equations/Groups/ Mappings menu allows you to do this.

5.7.2 Vector mappings

The simplest kind of mapping (apart from the null mapping) allocates every name in the *From* ID list to a name in the *Into* ID list, and is defined by specifying this allocation. ABLE also uses vector mappings to define stratifications, since a stratification has this same property of allocating every member of one ID list to a member of another list of stratum names. (Remember that the Create/Edit module also regards lists of stratum names as ID lists.) Stratifications are defined, as described in Section 5.6.2, by setting up this allocation list. One reason why this kind of mapping is called a vector mapping is that it can be defined, as stratifications typically are, by reading a vector of names from an AIF file.

When a named mapping in the model equations is a vector mapping, its effect is to give a value to each name in the Into ID list which is the sum of all the values given to members of the *From* ID list which are allocated to (or "mapped" to) that name in the *Into* list. The DIVISIONS mapping should make the meaning of that complicated sentence clear. DIVISIONS allocates each zone to a division. The value of DIVCOST in a particular division is then the sum of values ZONECOST for all zones in that division. DIVCOST is therefore, as would be expected from its name, the total cost for each division. When the null mapping is then applied to DIVCOST, the resulting MODELTOTAL becomes the company total cost, by summing the divisional totals.

Vector mappings are ideal for this kind of partial aggregation of values.

PRESMAP in the demonstration model is also a vector mapping. It allocates each scheme to a zone. Notice in this case that there are many more zones than schemes. Some zones have two or more schemes allocated to them, so that the KNOWNPRES values for these zones are indeed aggregations of selected SCHEMES costs. But many zones have no schemes allocated to them by PRESMAP, and for these zones the value of KNOWNPRES is therefore identically zero.

5.7.3 Matrix mappings

ABLE allows a more complex form of mapping which is defined by a table (or "matrix") of numbers. The table has a row for each name in the *From* list and a column for each name in the *Into* list. The effect of a matrix mapping in the cost equations is as follows. The value of the derived quantity for a given unit in the *Into* ID list is defined by the corresponding column of the matrix. The values of the quantity on the right of the mapping symbol ({) are multiplied by the numbers in that column, and then the results are summed.

For example, consider an equation

```
DISTCOST = DISTRICT { ZONECOST
```

in which DISTRICT is a matrix mapping. Suppose (to keep the example simple) that ZONECOST is defined on an ID list of 4 zones and DISTRICT maps this to an ID list of 3 districts by means of the matrix:

	Dist1	Dist2	Dist3
Zone1	0.5	0.5	0
Zone2	0	0	1
Zone3	0.2	0.6	0.2
Zone4	0.7	0.3	0

Then the DISTCOST in Dist1 will be 0.5 times the ZONECOST in Zone1, plus 0.2 times the ZONECOST in Zone3, plus 0.7 times the ZONECOST in Zone4.

Matrix mappings are very flexible. The example above represents a situation where Zone1 is deemed to lie half in Dist1 and half in Dist2; Zone2 is entirely in Dist3; and so on. As an example of the generality of matrix mappings, we can note that any vector mapping can be represented as a matrix mapping in which every row contains one value of 1 and all the other members in that row are 0.

5.7.4 The Mappings screen

On selecting Mappings from the Equations/Groups/Mappings menu, you will see the main mappings definition screen. Each row gives the name of a mapping, the name of its *From* ID list and the name of its *Into* ID list. Any mappings defined for stratification are shown, as well as any named mappings appearing in the equations. The stratification mappings will have been defined In the group definition stage, but mappings appearing in the equations must be defined here. A mapping name found in the equations but not yet defined will be shown, but its *Into* list name will be blank. If ABLE can deduce the *From* ID list its name will appear, otherwise that will also be blank.

The *From* and *Into* ID lists can be selected (or changed) in this screen by placing the cursor in the appropriate space and pressing F4 (to select from the ID lists already defined) or F11 (to read a new ID list from an AIF file).

You can then define (or change) or examine any mapping by keying [Ctrl+↵] with the cursor on that row. If the mapping is not yet defined, you will then need to select Vector or Matrix.

For a vector mapping, you then have a new screen with one column of names defining the mapping. You may change names by cycling through the names in the *Into* ID list, or read from an AIF file (in which case you may need to define the correspondence between the words found in the vector and the names in the Into list, which is done by cycling).

For a matrix mapping, the screen shows the whole matrix (and scrolls both ways if necessary). Numbers may be typed in or read from an AIF file. In this case, a special facility is provided to read an entire matrix block from the AIF file to fill the matrix. This is done by keying `[Ctrl+Esc]` with the cursor on the corresponding row of vector names. You can, of course, also read individual columns by selecting an individual vector in the AIF file as usual.

5.8 Worked example

The ABLE software provides a simple demonstration model including the files:

`NEWDEMO.AFS`	An ABLE System File containing the complete prior specification and data for the demonstration model.
`NEWDEMO.AVE`	A variance elicitation file relating to the quantities in the model.
`NEWDEMO.AIF`	An ABLE Input Format file containing the prior estimates and other information.
`NEWSTART.AFS`	An incomplete AFS file containing only the model equations.

The `NEWSTART.AFS` file represents practically a plain sheet on which to experiment with setting up prior information. The `.AVE` and `.AIF` files above provide sources of such information; the `NEWDEMO.AFS` file was itself built from these files.

This Section provides a detailed explanation of how that was done using the Create/Edit module enabling you to reproduce that process exactly.

A listing of the data in `NEWDEMO.AIF` is provided in Appendix B, which we recommend you have conveniently to hand in order to appreciate the effects of your actions. The other information you will need—the model equations and the properties of each base group—is given below.

5.8.1 The model equations

The model represents some typical parts of a water company's model for water distribution investment. The equations are shown in Table 5-1. When you inspect the equations on the screen, the base quantities are shown in blue. In Table 5-1 they are in bold type and are defined in column 2 when they first occur in the equations in column 1.

5.8.2 Properties of the base groups

The groups in this model have the following characteristics. Group names are as used in the `NEWDEMO.AVE` file.

Group name	Strata	Scaled?	Uncertainty levels
`RehabC`		N	Study/Prior
`Lengths`	Grade5/4/3	N	Detail/GIS/Prior
`Decay`		N	Prior
`Pcost`	High/Ave/Low	N	Study/Prior
`Schemes`		N	Detail/Outline
`LowPres`		N	Detail/Local/Prior
`Capacity`		N	Prior
`Growcosts`	Big/Small	N	Study/Prior
`NewDem`		Y	Local projection
`Leak`		Y	Leakage plan
`Growth`	Suburb/Rural/Indust/Comm	N	Prior
`PopEquiv`		N	Prior

Equations	Base Quantity definitions	
MODELTOTAL = { DIVCOST		
DIVCOST = DIVISION { ZONECOST		
ZONECOST = REHABC + PRESSURE + SYSGROW		
REHABC = LENTH * **REHABUC**	REHABUC	*unit cost in £/m to rehabilitate iron mains in poor condition.*
LENTH = IRON5 + IRON4 + (IRON3 * **DECAY**)	DECAY	*proportion of iron mains in service condition 3 that will decay to grade 4 or 5 over the AMP period.*
IRON5 = exp(**logIRON5**) IRON4 = exp(**logIRON4**) IRON3 = exp(**logIRON3**)	logIRON*n*	*Natural logarithm of length, in Km, of iron mains currently in service condition n*
PRESSURE = KNOWNPRES + OTHERPRES		
KNOWNPRES = PRESMAP { SCHEMES		
SCHEMES = exp(**logSCHEMES**)	logSCHEMES	*Natural logarithm of cost, in £K, to carry out individual schemes to remedy low pressure.*
OTHERPRES = **PRESCOST** * (LOWPROPS**0.5)	PRESCOST	*unit cost in £K/property to alleviate inadequate pressure.*
LOWPROPS = exp(**logLOWPROPS**)	logLOWPROPS	*Natural logarithm of number of properties receiving inadequate pressure and not addressed in individual schemes.*
SYSGROW = SYSGROWS + SYSGROWL		
SYSGROWL = BIGINC * (INCDEMAND - **CAPACITY**) * **GROWCL**	CAPACITY GROWCL	*Spare capacity, in Ml/d, to absorb growth without major expenditure.* *Unit cost, in £K/Ml/d, to increase capacity to accommodate increased demand.*
BIGINC = test(INCDEMAND > **CAPACITY**)		
INCDEMAND = GROWTH + **NEWDEMD** - **LEAKRED**	NEWDEMD LEAKRED	*increase in demand due to new customers in Ml/d over the AMP period.* *net reduction in leakage, in Ml/d, over the AMP period*
GROWTH = POPEQUIV * **GROWRATE**	GROWRATE	*Rate of growth, in Ml per thousand population equivalent, due to increasing demand from existing customers.*
POPEQUIV = exp(**logPOPEQUIV**)	logPOPEQUIV	*Natural logarithm of population equivalent, in thousands, of demand from existing customers.*
SYSGROWS = (1-BIGINC) * INCPOS * **GROWCS**	GROWCS	*Unit cost, in £K/Ml/d, to reinforce mains to meet increased demand if within the spare capacity.*
INCPOS = max(INCDEMAND,0)		

The lengths IRON5, IRON4 and IRON3 are believed to be correlated and so are formed into a single group. Similarly, GROWCS and GROWCL form a single correlated group. Other quantities are assumed independent.

Table 5-1: The equations of the demonstration model

5.8.3 Building the AFS file — detailed procedure

There follows the detailed keystroke by keystroke procedure for building the file `NEWDEMO.AFS` from the files `NEWSTART.AFS`, `NEWDEMO.AIF` and `NEWDEMO.AVE`. Basic users cannot carry out this exercise, but might wish to view the data on the file `NEWDEMO.AFS`.

Note that we are going to give *different* names to groups, levels of uncertainty and strata from those used on the AVE file. As noted in Section 4.5 this is *not good practice*, but is done here to show that ABLE can nevertheless extract the corresponding data from the `NEWDEMO.AVE` file when required.

Activity	Keystrokes	Comments
Start ABLE	Double-click icon Enter password for Standard level user (default `STD`) [↵] (or click `OK`)	 *You can change the password here.*
Select AFS file	Click on `Open` button (or select File>Open)	
	Double-click on `newstart.afs`	*You might first need to change the type of file listed from AVE to AFS files.*
Form Groups		
Select Group Data view	Click `Group Data` button (or select View>Group Data) Click OK in warning window	*(Omit if this is already the active view.)* *There is no data to lose.*
Group log iron mains lengths	Cursor to `logIRON5 [space]` Cursor to `logIRON4 [space]` Cursor to `logIRON3 [Ctrl+Insert]` Cursor left and up to `logIRON3` (in Groups column). Type `logIRONLEN`	*Form the correlated group* *Name the group*
Group growth costs	Cursor right and down to `GROWCS` (in Base Quantities column) `[space]` Cursor up to `GROWCL [Ctrl+Insert]` Cursor left to `GROWCL` (in Groups column). Type `GROWCOSTS`	*Form group*
REHABUC group	Cursor to `REHABUC [Ctrl+↵]`	

Unit names	`[F11]`
	Cursor to `NEWDEMO [Ctrl+↵]`
	
	`[Shift+↵]`
	`[Shift+↵]`
		Fill column from AIF file
		Select file
		
		Cursor is on zone already
		Accept

Strata	`[Shift+↵]`	*Unstratified.*

Activity	Keystrokes	Comments
Estimates	[F11] Cursor to ZONE2 [space] Cursor to RehabC [Shift+↵] [Shift+↵]	 *Select column for calibration.* *All estimates 35*
Uncertainty	[Insert] Type Study in top row and Prior in second row. [Ctrl+↵] [Shift+↵] [Shift+↵]	*Add a row as we have two levels of uncertainty (see Section 4.3.5)* *Go in and set levels* *Accept all "Prior"*
Scaling	[Shift+↵]	*Unscaled*
Random errors	[F9] Cursor to NEWDEMO [Ctrl+↵] [Shift+↵] [Shift+↵]	*Import data from AVE file* *Cursor is on* RehabC *already*
Systematic errors	[F9] [Shift+↵]	RehabC *is picked up automatically.*
DECAY group	Cursor to DECAY [Ctrl+↵]	
Unit names	[F4] [Shift+↵] [Shift+↵]	*Go to previously defined ID list for zones. It is important to get all subsequent uses of the same list this way.*
Strata	[Shift+↵]	*Unstratified.*
Estimates	[F11] Cursor to ZONE2 [space] Cursor to DECAY [Shift+↵] [Shift+↵]	
Uncertainty	Type Prior [Ctrl+↵] [Shift+↵] [Shift+↵]	*Only one level of uncertainty.*
Scaling	[Shift+↵]	
Random errors	[F9] Cursor to NEWDEMO [Ctrl+↵] Cursor to DECAY [Shift+↵] [Shift+↵]	
Systematic errors	[F9] [Shift+↵]	
logIRONLEN group	Cursor to logIRONLEN [Ctrl+↵]	
Unit names	[F4] [Shift+↵] Cursor right [Ctrl+End] Cursor right [Ctrl+End] [Shift+↵]	 *Copy into the other two columns.*

Activity	Keystrokes	Comments
Strata	`[Ctrl+↵]` `[Shift+↵]`	*Special stratification by lists.*

Estimates	`[F11]`
	Cursor to `ZONE2 [space]`
	Cursor to `Grade5 [Shift+↵]`
	
	Cursor right `[F11]`
	Cursor to `ZONE2 [space]`
	Cursor to `Grade4 [Shift+↵]`
	Cursor right `[F11]`
	Cursor to `ZONE2 [space]`
	Cursor to `Grade3 [Shift+↵]`
	`[Shift+↵]`
		First set of estimates selected, using calibration.
		
		Get other sets

Uncertainty	`[Insert] [Insert]`
	
	Type `Detail` on top line, `GIS` on second and `Prior` on third
	`[Ctrl+↵]`
	`[F11]`
	Cursor to `ZONE [space]`
	
	Cursor to `GIS [Shift+↵]`
	
	`[Shift+→]` and/or `[Shift+ ←]` to set
	against `True`
	against `Detail`
	`Y` against `GIS`
	`N` against `Prior`
	`[Shift+↵]`
	
	Cursor right `[Ctrl+End]`
	Cursor right `[Ctrl+End]`
	`[Shift+↵] [Shift+↵]`
		Add two rows as we have three levels of uncertainty (see Section 4.3.5).
		
		Select GIS column, with calibration, for uncertainty levels.
		
		Set interpretation of the GIS column in the AIF file.
		
		Copy to other columns.

Scaling	`[Shift+↵]`	

Random errors	`[F9]`
	Cursor to `NEWDEMO [Ctrl+↵]`
	Cursor to `Lengths [Ctrl+↵]`
	
	`[Shift+→]` and/or `[Shift+ ←]` to set
	`Grade5` against `logIRON5`
	`Grade4` against `logIRON4`
	`Grade3` against `logIRON3`
	`[Shift+↵] [Shift+↵]`
		Import data from AVE file
		
		Allocate strata. You will find that here and elsewhere in the demonstration, ABLE has correctly guessed the right allocation and so no cycling is actually needed.

Systematic errors	`[F9] [Shift+↵]`	*Import data from AVE file*
logSCHEMES group	Cursor to `logSCHEMES [Ctrl+↵]`	

Unit names	`[F11]`
	
	Cursor to `Scheme`
	`[Shift+↵] [Shift+↵]`
		A new ID list is needed this time.

Strata	`[Shift+↵]`	*Unstratified.*

Activity	Keystrokes	Comments
Estimates	[F11] Cursor to Scheme [space] Cursor to SCost [Shift+↵] [Shift+↵]	*To calibrate*
Uncertainty	[Insert] Type Detail on top line and Outline on second [Ctrl+↵] [F11] Cursor to Scheme [space] Cursor to Detail [Shift+↵] [Shift+→] and/or [Shift+←] to set against True D against Detail O against Outline [Shift+↵] [Shift+↵] [Shift+↵]	*To allocate levels*
Scaling	[Shift+↵]	
Random errors	[F9] Cursor to NEWDEMO [Ctrl+↵] Cursor to Schemes [Shift+↵] [Shift+↵]	
Systematic errors	[F9] [Shift+↵]	
PRESCOST group	Cursor to PRESCOST [Ctrl+↵]	
Unit names	[F4] [Shift+↵] [Shift+↵]	
Strata	[F11] Cursor to Pstrat [Shift+↵] [Ctrl+↵] [F11] Cursor to Zone [space] Cursor to Pressure [Shift+↵] [Shift+→] and/or [Shift+←] to set H against High A against Average L against Low [Shift+↵] [Shift+↵] [Shift+↵]	*Select names of pressure strata* *Assign interpretation of Pressure column in the AIF file.*
Estimates	[Ctrl+↵] [F11] Cursor to Pcost [Shift+↵] [Shift+↵] [Shift+↵]	*Select estimates by strata* *Go to the AIF file again to get these estimates*

Activity	Keystrokes	Comments
Uncertainty	[F4] Cursor to column with Study and Prior in it [Shift+↵] [Ctrl+↵] [Shift+↵] [Shift+↵]	*Replace column from ID list* *This group has the same levels as REHABUC, so get them this way*
Scaling	[Shift+↵]	
Random errors	[F9] Cursor to NEWDEMO [Ctrl+↵] Cursor to PCost [Ctrl+↵] [Shift+→] and/or [Shift+ ←] to set High against High Average against Average Low against Low [Shift+↵] [Shift+↵]	
Systematic errors	[F9] [Shift+↵]	
logLOWPROPS group	Cursor to logLOWPROPS [Ctrl+↵]	
Unit names	[F4] [Shift+↵] [Shift+↵]	
Strata	[Shift+↵]	*Unstratified.*
Estimates	[F11] Cursor to ZONE2 [space] Cursor to LowProps [Shift+↵] [Shift+↵]	
Uncertainty	[Insert] [Insert] Type Detail on top line, Local on second and Prior on third [Ctrl+↵] [Shift+↵] [Shift+↵]	*Three levels of uncertainty include two kinds of study—Detail and Local—but all prior estimates are at Prior level*
Scaling	[Shift+↵]	
Random errors	[F9] Cursor to NEWDEMO [Ctrl+↵] Cursor to LowPres [Ctrl+↵] [Shift+↵]	
Systematic errors	[F9] [Shift+↵]	
CAPACITY group	Cursor to CAPACITY [Ctrl+↵]	
Unit names	[F4] [Shift+↵] [Shift+↵]	
Strata	[Shift+↵]	
Estimates	[F11] Cursor to ZONE1 [space] Cursor to Capacity [Shift+↵] [Shift+↵]	

Activity	Keystrokes	Comments

Uncertainty	`[F4]`
	Cursor to column with just `Prior` in it
	`[Shift+↵] [Ctrl+↵]`
	`[Shift+↵] [Shift+↵]`	

Scaling	`[Shift+↵]`	

Random errors	`[F9]`
	Cursor to `NEWDEMO [Ctrl+↵]`
	Cursor to `Capacity`
	`[Shift+↵] [Shift+↵]`	

Systematic errors	`[F9] [Shift+↵]`	
GROWCOSTS group	Cursor to `GROWCOSTS [Ctrl+↵]`	

Unit names	`[F4] [Shift+↵]`
	Cursor right
	`[Ctrl+End] [Shift+↵]`	

Strata	`[Ctrl+↵] [Shift+↵]`	*Stratify by lists.*

Estimates	`[Ctrl+↵]`
	Type `100` against `GROWCS`
	`[Shift+↵]`
	Cursor right `[Ctrl+↵]`
	Type `400` against `GROWCL`
	`[Shift+↵] [Shift+↵]`
		Enter estimates nominally by stratification, but in each case only one stratum is relevant. Data are not on the AIF file so are typed in.

Uncertainty	`[F4]`
	Cursor to column with `Study` and `Prior`
	`[Shift+↵] [Ctrl+↵]`
	`[Shift+↵] [Shift+↵]`	

Scaling	`[Shift+↵]`	

Random errors	`[F9]`
	Cursor to `NEWDEMO [Ctrl+↵]`
	Cursor to `Growcosts`
	`[Shift+↵]`
	`[Shift+→]` and/or `[Shift+←]` to set
	`Big` against `GROWCL`
	`Small` against `GROWCS`
	`[Shift+↵] [Shift+↵]`	

Systematic errors	`[F9] [Shift+↵]`	
NEWDEMD group	Cursor to `NEWDEMD [Ctrl+↵]`	
Unit names	`[F4]` `[Shift+↵] [Shift+↵]`	
Strata	`[Shift+↵]`	

Estimates	`[F11]`
	Cursor to `ZONE1 [space]`
	Cursor to `NewDemand`
	`[Shift+↵] [Shift+↵]`	

Activity	Keystrokes	Comments

Uncertainty	`Type LocalProj`
	`[Ctrl+↵]`
	`[Shift+↵] [Shift+↵]`
		One level of uncertainty—local projection of new demand.

Scaling	Cursor to `Yes [Ctrl+↵]`	*Scaled group, default scaling.*

Random errors	`[F9]`
	Cursor to `NEWDEMO [Ctrl+↵]`
	Cursor to `NewDem [Ctrl+↵]`
	`[Shift+↵]`	

Systematic errors	`[F9] [Shift+↵]`	
LEAKRED group	Cursor to `LEAKRED [Ctrl+↵]`	
Unit names	`[F4]` `[Shift+↵] [Shift+↵]`	
Strata	`[Shift+↵]`	

Estimates	`[F11]`
	Cursor to `ZONE1 [space]`
	Cursor to `LeakRed`
	`[Shift+↵] [Shift+↵]`	

Uncertainty	`Type Plan`
	`[Ctrl+↵]`
	`[Shift+↵] [Shift+↵]`
		Only one level of uncertainty

Scaling	Cursor to `Yes [Ctrl+↵]`	

Random errors	`[F9]`
	Cursor to `NEWDEMO [Ctrl+↵]`
	Cursor to `Leak [Ctrl+↵]`
	`[Shift+↵]`	

Systematic errors	`[F9] [Shift+↵]`	
GROWRATE group	Cursor to `GROWRATE [Ctrl+↵]`	
Unit names	`[F4]` `[Shift+↵] [Shift+↵]`	

Strata	Press `[Insert]` 3 times and type the lines
	`Suburb, Rural,`
	`Industry, Commerce`
	`[Ctrl+↵]`
	`[F11]`
	Cursor to `ZONE [space]`
	Cursor to `Type [Shift+↵]`
	`[Shift+↵]`
	`[Shift+↵] [Shift+↵]`
		Find stratification on AIF file
		ABLE correctly allocates correspondence

Estimates	`[Ctrl+↵]`
	Type `0.25` against each stratum name
	`[Shift+↵] [Shift+↵]`
		Allocate by strata, but we actually have the same prior estimate for each.

Activity	Keystrokes	Comments
Uncertainty	[F4] Cursor to column with just `Prior` in it [Shift+↵] [Ctrl+↵] [Shift+↵] [Shift+↵]	
Scaling	[Shift+↵]	
Random errors	[F9] Cursor to `NEWDEMO` [Ctrl+↵] Cursor to `Growth` [Shift+↵] [Shift+↵] [Shift+↵]	*Correspondence is right first time*
Systematic errors	[F9] [Shift+↵]	
logPOPEQUIV group	Cursor to `logPOPEQUIV` [Ctrl+↵]	
Unit names	[F4] [Shift+↵] [Shift+↵]	
Strata	[Shift+↵]	
Estimates	[F11] Cursor to `ZONE1` [space] Cursor to `PopEquiv` [Shift+↵] [Shift+↵]	
Uncertainty	[F4] Cursor to column with just `Prior` in it [Shift+↵] [Ctrl+↵] [Shift+↵] [Shift+↵]	
Scaling	[Shift+↵]	
Random errors	[F9] Cursor to `NEWDEMO` [Ctrl+↵] Cursor to `PopEquiv` [Shift+↵] [Shift+↵]	
Systematic errors	[F9] [Shift+↵]	
Mappings	Click on `Mappings` button (or select View>Mappings). Click OK in warning window	*Data are saved when each group is finished, so nothing will be lost.*
Division mapping ID list	Cursor to `DIVISION` row and to the blank field in the `Into` column [F11] Cursor to `Divisions` (on the bottom line) [Shift+↵]	*Select vector from AIF file (select NEWDEMO file and* [Ctrl+↵] *if necessary)*

Activity	Keystrokes	Comments
Define Mapping	[Ctrl+↵] Cursor to Vector [Ctrl+↵] [F11] Cursor to ZONE [space] Cursor to Division [Shift+↵] [Shift+→] and/or [Shift+ ←] to set Central against C Mid_Shire against MS West against W Annex against A South_Shire against SS East against E North against N [Shift+↵] [Shift+↵]	 *Select vector from AIF file*
PRESMAP mapping ID list	Cursor to PRESMAP row and to the blank field in the Into column [F11] Cursor to ZONE [Shift+↵]	
Define Mapping	[Ctrl+↵] Cursor to Vector [Ctrl+↵] [F11] Cursor to SZone [Shift+↵] [Shift+↵] [Shift+↵]	 *Exit. Select* No *to continue working with this file (e.g.Report), or* Yes *if you wish to open a new file*

ABLE:

Chapter 6 — Designing study programmes

6.1 Introduction

The purpose of the Design module is to help to construct programmes of detailed studies, in such a way as to minimise posterior variance. That is to say, the module analyses the data already stored within the cost model and identifies the areas that would most reward investigation to obtain more accurate estimates.

There may be areas in which you know that more accurate data cannot be obtained, or only at great expense. If so, then the improvement to the overall accuracy of the model must be shown to be commensurately great before such a study can be regarded as feasible. On the other hand, data may have become available as a side-product of some other activity (or that could be arranged to be so) and are therefore virtually free. In this case, no matter how slight the improvement to the model, you will probably wish to insist that the availability of these data be taken into account when identifying areas to be studied.

In any case, you will almost certainly want to place some cost constraints on studies.

Before approaching the Design Module, you should already have decided on some particular areas of interest—you may know, for example, that the accuracy of your estimates in certain parts of the model is particularly poor and can therefore more usefully be improved. You will need to have ascertained the extent to which improvement is possible, represented as a level of uncertainty. You are going to be asking the software for the net effect on the cost model of reducing levels of uncertainty in given parts of the equations. There is little point in establishing that the equations can be greatly improved by even a small improvement in some data that it subsequently transpires cannot be obtained.

The Design Module operates broadly as follows. You first specify one or more types of study that might be performed in any of a collection of zones (zones, catchments, sites etc). You also specify how many studies might be performed, or attach a cost to each possible study and specify a maximum total cost. Finally, you specify which derived quantity is of primary interest. The program then tries to find that combination of possible studies which meets the constraints on number or costs of studies, and which maximises the gain in information about the derived quantity of interest.

Overall, a design request involves:

- specifying one or more *Study Types*
- giving a *Quantity of Interest* for each study type
- setting *Costs* for individual studies of each study type
- optionally forcing the *Inclusion or exclusion* of specified individual studies of any study type
- setting a *Budget.*

! Users at all levels of expertise can design study programmes using this module, except only that Basic level users cannot store Study Types in an AST file. Only Expert level users can design programmes involving more than one study type.

! The Design module needs to carry out similar computations to those performed by the Reporting module. Note, however, that simulation-based computation is not available in the Design module. The setting of the button on the toolbar which selects exact or simulation-based computation in the Reporting module is ignored, and the Design module always attempts to use exact computation. An error message will be generated if the equations defining a Quantity of Interest include operations which are not permitted in exact computation. It is hoped to include the option of simulation-based computation for the Design module in a future release of ABLE.

6.1.1 Designing estimation methods

The Design module will help you to select what zones or assets to study in order to achieve the best effect in terms of reducing the variance of the derived quantity of interest. The amount of reduction of variance that is achievable from a given “budget”, however, depends on the quality of prior estimates and the quality of the study estimates that will result from the selected study programme. It is worth setting out here some important comments in this regard, because in a sense they relate to how one designs the methodologies that lead to prior and study estimates.

For any group, each different level of uncertainty corresponds to a different kind of estimate which can be made of the quantities in that group; see Section 4.2.2. Systematic and random error variances describe the quality of the estimation process which is represented by a given level of uncertainty. Systematic error variances relate to the likely magnitude of systematic bias in the estimation procedure. Random error variances relate to the extent to which the procedure fails to take account of the particular circumstances of each zone or unit which affect the value of the quantity being estimated.

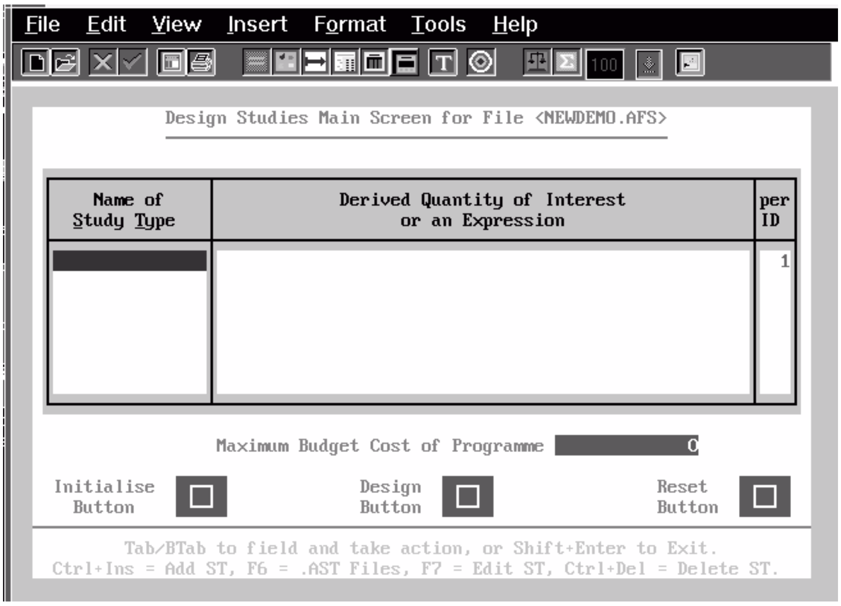
Ideally, an estimation procedure should aim to reduce both kinds of error, random and systematic, as far as possible. In general, one might think of prior estimates having large error variances and detailed engineering studies having much lower error variances. ABLE uses a small number of high quality study estimates to improve the poorer prior estimates which are initially supplied for all quantities in the group. However, to obtain maximum value from ABLE’s updating procedure, the *balances* between systematic and random error variances are important.

The goal should be to reduce *random* error as far as possible in the prior estimates, and for study estimates to minimise *systematic* error. This is by no means straightforward to achieve, and the natural tendency may even be completely the other way around. A simple and obvious way to set prior estimates for many quantities, particularly those that represent unit costs in some sense, is to give the same prior estimate for every zone. An example in the `NEWDEMO` model is the `REHABUC` quantity, whose prior estimate is 35 (£ per metre) in every zone. Such an estimate clearly takes no account at all of the specific features of each zone, and so typically will have large random error. On the other hand, 35 may be quite accurate when averaged over all zones, so the systematic error variance is much smaller. Where possible, this kind of prior estimation process should be avoided. Random error should be reduced wherever possible by thinking carefully about individual zones.

When defining detailed engineering studies, on the other hand, the natural thing to do its to study (in detail) the specific features of the selected zone, in order to obtain an accurate estimate for that zone. This is fine in the sense that it is always good to reduce errors, but the principal effect is to reduce random error. It is vital when designing the procedures for detailed studies to think about reducing systematic error too. Low systematic error variance should be the goal of procedures for study estimates.

6.2 The Design Studies main menu

On entering the Design Module you see the Design Studies main menu:



The options are briefly as follows:

Study Type:	allows you to set up types of study for the study programme
Quantities of Interest:	allows you to select a quantity of interest for each study type.
Per ID	allows you to force certain individual studies to be included in, or excluded from the programme; and to attach a cost to each study.
Maximum Budget:	sets the maximum total cost for the study programme.
Initialise:	After setting up study types, this option invokes the (possibly lengthy) initialisation calculation.
Design:	After setting up all the parameters via the steps above, this option sets the Design Module to search for the best programme of studies.
Reset:	Allows you to reject the option of further studies and design a completely fresh programme. Equivalent to selecting Initialise again, but much faster.

The options are described in detail in Sections 6.2.1–6.2.7. Explanatory text may be given first, the sections describing the actions undertaken at the keyboard are flagged 1-2-3 in the margin.

6.2.1 Study Types

A Study Type consists of

1. one or more Base Quantities (BQs), all defined on the same ID list
2. an achievable level of uncertainty for each quantity

An individual study of that type consists of visiting an individual zone, or asset, in the ID list, and then returning an improved estimate of each BQ at the corresponding level of uncertainty.

The ID list for a BQ specifies the set of zones, catchments, sites etc for each of which that quantity is defined. For example, a BQ [Properties] might be defined on an ID list of zone names, implying that it takes a value for each such zone (the number of properties in that zone). All the selected quantities for a particular study type are defined in the same ID list, with the interpretation that a single study consists of investigating one of the units (i.e. zones, sites or whatever) in that ID list and returning a *study estimate* for the value of each selected quantity in that unit. The accuracy of each resulting study estimate is given by the specified level of uncertainty for each quantity.

For instance, a study type might be specified by selecting the single BQ [Properties] and an associated level of uncertainty “search”, which is the level of uncertainty for an estimate of number of properties based on a thorough search of all relevant registers and maps. A single study of this type would therefore consist of visiting a particular zone, carrying out a properties search and returning the new estimate of [Properties] for that zone.

More often, a study type will consist of more than one selected BQ (e.g. [Properties], [New Properties], [Population], [New connections cost]), with a level of uncertainty for each. A single study will then return new estimates for all the selected quantities.

A programme of studies is specified by stating a collection of individual studies of one or more types. For each study type involved this consists simply of stating which units in its ID list are to be studied. However, designing a programme using more than one study type is available only to Expert level users, because this feature is still experimental in ABLE.

6.2.1.1 Study Types option

When the cursor is in the Study Types field you may:

- define a new type
- fetch a previously saved study type from a file.
- edit a currently defined type
- delete a currently defined type

To define a new type, key `[Ctrl+Insert]`. You are then presented with a screen that allows you to specify Base Quantities. If, however, you press the `F3` key, the model equations are displayed and you can select or deselect BQs (but all must have the same ID list) by tabbing through the equations and pressing the space bar as the required BQs are highlighted. Press `[Shift+↵]` to load the BQs into the table.

The levels of uncertainty are defined by tabbing into the right column of the table and cycling through the possible levels of uncertainty for each quantity. This is achieved by keying `[Shift+↓]` and/or

[Shift+↑]. Note that True is always a possible level of uncertainty, allowing you to define studies which determine the true values of quantities.

To fetch a previously saved study type, press F6. You can then select the appropriate study type from the list displayed by tabbing to the required study type and pressing [Shift+↵].

More than one study type can be saved in a file.

To edit a currently defined type, press F7 when the appropriate name is highlighted in the Study Types field. You can supply a new name for this study type and/or save it in a .AST file.

To delete a Study Type, key [Ctrl+Delete] when the name is highlighted in the Study Types field.

You can quit any screen (abandoning all changes) by keying [Shift+Esc].

! AST files are not only used by the design Module, they also facilitate entry of data to the Update Module. You should save all Study Types that are actually to be used, so that when new estimates are available from these studies they can be passed neatly and easily to the Update Module.

6.2.2 Quantity of interest

The quantity of interest for a Study Type is the derived quantity whose variance, aggregated over the whole ID list of zones/assets, is to be minimised by the design module. It must be defined on the same ID list as the base quantities specified for that Study Type.

In order for the module to be able to identify a best programme of studies, it needs a criterion to optimise. Formally, this consists of stating for each study type a single derived quantity of interest. That quantity must be defined on the same ID list as the BQs specified for a study of that type. The criterion is then to minimise the variance of the sum of all values of the quantity of interest over all units in its ID list.

One point to bear in mind is that the actual posterior variance of any derived quantity after a programme of studies has been carried out is not predetermined but typically depends on the estimates returned by those studies. It is therefore not strictly possible to take this posterior variance as the criterion when designing the studies. The module in fact computes the *preposterior* variance, which is the prior expectation of the posterior variance.

! **Expert users only**: For more than one study type the criterion is the sum of the variance criteria for the different study types. If more than one study type is to be used in the study programme, the resulting criterion may not be meaningful unless the corresponding quantities of interest are independent of each other. In general, the choice of quantities of interest when designing study programmes requires careful consideration and some statistical knowledge.

6.2.2.1 Quantity of Interest option

On tabbing into this option you can either type in an expression for a new derived quantity (on the same ID list as the BQs specified for a study of the type named in the previous field), or you can press F3 to display the model equations.

On pressing `F3`, the next screen shows those model equations that define derived quantities having the right ID list (and only those equations). You can select the derived quantity of interest by tabbing through the quantities on the screen and pressing the space bar to select the required derived quantity.

6.2.3 Per ID

On tabbing into this field, you can key `[Ctrl+↵]` to display a table enabling you to assign costs to each of the individual studies, and to specify forced inclusions and exclusions. The table appears as follows:

Change ID List Information for Study Type <ZONE>

Cost of Study	ID List	Forced Inclusions	Forced Exclusions
1	C1		
1	C2		
1	C3		
1	MS1		
1	MS2		
etc			

Tab, Ctrl/Alt↓↑=Move, Space=pick, CtrlDel=Unpick, F2=Tidy, Shift-Enter=Exit

6.2.3.1 Study costs

A cost may be assigned to each zone/asset in the ID list for each Study Type, to represent the cost of a study of that type on that zone/asset. These need not be real (i.e. pounds and pence) costs, but must reflect the proportions by which costs of various studies vary.

Costs all set to 1 (which is the default) correspond to all studies of that type costing the same. This interacts with the Budget, see below.

You can simply type costs into the Cost of Study column. Other options are:

`[Ctrl+Delete]` resets all costs to the default on entering the Design Module (a column of ones).

`F11` causes the column to be fetched from an AIF file.

`F3` enables you to set the costs to the current estimates of any base or derived quantity. `F3` allows you to select one quantity (which must have the same ID list as the study type) by use of `[Shift+→]` or `[Shift+←]` to scroll through the suitable quantities; or you may type in an expression defining a new derived quantity. For example, to use current estimates of the quantity [Population], but also to have a fixed cost of 10 per zone for setting up, you would type

```
10+Population
```

The current estimates of the selected quantity (as would be reported by the Report Module, see Chapter 8) are then copied into the costs column.

6.2.3.2 Forced inclusions and exclusions

Columns 2, 3 and 4 of the Per ID table represent:

- the complete ID list for this study type
- A list of names from that ID list that are currently selected for forced inclusion in the study programme.
- A list of names from that ID list that are currently selected for forced exclusion in the study programme.

On entering the Design Module both the inclusion and exclusion lists are empty. The inclusion list is reset to empty after any use of the Design Option (see Section 6.2.7).

You may select individual names in the ID list to be copied to the inclusion or exclusion list; or you may select individual names for deletion from either list. There are also two ways of adding collections of names to the inclusion or exclusion list, using a mapping or a list of names saved earlier.

When you have tabbed into the Inclusions or Exclusions column:

`F4` provides a menu of all mappings on the AFS file for which the ID list of this study type is the right ID list.

You are then presented with a menu consisting of the left ID list of the mapping. Selecting one or more names in this list causes all the names in the right ID list which map into the these names on the left ID list to be copied to the inclusion or exclusion list in the main Include/Exclude screen. (In the case of a matrix mapping, any non-zero entry in the relevant row(s) causes that name to be selected.)

`[Ctrl+Insert]` Allows you to “spill” from the ID list into the current column.

`F7` Allows you to use short term memory. The Design Module remembers lists of names, and they may be recalled for later inclusion or exclusion. Whenever you exit from the Include/Exclude or Design options, the currently selected lists of names are saved. Using this key in the include/exclude columns brings up a menu of previously saved lists of names, and selecting one copies that list to the inclusion or exclusion list.

`F2` This feature reflects the fact that inclusion takes precedence over exclusion. Any name appearing in both inclusion and exclusion lists is deemed to be forced inclusion. By pressing `F2` you can tidy a display by removing from the exclusion list any names currently in the inclusion list. On exiting from the Include/Exclude option, and after saving the lists for later reference, this tidying happens automatically (and is therefore apparent if the `per ID` option is selected again).

6.2.4 Budget

The budget is the maximum total cost of a study programme, aggregated over all selected zones/assets in all the current Study Types.

If the default of all studies having unit cost is selected, the budget simply becomes the number of studies to be selected.

At this option, you simply type in a number. The Design option will choose studies with a total cost up to or including this budget. After using the Design option, the budget is reduced by the total cost of the chosen studies.

6.2.5 Initialise option

This option sets the initialisation process in operation. This may take some time and a message to that effect is displayed.

An error is signalled if no study types are defined, or if quantities of interest have been defined for all study types.

This option must be invoked. If not, an error will be signalled when the Design option (Section 6.2.6) is invoked.

6.2.6 Design

This option simply causes the Design Module to try to compute the best programme of studies according to the parameters you have chosen within the options described above (study types, quantities of interest, inclusion and exclusion lists, costs and budget). An error is signalled if the initialisation option has not been used. When the calculations are completed the selected programme of studies is displayed on the screen and may be copied to printer or printer file. During the calculation intermediate results may be displayed, and these can be controlled by a Configuration option (see Chapter 3).

6.2.7 The Reset option

After a programme of studies has been chosen, the Design module will be holding information to reflect the state of information that would obtain after completion of the study programme. Further studies could now be added to the programme by altering some or all of the parameters and choosing another design. You may, however, wish not to choose further studies but to design a completely fresh programme. Choosing the Reset option restores the position from the last initialisation; Reset is equivalent to choosing the Initialise option again, but much quicker.