This chapter contains detailed information on every command of LINEAR.

File

The Files menu is

```
        Load        Edit        Record        Save        Path
```

It is reached from the main menu by choosing '**F**`ile`'. The first menu you see on starting LINEAR with nothing else on the command line is a modified version of this, with no '**S**`ave`' (because there is nothing to save yet) but additional choices '**O**`ptions`' and '**Q**`uit`'.

File

Load

This loads a file into memory. You have a choice:

```
load                Problem                Tab                Dif
```

These are the three types of file that LINEAR can load. Problem files are text files stating the problem. TAB files are binary files produced by the '**S**`ave`' command, containing an exact image of the state of the problem in memory. DIF files are mainly used to communicate with spreadsheet programs. Chapter 3 explains how to set up a problem file or a DIF file to be loaded.

After choosing the type of file, enter the name of the file (see the section **Selecting files** in Chapter 4).

You can enter a problem file directly from the keyboard. To do this, enter '`con:`' when asked for the name of the file, and then type in the problem.

Once the file you specify has been found, LINEAR signals '`Loading` (name of file)', and when loading is complete '`Loaded` (name of file)' and the amount of free memory remaining. You are then returned to the main menu. The `(not solved)` indicator appears on the bottom line to remind you to use '**S**`olve`' (except in the case of a TAB file which was saved after the problem was solved).

The only '**O**`ptions`' setting that is affected by '**L**`oad`' is '**M**`aximize`' or '**M**`inimize`'. The others remain as they were.

File
Edit

This temporarily exits from LINEAR, edits a problem file using the text editor of your choice, and returns to LINEAR, loading that problem file. To use this, LINEAR must be invoked using the batch file `LIN.BAT` rather than directly from DOS (see the section **Batch files and editing** in Chapter 4). First you are asked the name of the file to edit (see the section **Selecting files** in Chapter 4). If you have not specified the name of an editor in the configuration file `LINEAR.CFG` or in the ‘**O**`ptions`’ menu, you are also asked for the name of the editor.

Note that when LINEAR returns after the editor is finished, it does so with a “blank slate” as if it had just been invoked from the command line — any ‘**O**`ptions`’ that had been set before editing will not be remembered, and recording will not be active.

File
Record

This starts or stops sending data to a disk file as well as writing it to the screen. First it will ask you for the name of the output file (see the section **Selecting files** in Chapter 4). Once the file is selected, any further output (with a few exceptions, as noted below) from LINEAR’s commands will be sent to the file as well as to the screen. This will continue until you choose ‘**R**`ecord`’ again or exit from LINEAR with ‘**Q**`uit` **E**`xit`’. Do not remove the disk (if you are recording to a floppy disk) or turn off the computer until you have done one of those, or some data may be lost. Of course, there must be sufficient room on the disk for the data you wish to record.

Thus to make a file containing the values of the basic variables, you could use the following sequence: ‘**R**`ecord`’, name of the file, ‘**V**`alues` **V**`ariables` **R**`ecord`’.

If the output file you choose already exists, you will have a choice:

```
file exists          Append          Rewrite          Try another
```

A`ppend` will cause new output to be added at the end of the current contents of the file.

R`ewrite` will cause the original file to be erased, and a new file prepared for output.

T`ry another` brings you back to the beginning of ‘**R**`ecord`’, leaving the original file intact, so that you can choose another file name.

A letter ‘`R`’ on the bottom line indicates when recording is active.

Menus are not sent to the output file. If ‘**H**`eaders`’ is OFF (this is set by the ‘**O**`ptions`’ command), LINEAR does not send “header lines” such as ‘`Values of basic variables`’ (in ‘**V**`alues` **V**`ariables`’), or other items that precede actual data, to the file. This is useful if you intend to use the file produced by ‘**R**`ecord`’ as input for a database program. ‘**H**`eaders`’ affects recording for the commands ‘**A**`nalyze`’, ‘**T**`ableau` **S**`how tableau`’, and ‘**V**`alues`’.

When all the data you want is in the file, go back to ‘**R**`ecord`’. This will automatically “close” your file, which means that it is safe to remove a floppy disk containing that file. Your file is also closed if you end the program with ‘**Q**`uit` **E**`xit`’. You are asked

```
record to file?                  Yes                 No
```

No will cause the output to go only to the screen, as is the case initially.

Yes will proceed in the same way as the first time you chose ‘**R**ecord’. Note that if you want to continue recording to the same file (e.g. if you chose ‘**R**ecord’ again accidentally), you should choose the same file name as before (it will be the default) and then ‘**A**ppend’.

Instead of a disk file, you can send the output to a parallel printer by entering ‘`prn:`’ or ‘`lpt1:`’ as the file name, or to a serial port by entering ‘`aux:`’ or ’`com1:`’.

You can also echo screen output to LPT1 by using Ctrl PrtSc or setting ‘**P**rinting’ ON in ‘**O**ptions’. This differs from ‘**R**ecord’ in that ‘**H**eaders’ has no effect. Everything except menus will be sent to the printer. A letter ‘`P`’ on the bottom line indicates when printing is on.

File

Save

This allows you to save the current state of the problem into a file.

You first choose the type of file to save: ‘**T**ab’ or ‘**D**if’. A TAB file is the best choice if you intend to ‘**L**oad’ this file later on, while DIF files are useful for communicating with spreadsheet programs. Next you enter the name of the file (see the section **Selecting files** in Chapter 4).

If the file you specify already exists, you are given the choice of ‘**R**ewrite’ or ‘**T**ry another’.

Rewrite will cause the original file to be erased, and a new file prepared for output.

Try another leaves the original file intact and lets you choose another file name.

When saving is complete, LINEAR writes ‘`Saved` (name of file)’ and returns to the main menu.

The only ‘**O**ptions’ setting that is saved is ‘**M**aximize’ or ‘**M**inimize’ (the others are part of the state of the machine, not of the problem itself).

File

Path

This shows and changes the default drive and current directory.

- You are asked for the drive, with the default drive in brackets. Enter the letter of the drive you wish, or just press ↵ to keep this drive.

- You are asked for the directory, with the current directory on the default drive shown in brackets. Specify the directory you wish (as you would in the CHDIR or CD command in DOS, **not** including the drive), or just press ↵ to make no changes. For example, ‘\’ changes to the root directory, ‘`junk`’ changes to a subdirectory ‘junk’ of the current directory, and ‘`..`’ changes to the parent of the current directory.

Solve

This solves the linear programming problem, using the simplex method (see Chapter 2 and Appendix A). When the solution has been found, or the problem has been found to be infeasible or unbounded, a message appears on the screen with the number of pivots and (if the conclusion was an optimal solution) the optimal value of the objective function.

In some problems there may be a very large number of degenerate pivots, and even the possibility of cycling. Degenerate pivots are indicated as such if '**S**how pivots' is ON. If necessary, you can interrupt '**S**olve' by pressing Ctrl Break. Use '**C**hange **A**dd **P**erturbation' to remove the degeneracy before solving the problem. After the problem is solved, you are given the opportunity to remove the perturbation, restoring the original problem.

See the '**O**ptions' command for three settings which affect the operation of '**S**olve': '**M**aximize/**M**inimize', '**S**how pivots' and '**K**eep artificials'.

Values

This command shows you the following menu:

```
Variables        Shadow prices        Ranging        Accuracy
```

Values
Variables

This command lists the objective function and all the basic variables with their values in the current basic solution. Slack variables are indicated with 'SLACK' and the name of the constraint, artificial variables with 'ARTIF' and the name of the constraint. The objective function is listed first, followed by the decision variables in the order in which they were declared, followed by the slack and artificial variables in the order of the corresponding constraints (this order may change, however, if you use the '**C**hange **O**bjective' or '**C**hange **D**elete **C**onstraint' commands). Any variable that is not listed and has not been deleted is nonbasic, and therefore has the value 0.

Values
Shadow prices

This command lists the shadow prices of all variables, including slack and artificial variables, in the current basic solution. See the section 'Sensitivity analysis' in Chapter 2 for how to interpret this information. Slack variables are indicated with 'SLACK' and the name of the constraint, artificial variables with 'ARTIF' and the name of the constraint. The decision variables are listed first, in the order in which they were declared, followed by the slack and artificial variables in the order of the corresponding constraints (this order may change, however, if you use the '**C**hange **O**bjective' command).

Artificial variables which have left the basis while the '**K**eep artificials' option was OFF, and slack or artificial variables for deleted constraints, will not be shown. Otherwise, any variable that is not listed is basic.

Values

Ranging

A menu gives you a choice between two types of ranging:

```
ranging:                    Objective                    Rhs
```

O`bjective` ranging tells how the coefficients of each decision variable in the objective function can vary with the current solution remaining optimal. All the decision variables are listed, in the order in which they were declared (this order may change, however, if you use the '**C**`hange` **O**`bjective`' command).

For example, one line of the output of this command might be

```
X1           1.0000        2.0000        3.2000
```

The middle value 2.0000 is the coefficient of `X1` in the objective function. If this coefficient were replaced by any number between 1.0000 and 3.2000, the current solution would remain optimal. Only the value of the objective function would change.

The assumption is always that the current solution is indeed optimal, and that only one coefficient at a time is to be changed.

R`hs` (or 'Right-hand-side') ranging tells how the constant terms of each constraint can change with the current basis remaining optimal. All the constraints are listed, in the order in which they appeared (this order may change, however, if you use the '**C**`hange` **O**`bjective`' command). For any equality constraint whose artificial variable has left the basis while the option '**K**`eep artificials`' was OFF, this information will be unavailable, with the numbers replaced by '???'.

For example, suppose one constraint of the problem was

```
3 x1 - 4 x7 <= 5 [m1];
```

The corresponding line of output from this command might be

```
M1          3.0000       5.0000       6.0000
```

This would indicate that the same basis would be optimal if the constraint were '`3 x1 - 4 x7 <=` *t*', where *t* is anything from 3 to 6. Changing *t* within this interval could change the values of the objective function and basic variables, but the basic or nonbasic status of the variables would stay the same. Moreover, the shadow price of this constraint would be the effect on the objective value per unit of change in *t* within this interval. For changes outside the interval, the shadow price would no longer be valid.

Values

Accuracy

This provides a check on the numerical accuracy of the current solution. This is especially useful when a large number of pivots has taken place. ‘**`A`**`ccuracy`’ works by reading the problem file again and checking the objective and each constraint against the current tableau. It will only work for a problem that exists in the form of a problem file, not a DIF file.

You are asked for the name of the problem file: specify the same file that was originally used to load the problem. The file is read in. In each row (objective or constraint), every basic variable is expressed in terms of the nonbasic variables as in the current tableau, and the equation is checked to see how well it balances. The name of the objective or constraint, the error in the right-hand-side (i.e. constant term) and the maximum error in the coefficients of variables are listed. Any error less than 10^{-10} is listed as 0.0.

For example, suppose that a constraint of the problem was

```
2 X1 + 3 X2 <= 7
```

which, by including a slack variable called `C_1`, becomes $2X_1 + 3X_2 + C_1 = 7$.

Suppose further that in the current tableau, `X2` and `C_1` are basic, with rows of this tableau corresponding to equations

$$X_2 = 1.3 - 2.1X_1 + 0.3X_3$$
$$C_1 = 3 + 4.35X_1 - 0.99X_3$$

Substituting these in, $2X_1 + 3X_2 + C_1 = 6.9 + 0.05X_1 - 0.09X_3$ (instead of 7). This would be reported as ‘`RHS error`’ of 0.1 and ‘`Max coeff error`’ of 0.09.

Analyze

This performs parametric sensitivity analysis on a variable (see Chapter 2). It assumes that an optimal solution to the problem has already been found.

You are first asked for the name of the variable or constraint to analyze. If this is a decision variable, you have the choice of two types of analysis: ‘**`O`**`bjective`’ and ‘**`A`**`s parameter`’. If it is a constraint, there is the additional choice ‘**`R`**`hs`’.

`O``bjective` analysis considers what happens to the optimal solution when you change the coefficient in the objective of the variable being analyzed.

`A``s parameter` analysis considers what happens to the optimal solution when you treat the variable being analyzed as a parameter (**rhs parameter**) which can be assigned fixed values.

`R``hs` analysis considers what happens to the optimal solution when you vary the right-hand-side of the constraint being analyzed.

In all three types of analysis, you are shown information about the variable or constraint and the interval of change in which the current basis is optimal. Your menu choices include pivoting to an adjacent interval. By repeated pivoting, you can explore all the intervals. After some pivoting in **`A`**`nalyze`, the tableau will no longer be optimal for the original problem. The `(not solved)` indicator comes on when you leave **`A`**`nalyze`, reminding you to restore the optimal solution with **`S`**`olve` before doing anything else (e.g. analyzing another variable). You can not have more than one parameter varying at the same time.

Analyze

Objective

This considers what happens to the optimal solution when you change the contribution to the objective of the variable (say X) being analyzed. It is usually done for decision variables, but can also be done for slack variables.

- If X is basic, you are given its actual contribution to the objective, its current value, and the interval of contributions within which the current basic solution is optimal, with the values of the objective function at the endpoints of that interval. Since the solution does not change within the interval, the change in the objective value is due solely to the new coefficient multiplying the current value of X. The menu shows

```
pivot Higher    pivot Lower    values Up    values Down    Shadow prices
```

If an endpoint of the interval is ±∞, i.e. the variable is at its maximum or minimum feasible value, the corresponding '`pivot`' and '`values`' items will not be present.

`pivot` **H**`igher` pivots to the next interval in the direction that increases X (increasing the objective contribution in a maximization problem, or decreasing it in a minimization problem). You are told which variables enter and leave the basis, the new value of X, the new coefficient interval, and the values of the objective function at the endpoints of that interval. The same menu repeats.

If the problem is unbounded in that interval, no pivot is done, and you are told '`No limit to increase`'.

`pivot` **L**`ower` pivots to the next interval in the direction that decreases X (decreasing the objective contribution in a maximization problem, or increasing it in a minimization problem). You are told which variables enter and leave the basis, the new value of X, the new interval, and the values of the objective function at the endpoints of that interval. The same menu repeats.

If the problem is unbounded in that interval, no pivot is done, and you are told '`No limit to decrease`'.

`values` **U**`p` shows what happens if X is forced to increase, i.e. if $X \geq r$ is required, where r is in the interval between the current value of X and the value it would have if you chose '`pivot` **H**`igher`'. Within this interval, a nonbasic variable (the same one that would enter in '`pivot` **H**`igher`') will increase, and X and the other basic variables will change accordingly. Each of these variables is listed (first X, then the entering variable, then the other basic variables in order), with '`low`' (the value at the current solution), '`high`' (the value at the end of the interval), and '`coefficient`' (the amount of change in that variable per unit increase of X).

`values` **D**`own` is analogous, with X forced to decrease.

S`hadow prices` shows how the shadow prices of the nonbasic variables depend on the contribution of X within the current interval. For example, some of the output might be as follows:

```
Dependence of shadow prices on contribution of X
                 At contribution:
                  4.5000      27.0000    Coefficient
Z                12.2958      12.2958       0.0
V                 9.0000       0.0         -0.4000
```

The column headed 4.5000 shows the shadow prices when the contribution of X is 4.5000, the low endpoint of the current interval. The next column does the same for 27.0000, the high endpoint of the interval. The last column shows the change in each shadow price per unit increase in the contribution of X.

- If X is nonbasic, you are given the shadow price rather than the current value (which is 0). The first ‘pivot **H**igher’ will make X basic. The ‘**S**hadow prices’ command simply shows the shadow prices of the nonbasic variables, which (except for X itself) do not depend on the contribution of X.

In some cases you may want to have an objective parameter multiplying, not just one variable, but some combination of variables. This can be done by adding a new inequality constraint, as explained in the section **Sensitivity analysis** in Chapter 2.

Analyze

As parameter

This considers what happens to the optimal solution when you treat the variable being analyzed as a parameter (**rhs parameter**) which can be assigned fixed values. It is often done with an artificial variable.

- If the variable X being analyzed is nonbasic, you are shown the shadow price, the endpoints of the "parameter interval" of values of X within which the current basis would be optimal, and the corresponding values of the objective. In this type of analysis X is allowed to take negative values. The menu shows

```
pivot Higher      pivot Lower      Values      Shadow prices
```

pivot **H**igher pivots to the next interval of values of X, increasing X.

pivot **L**ower pivots to the next interval of values of X, decreasing X.

These choices will not be present if the interval extends to $+\infty$ or $-\infty$. On a pivot, you are told which variables enter and leave the basis, and given information about the new interval.

Values shows how the objective and basic variables depend on X in the current interval. For example, part of the output might be as follows:

```
Values in interval: low      high     coefficient
X                -3.9803       3.0000       1.0000
OBJECTIVE        59.0789     -26.7495     -12.2958
Y                 6.9803       0.0         -1.0000
C_1               1.0789       1.0789       0.0
```

The ‘low’ column shows the left endpoint of the current interval for X and the values of the objective function, Y and C_1 at that endpoint. The ‘high’ column does the same for the right endpoint. The ‘coefficient’ column shows the change in each variable per unit change in X within the interval.

Shadow prices shows the shadow prices of the nonbasic variables when X is in the current interval. These do not change within the interval.

- If `X` is basic, you are shown its current value and two shadow prices, one for increasing `X` and one for decreasing `X`. The menu shows

```
pivot Higher    pivot Lower    values Up    values Down    Shadow prices
```

If any of the shadow prices is infinite, the corresponding '`pivot`' and '`values`' commands will not be present.

`pivot Higher` pivots to an interval of values of `X` extending above the current value. In this pivot, `X` will leave the basis. `X` will remain nonbasic for the rest of the analysis.

`pivot Lower` does the same, but with an interval extending below the current value.

`values Up` and `values Down` are the same as the '`Values`' command in the nonbasic case, for the intervals that would be obtained with '`pivot Higher`' and '`pivot Lower`' respectively.

`Shadow prices` is the same as in the nonbasic case.

Analyze RHS

This considers what happens to the optimal solution when the constant or "right-hand-side" term of a constraint is changed. For this purpose the constraint is considered as if it had been written in a "standard form", with $\geq$ converted to $\leq$. Thus the RHS of each of the following constraints is 5:

$$x - 5 = y$$
$$x - 5 \leq y$$
$$y \geq x - 5$$

First you are told the original value of the RHS of this constraint. Next LINEAR tells you whether or not the constraint is **binding** — this means that the slack variable is nonbasic in the current solution.

- Changing the RHS of a non-binding inequality constraint will only affect the slack variable for that constraint, as long as that slack variable is positive. You are told the interval in which this is the case, extending from the current value of the "left-hand-side" of the inequality to $+\infty$.

`pivot Lower` pivots to an interval of values of the RHS below the current one, in which the constraint will be binding.

`Values` shows the current values of the objective and all basic variables.

`Shadow prices` shows the current values of the shadow prices of all nonbasic variables.

- If an equality constraint is non-binding in an optimal solution that LINEAR produces, that equality must be automatically true (implied by other equality constraints). Since changing the RHS of this constraint would make the problem infeasible, no further analysis can be done. LINEAR simply signals '`Equality constraint still in basis`', and '`Continue`' returns you to the main menu.

- For a binding constraint, you are given the shadow price of the constraint, the interval of RHS values for which the current basis is optimal, and the values of the objective corresponding to the endpoints of this interval.

`pivot Higher` pivots to the next interval, increasing the RHS, if possible.

pivot **L**ower pivots to the next interval, decreasing the RHS, if possible.

Values shows how the objective and basic variables depend on the RHS in the current interval. For each of these you are shown the values at the two endpoints of the interval and the “coefficient”, i.e. the change in this variable per unit change in the RHS within the interval.

Change

This allows you to change the current problem, by changing an entry of the tableau, adding or deleting a variable, constraint or perturbation, or changing objectives. You see the following menu:

```
change     Entry      Objective      Add      Delete      Cut plane
```

Change

Entry

This command allows you to examine and change an individual entry of the current tableau.

You are asked to identify the entry to alter, entering first the row name (i.e. the objective name, or the basic variable corresponding to the row) and then the column (the nonbasic variable, or ‘RHS’. The current value of this entry is shown, and you are asked for the new value. To keep the current value, just press ↵ . Before returning to the main menu, you are given the opportunity to change more entries. The (not solved) indicator reminds you that the current solution may no longer be optimal, so you should use ‘**S**olve’ again.

Note that this command acts on the current tableau, not the original formulation. It does not affect the values that ‘**V**alues **R**anging’ or ‘**A**nalyze’ use for the objective coefficients or right-hand-side entries, which are set when the problem is loaded. ‘**V**alues **A**ccuracy’ will report “errors”, since the current tableau will not correspond to the original problem.

Change

Objective

This command interchanges a row of the tableau with the objective row. You are asked to identify the row, by its basic variable. For example, suppose the current tableau (arising from our example EXAMPLE1.PRB) is

	x_1	m_2	x_3	
p	-5	15	0	1500
m_1	1	-0.5	0.5	50
x_2	0.5	0.5	0.5	50

‘**C**hange **O**bjective’ with ‘M1’ changes the tableau to

	x_1	m_2	x_3	
m_1	1	-0.5	0.5	50
p	-5	15	0	1500
x_2	0.5	0.5	0.5	50

This corresponds to a problem in which m_1 is the objective, i.e. we wish to maximize the amount of slack time on machine M_1. On the other hand, the original objective p has become an ordinary variable. Like any variable, it is required to be non-negative, so that in effect we have a constraint that $p \geq 0$. Thus the effect of this command is to make the specified variable the objective function, and the original objective a constraint.

If the original objective is to be maximized, as in the example above, the specified will be maximized as well, and the original objective function is required to be ≥ 0. If the original objective was to be minimized, the new objective is to minimize the negative of the specified variable, and the original objective function is required to be ≤ 0.

It is not too likely that you would want to use ‘`Change Objective`’ alone, but combined with the other ‘`Change`’ commands it is a powerful tool for “multi-objective programming”. Thus in our example you might wish to see how much time on machine 1 is required in order to make a profit of $1000. After changing the objective as above, you could ‘`Change Entry`’ for `P` and `RHS` to 100, in effect changing the constraint $p \geq 0$ to $p \geq 1400$, and then solve the new problem.

As another example, to switch to a completely new objective $q = 5x_1 + 3x_2$ and ignore the old one, you could do the following:

- ‘`Change Add Constraint`’ `5 x1 + 3 x2 >= 0 [ q ];`
- ‘`Change Objective`’ `Q`
- ‘`Delete Constraint`’ `P`

This sequence of commands first makes a new row of the tableau, with basic variable `Q`. As slack variable for the constraint, `Q` will be defined as $5x_1 + 3x_2$. Then `Q` is made into the objective, with the original objective `P` being made into a constraint. Finally, that constraint is discarded.

The `(not solved)` indicator reminds you that the current solution may no longer be optimal, so you should use ‘`Solve`’ again.

Change

Add

You are given the choice of adding a constraint, a variable, or a perturbation. If a perturbation has already been added, the last choice is not present.

Add Constraint

You are asked to enter a new constraint from the keyboard. The same format is used as in a problem file (see chapter 3). In particular, you may use arithmetic expressions and any parameters that were defined in the original problem file. The constraint may extend over any number of lines — press ↵ at the end of each line. End the constraint with a semicolon '`;`', as in a problem file.

You may use slack variables as well as original variables in writing your constraint (this is not allowed in a problem file). Thus if `C1` is a slack variable and `X` a decision variable, it would be permissible to write '`C1 + X > 3;`' for the new constraint.

If you examine the tableau after adding a constraint, you may notice that the new row does not look like what you added. If your new constraint involves basic variables, LINEAR expresses them in terms of the nonbasic variables when creating the new row.

If LINEAR finds an error in the new constraint you enter, it will inform you about it as usual, and may display the '`Begin skip`' message. If the line you entered did not have a semicolon on it, LINEAR will ignore what you type until it finds a semicolon. Therefore you should first type '`;`'. You may then enter the correct constraint. Note that pressing ↑ will recall the last line you entered. You may use the cursor keys to go to the beginning of the line, insert the '`;`', and correct the mistake you made before pressing ↵ .

The `(not solved)` indicator reminds you that the current solution may no longer be optimal, so you should use '**`S`**`olve`' again.

Add Variable

This allows you to add a new variable to the problem.

- LINEAR asks for the name of the new variable. This should follow the usual rules. If you want the new variable to be artificial, type '`*`' immediately before the name.
- LINEAR asks for the contribution of the new variable to the objective function. Enter a number.
- LINEAR asks for a constraint involving the new variable. Enter the name of such a constraint. Then LINEAR asks for the coefficient of the new variable in this constraint. Enter a number. This assumes that the constraint is written with variables on the left, and $\geq$ changed to $\leq$. Thus if the original constraint was $x \geq 5$ and including the new variable y makes it $x \geq 5 - 2y$, you would enter '`-2`' (since the constraint in standard form would be $-x - 2y \leq -5$).
- The last step is repeated for every constraint that involves the new variable. When all of these constraints have been done, press ↵ in response to '`Constraint involving ...`'.

The `(not solved)` indicator reminds you that the current solution may no longer be optimal, so you should use '**`S`**`olve`' again.

Add

Perturbation

This command removes degeneracy from a problem and prevents cycling (see Appendix A). It should only be used if '**S**olve' produces a large number of degenerate pivots. A small random amount is added to each entry of the RHS column. A new nonbasic artificial variable named `&&PERT` is added as well, for the purpose of keeping track of these changes through future pivots. After solving the perturbed problem with '**S**olve', you will be given the opportunity to remove the perturbation. This need not be done immediately: you can do it later with '**C**hange **D**elete **P**erturbation'.

This command does not appear on the menu if a perturbation has already been added.

Change

Delete

You are given the choice of deleting a constraint, a variable, or (if '**C**hange **A**dd **P**erturbation' has been used) a perturbation. The distinction between deleting a constraint or a variable really involves the meaning of the word "delete", rather than what is deleted. Deleting a constraint means removing the requirement imposed by that constraint, i.e. allowing the slack variable to take positive and negative values. It is accomplished by removing the corresponding row from the tableau. Deleting a variable, on the other hand, means requiring that variable to be 0, i.e. making the variable artificial.

Delete

Constraint

You are asked for the name of the constraint to delete. The slack variable for this constraint must be basic in order for this to work. The corresponding row of the tableau will be deleted, and the value of the slack variable will no longer be accessible.

- If the slack variable is nonbasic, you must first use '**T**ableau **D**o pivot' to make it basic, pivoting on any nonzero entry in its column. To find candidates for such a pivot, you can use '**A**nalyze **A**s parameter **V**alues' for this slack variable — any variable with a nonzero "coefficient" entry can leave the basis on this pivot.

- If you enter the name of a decision variable rather than a constraint, LINEAR will display a warning message. This is just in case you really intended to use '**D**elete **V**ariable' instead. In that case, you should press Esc to return to the main menu. If you choose '**C**ontinue' the row corresponding to this variable is removed from the tableau. The effect is to allow the variable to take both positive and negative values. However, you no longer have access to the value of this variable.

The `(not solved)` indicator reminds you that the current solution may no longer be optimal, so you should use '**S**olve' again.

Delete

Variable

You are asked for the name of the variable to delete. All that actually happens is that the variable is made artificial, i.e. its value is required to be zero. It may still be accessed in the same way as any other artificial variable. The `(not solved)` indicator reminds you that the current solution may no longer be optimal, so you should use '**S**`olve`' again.

- If the variable you specify is actually a slack variable, LINEAR displays a warning message. This is just in case you really wanted to use '**D**`elete` **C**`onstraint`'. In that case, press Esc to return to the main menu. If you choose '**C**`ontinue`', the effect is to make the constraint into an equality.

Delete

Perturbation

This command only appears on the menu if a perturbation has been added by '**C**`hange` **A**`dd` **P**`erturbation`' and has not yet been deleted. You should do this after using '**S**`olve`' on the perturbed problem. It removes the variable `&&PERT` and changes the RHS column so that the current basic solution is once again a solution of the original problem. '**S**`olve`' gives you the opportunity to issue this command immediately, but if you prefer you can do it later.

There is a slight possibility that this command will result in a basic solution that is not quite optimal. To check this, use '**S**`olve`' or '**T**`ableau` **G**`et candidates`'.

Change

Cut plane

This command is for use in integer programming. The problem should be one where you want all the variables to take integer values (an "all-integer" problem). This includes the slack variables — if the constraints only involve integers, that will be true automatically when the decision variables are integers. The objective, on the other hand, may involve non-integer coefficients.

'**C**`ut plane`' uses the **method of integer forms** to produce a new constraint, known as a **cutting plane**, which would be satisfied by all integer solutions of the problem, but is not satisfied by the current basic solution. Any basic variable can be used for creating such a constraint, if its value in the basic solution is not an integer.

You are asked for the basic variable to use in producing the cutting plane. LINEAR finds the basic variable that would produce the "deepest" cut, i.e. make the perpendicular distance from the current basic solution to the cutting plane as large as possible. It is shown as the default in square brackets, along with its value. You may press ↵ to use it, or enter the name of another basic variable. LINEAR adds the new constraint, and tells you its name ('`C_`' and a number). The `(not solved)` indicator reminds you that the current solution is no longer optimal, so you should use '**S**`olve`' again. You may repeat the process until an optimal solution involving only integers is obtained.

For example, suppose we start with the following problem:

$$
\begin{array}{c}
\text{maximize} \quad y+2z \\
\text{subject to} \\
\begin{array}{rcl}
x- \ y+2z & \le & 7 \\
3x+2y+ \ z & \le & 12 \\
x+ \ y \quad & \ge & 2
\end{array} \\
x \ge 0, y \ge 0, z \ge 0
\end{array}
$$

The optimal solution is $x = 0$, $y = 3.4$, $z = 5.2$, for an objective value of 13.8. However, if we require the variables to be integers this is unacceptable. Therefore we choose '**C**`ut plane`', using the default variable z. Solving the new problem, we obtain a solution with $x = 0$, $y = 3.5$, $z = 5$, for an objective value of 13.5. Since this solution is still not all integers, we use '**C**`ut plane`' again. This time the solution $x = 0$, $y = 3$, $z = 5$ is all integers, so it is the optimal all-integer solution for our problem.

In a very small problem such as this one, a few cuts will usually produce an all-integer solution. In a larger problem it may require a very large number of cuts, so that the method may not be practical at all.

Tableau

This is a group of commands dealing with the tableau and pivoting. The menu is

```
Show tableau         Do pivot         Get candidates
```

Tableau

Show tableau

This command lets you see some or all of the current tableau, in implicit basis format (see chapter 2). For example, here is how it would show the tableau obtained after loading `EXAMPLE1.PRB`:

```
                X1          X2          X3          RHS
    PROFIT   -20.0000    -30.0000    -15.0000      0.0
        M1     1.5000      1.0000      1.0000    100.0000
        M2     1.0000      2.0000      1.0000    100.0000
```

- You are asked to choose '**S**`ome columns`' or **A**`ll columns`'.

S`ome columns` lets you select which columns of the tableau will be shown, and in what order. You enter the list of columns, pressing ↵ after each one. A column is specified by the name of a nonbasic variable or '`RHS`'. When you are done, press ↵ again. Up to six columns are displayed side by side — if more than six are specified, they are shown in groups of six.

A`ll columns` selects all columns of the tableau to display: all nonbasic variables and then the RHS.

- You are asked to choose '**S**`ome rows`' or **A**`ll rows`'.

S`ome rows` lets you select which rows of the tableau will be shown, and in what order. You enter the list of rows, just as in '**S**`ome columns`'. A row is specified by the name of a basic variable or the objective.

A`ll rows` selects all rows of the tableau to display: first the objective, then all basic variables.

Tableau

Do pivot

This allows you to perform a pivot operation of your choice. You are asked to specify the variables entering and leaving the basis. The entering variable must be nonbasic (but may be artificial) and the leaving variable must be basic. No pivot is done if the pivot entry is too close to 0. Otherwise, LINEAR tells you that the pivot was done.

Tableau

Get candidates

This shows you possible candidates for the next pivot to use in solving the problem or obtaining alternative optimal solutions. The criteria for finding these pivots are related to, but not exactly the same as, those that '`Solve`' uses (see Appendix A).

- If there are artificial variables in the basis, you are shown possible pivots leading to their removal from the basis without making any other variables negative. Each basic artificial variable is tried as a "target row" for this purpose. If a basic artificial variable can not be removed from the basis, you are told either '`inconsistent constraint`' (if it has a nonzero value and therefore makes the problem infeasible) or '`can be ignored`' (if its value is automatically zero). The pivot shown may not immediately remove the artificial variable from the basis, but will make its value closer to zero.
- If there are no artificial variables that can be removed from the basis, the next priority is basic variables with negative values. You are shown possible pivots which increase their values without making any variables become negative that were not already negative. Each negative basic variable is tried as a "target row" for this purpose. If the value of a negative basic variable can not be increased, you are told '`Can't make feasible`'.
- If the current tableau is feasible, LINEAR looks at every non-artificial nonbasic variable with a negative shadow price. It tries find a pivot with that nonbasic variable entering the basis, improving the objective function while maintaining feasibility. If no such pivot is found, you are told that the problem is unbounded: it is possible to improve the objective arbitrarily much by increasing the value of this variable.
- If the current tableau is optimal, LINEAR tries for alternative optimal solutions, looking at every non-artificial nonbasic variable with a shadow price of zero. If no pivot is found, the value of this variable can be increased arbitrarily far without affecting feasibility and without changing the value of the objective.
- If the current tableau is optimal and all non-artificial nonbasic variables have positive shadow prices, you are told that the optimal solution is unique.

After looking for pivot candidates, LINEAR asks whether you wish to do a pivot now. '`Yes`' puts you into the '`Do pivot`' command, while '`No`' returns to the main menu.

Options

This command displays a number of settings, and allows them to be changed. Select the setting you wish to change using ↑ or ↓ and ↵ , or by pressing the capital letter.

Decimals is the number of decimal places used in displaying numbers (except in '**C**ut plane'). It only affects the way numbers are displayed, not the precision with which they are stored internally. The number of decimals may range from 0 to 9. However, any number that would not fit in the field assigned to it is written in scientific notation. Thus with '**D**ecimals' set to 2, '**S**how tableau' (which uses a field size of 10) will show '1234567.89' or '-123456.78', but '-1.235E+06' instead of '-1234567.89'. After choosing this option, type a number from 0 to 9 and press ↵ .

Maximize or **M**inimize indicates whether the objective will be maximized or minimized. Choosing this option switches between the two. This option is set when a file is loaded, as indicated in the file.

The remaining settings switch between ON and OFF when you select them.

Show pivots: If this is ON, at each pivot operation you are shown the entering and leaving variables, and the value of the objective function after the pivot. Degenerate pivots are indicated by the word '(degenerate)'. In addition, you are told when the algorithm begins Phases 0, 1 and 2.

— If OFF, this information is not given; instead, at each pivot a period '.' is written on the screen.

Keep artificials: If this is is ON, artificial variables are kept as nonbasic variables when they leave the basis.

— If OFF, they are deleted from the tableau. Since an artificial variable can never re-enter the basis once it has left, this is harmless as far as finding the optimal solution is concerned. Since it reduces the size of the tableau somewhat, it may be helpful in speeding up the solution process, particularly if there are many equality constraints. On the other hand, when an artificial variable is deleted the shadow price is unavailable, and there can be no sensitivity analysis concerning it.

Printing: When this is ON, screen output (except menus) is echoed to the printer (LPT1). This option can also be toggled ON and OFF by pressing Ctrl PrtSc in any other menu.

Headers: When this is OFF, "header lines" such as 'Values of basic variables' are omitted from output sent to a file by the '**R**ecord' command. When it is ON, these are included. OFF may be convenient if the results file will be used as input by a database program, for example.

Editor is the name by which your text editor is invoked by the '**E**dit' command. For example, if your editor is ED.EXE, and you use would use the DOS command 'ED FOO' to edit a file named 'FOO', you should specify the name as 'ED'. A drive and path may occur as part of the name. However, if your editor command is actually a batch file ED.BAT, you must specify the name as 'CALL ED' (and this will only work if you are using DOS 3.3 or higher).

Ok returns to the main menu, leaving the settings as currently shown.

The default initial settings are as follows: '**D**ecimals' 4, '**S**how pivots', '**K**eep artificials' and '**H**eaders' ON, '**P**rinting' OFF, and '**E**ditor' blank. You can change the initial settings of '**D**ecimals', '**S**how pivots', '**K**eep artificials' and '**E**ditor' in LINEAR.CFG (see the section **Configuring** LINEAR in Chapter 4).

Quit

This command ends the LINEAR program and returns you to DOS. You are first given another chance to reconsider: '**C**ontinue' returns to the main menu. Before exiting, LINEAR automatically closes all open files.