LBL-23939 c.2
UC-28

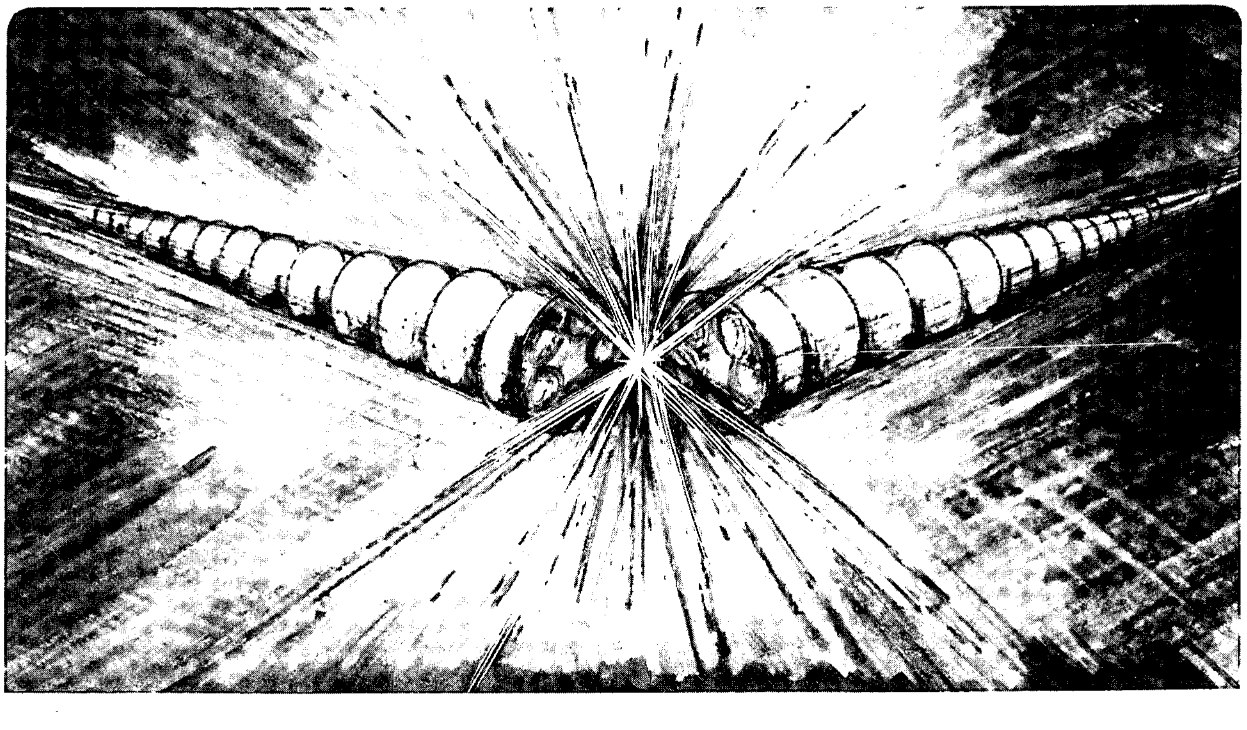
Lawrence Berkeley Laboratory
UNIVERSITY OF CALIFORNIA

Accelerator & Fusion Research Division

LATTICE...A Beam Transport Program

J. Staples

June 1987



LBL-23939 c.2

Prepared for the U.S. Department of Energy under Contract DE-AC03-76SF00098

LBL-23939

LATTICE ... a beam transport program*

John Staples

Lawrence Berkeley Laboratory
University of California
Berkeley, CA 94720

June 1987

*This work was supported by the Director, Office of Energy Research, Division of Nuclear Physics, Office of High Energy & Nuclear Physics, Nuclear Science Division, U.S. Department of Energy under Contract No. DE-AC03-76SF00098

LATTICE

. . . a beam transport program.

John Staples, LBL

June 1987

Contents

Section 1 Introduction

LATTICE is a computer program that calculates the first order characteristics of synchrotrons and beam transport systems. The program uses matrix algebra to calculate the propagation of the betatron (Twiss) parameters along a beam line. The program draws on ideas from several older programs, notably Transport and Synch, adds many new ones and incorporates them into an interactive, user-friendly program.

LATTICE will calculate the matched functions of a synchrotron lattice and display them in a number of ways, including a high resolution Tektronix graphics display. An optimizer is included to adjust selected element parameters so the beam meets a set of constraints. LATTICE is a first order program, but the effect of sextupoles on the chromaticity of a synchrotron lattice is included, and the optimizer will set the sextupole strengths for zero chromaticity.

The program will also calculate the characteristics of beam transport systems. In this mode, the beam parameters, defined at the start of the transport line, are propagated through to the end.

LATTICE has two distinct modes: the *lattice* mode which finds the matched functions of a synchrotron, and the *transport* mode which propagates a predefined beam through a beam line. However, each mode can be used for either type of problem: the transport mode may be used to calculate an insertion for a synchrotron lattice, and the lattice mode may be used to calculate the characteristics of a long periodic beam transport system.

It is the author's opinion that manuals should be as short as possible. We will begin with a sample synchrotron problem and a sample beam transport problem, followed by a description of LATTICE.

Section 2 Sample Synchrotron Calculation

In this section we present a complete calculation of a very simple synchrotron lattice. The lattice parameters are entered into the program (they may also be read from offline storage if they already exist), the lattice parameters are then displayed and the quadrupole gradients are changed to set the machine to a desired tune.

First, LATTICE must be set into execution. To get an executable copy of LATTICE, refer to Section 7 of the manual. Assume that an executable version is in your directory with the filename LATTICE.EXE. Type the input data to your terminal exactly as you see it printed below.

In the following example, the user's input to the keyboard is shown in **Courier Bold Font** and the response from the computer is shown in Courier Listing Font.

First, we will get LATTICE running. Then, the rigidity and emittance of the beam, along with a description of the elements in the lattice and their order is given.

```
run lattice
  Lattice program running...
```

(News items appear in this place)

```
Next
b 3 1 1                                  (beam command)
Beam rigidity =     3.0000 t-m
x,y emittance =     1.0000    1.0000 cm-mrad
dp/p =  0.000%
Next
```

Here, we will enter the characteristics of a few elements into the *element dictionary*. These include two quadrupoles, a short drift space, a bend magnet and a long drift space.

The format of the element description is:

```
    name     type    vary code    p1    p2    p3    x-aper  y-aper
```

where **name** is the name of the element, up to six characters, **type** is the type of element, which includes quads, bends, drifts, sextupoles and several others, which can be abbreviated by its first letter.

The **vary code** indicates whether any parameter of the element is to be made a variable for the optimizer package. The first parameter **p1** is usually the length of the element, the second parameter **p2** is usually the magnetic field or gradient, and the third parameter **p3** depends specifically on the element type. The fourth and fifth parameters define the x- and y-aperture radius, which is used solely in determining the geometrical acceptance of the beam line.

```
e                                    (elements command)
Enter elements and terminate with an 'end'
f q 0 .2 1
d q 0 .2 -2
o d 0 .3
b b 0 2.0944 1.5
s d 0 2.5
end
element list currently contains          5 definitions
Next
```

Next, these elements are given an order in the lattice, and we specify that the lattice consists of 6 periods of this arrangement.

```
l                                    (lattice entry command)
Enter lattice elements and terminate with 'end'
f o b o d d s f end
  8 elements in beam line
Next
p 6                                  (periods command)
machine consists of 6 periods
Next
```

We first print the order and parameters of elements in the lattice period and then print the characteristics of this lattice. In particular we observe the overall circumference and bend angle, the betatron functions at the end of the period, and the tune.

```
d
      name       type  vcode   lth,angle    b,b',s     n,gap   xaper   yaper
      ----       ----  -----   ---------    ------     -----   -----   -----
    1 f          quad   0.0     0.200000    1.000000   0.0000
    2 o          drift  0.0     0.300000    0.000000   0.0000
    3 b          bend   0.0     2.094400    1.500000   0.0000
    4 o          drift  0.0     0.300000    0.000000   0.0000
    5 d          quad   0.0     0.200000   -2.000000   0.0000
    6 d          quad   0.0     0.200000   -2.000000   0.0000
    7 s          drift  0.0     2.500000    0.000000   0.0000
    8 f          quad   0.0     0.200000    1.000000   0.0000
Next
```

```
g                                               (go command)
*
Matched functions for 6 periods
Total length =    35.966  Total bend =  360.001,    0.000 degrees
betax =    4.7021 meters    alphax =  -0.8296    nu x  =    1.740
betay =    4.1584 meters    alphay =   0.0287    nu y  =    1.044
eta x =    2.5859 meters    eta' x =   0.0598    gamtr =    1.562
eta y =    0.0000 meters    eta' y =   0.0000
Next
```

We wish to change the tune ν_x and ν_y to 1.5 in both planes. First, we must specify which components of which elements will be modified during the optimization process. We choose the gradients of the quadrupoles, which are labeled "F" and "D". Then we enter the fit parameters: we wish the tune in both planes to be 1.5. Finally we ask LATTICE to iterate to a solution. The program provides us with information for each step of the solution, and summarizes the changes in the fitted values (the tunes), as well as the changes in the quadrupole gradients.

```
v                                               (vary command)
  Enter pairs of names and par types, terminate with "end"
f b d b end
  2 parameters made variable
Next
f                                               (fit command)
Enter fit functions, target value:
nux 1.5
nuy 1.5
end
Fit parameters for a machine of  6 periods:
variable  target value
    nux         1.5000
    nuy         1.5000
Next
i                                               (iterate command)
Error criterion is 0.00010, relaxation parameter is 1.000
Step  item       val           target       difference
 0     nux      1.74023       1.50000        0.24023
 0     nuy      1.04382       1.50000       -0.45618

 1     nux      1.48674       1.50000       -0.01326
 1     nuy      1.44012       1.50000       -0.05988

 2     nux      1.50168       1.50000        0.00168
 2     nuy      1.50228       1.50000        0.00228

 3     nux      1.50000       1.50000        0.00000
 3     nuy      1.50000       1.50001        0.00001

converged in   3 iterations
variable    initial value    final value    target value
    nux          1.74023        1.50000        1.50000
    nuy          1.04382        1.50001        1.50000

initial and final values of element parameters
element  par  vary code  initial value  final value
 f        b'       1.2       1.000000     0.659969
 d        b'        2.2       -2.000000    -2.866538
Next
```

We see that using the optimizer involves three steps: the variables must be set (the gradient of two quadrupoles in this case), the fitting parameters must be specified (ν_x and ν_y) and then the optimizer invoked. We can solve for as many as 10 variables, and the solver can be tailored to work under difficult conditions.

We take another look at the overall lattice parameters and notice that the tunes are now what we desire. The betatron functions as well as the other matched functions at the end of the sector have also changed.

```
g
*
Matched functions for 6 periods
Total length =    35.966  Total bend =  360.001,    0.000 degrees
betax =    5.3093 meters    alphax =  -0.8813   nu x   =    1.500
betay =    2.6742 meters    alphay =   0.0362   nu y   =    1.500
eta x =    3.4845 meters    eta' x =   0.1909   gamtr =    1.319
eta y =    0.0000 meters    eta' y =   0.0000
Next
```

To get a more detailed listing of the betatron functions at the end of each element in the period, we cycle through all the elements.

```
c                                              (cycle command)
*
elemt   lth   sum l   betax alphax etax  eta'x   psix betay  alphay  etay  eta'y  psiy
        (m)    (m)     (m)               (m)  (rad)  (deg)  (m)               (m)  (rad)  (deg)
        0.     0.      5.31  -0.88  3.48   0.19    0.    2.67    0.04  0.00  0.00    0.
f       0.20   0.20    5.63  -0.70  3.51   0.04    2.1   2.70   -0.16  0.00  0.00    4.3
o       0.30   0.50    6.07  -0.78  3.52   0.04    5.0   2.83   -0.27  0.00  0.00   10.5
b       2.09   2.59    3.66   1.47  2.82  -0.64   26.6   5.63   -1.07  0.00  0.00   42.2
o       0.30   2.89    2.86   1.21  2.63  -0.64   31.9   6.30   -1.18  0.00  0.00   45.1
d       0.20   3.09    2.50   0.57  2.55  -0.15   36.2   6.54    0.01  0.00  0.00   46.9
d       0.20   3.29    2.39   0.02  2.57   0.34   41.0   6.29    1.20  0.00  0.00   48.6
s       2.50   5.79    4.93  -1.03  3.43   0.34   87.8   2.73    0.23  0.00  0.00   85.7
f       0.20   5.99    5.31  -0.88  3.48   0.19   90.0   2.67    0.04  0.00  0.00   90.0
        nu x          =   1.500       nu y          =    1.500
        tr gamma      =   1.319       compaction    =  0.57493
        beta x min    =   2.387       beta y min    =    2.674
        beta x max    =   6.070       beta y max    =    6.537
         eta x min    =   2.553        eta y min    =    0.000
         eta x max    =   3.518        eta y max    =    0.000
Next
```

We desire a high resolution plot of the lattice functions. This is done with an off-line support program LATGRAPH which reads a data file produced by LATTICE. We generate this file next.

```
gx                                             (graph external command)
Graphfile name? => test1
Next
```

A file ''test1.gx'' is generated which contains information to be processed by the offline graphics processor program LATGRAPH. Finally, we terminate operation of LATTICE.

```
quit
End of program
```

The output of the graphics processor program LATGRAPH is included here at reduced scale. Figure 1 shows the beam size and the Figure 2 shows the betatron amplitudes in one period of the lattice. To see the physical layout of the lattice, another offline program LATDRAW is used to produce the Figure 3. Refer to Section 6 for details on these graphics processor programs.

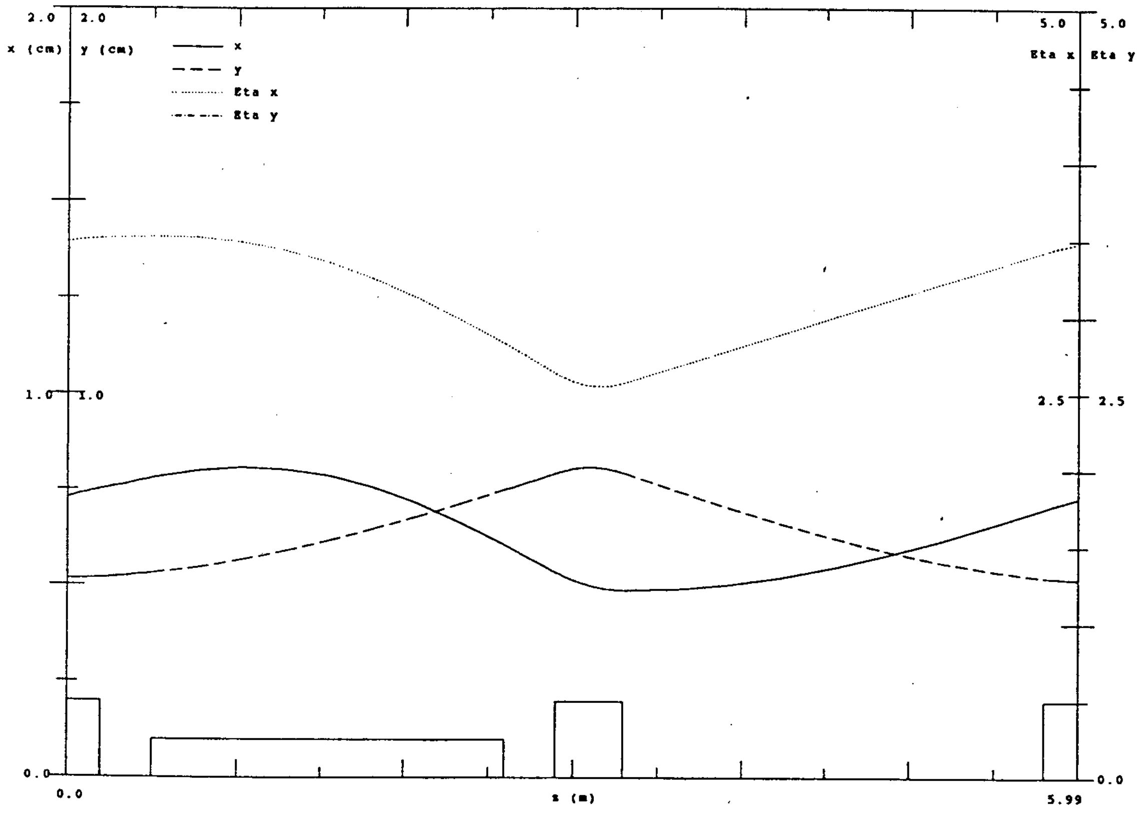


Figure 1. Beam size in one period.

There are several things to notice here. The elements types are entered into an *element dictionary*. Then the elements are arranged in order through the *lattice list*. A few more parameters such as the beam rigidity and the number of times this period of the lattice is repeated for the whole synchrotron are also entered.

When the basic information is in, commands can be issued to calculate various characteristics of the lattice. We saw how easy it is to specify the parameters to be optimized.

A large number of additional commands are available, for editing and storing the lattice, both in internal storage registers and on external disk files, for producing various types of graphical output, both online and offline, and other operations.

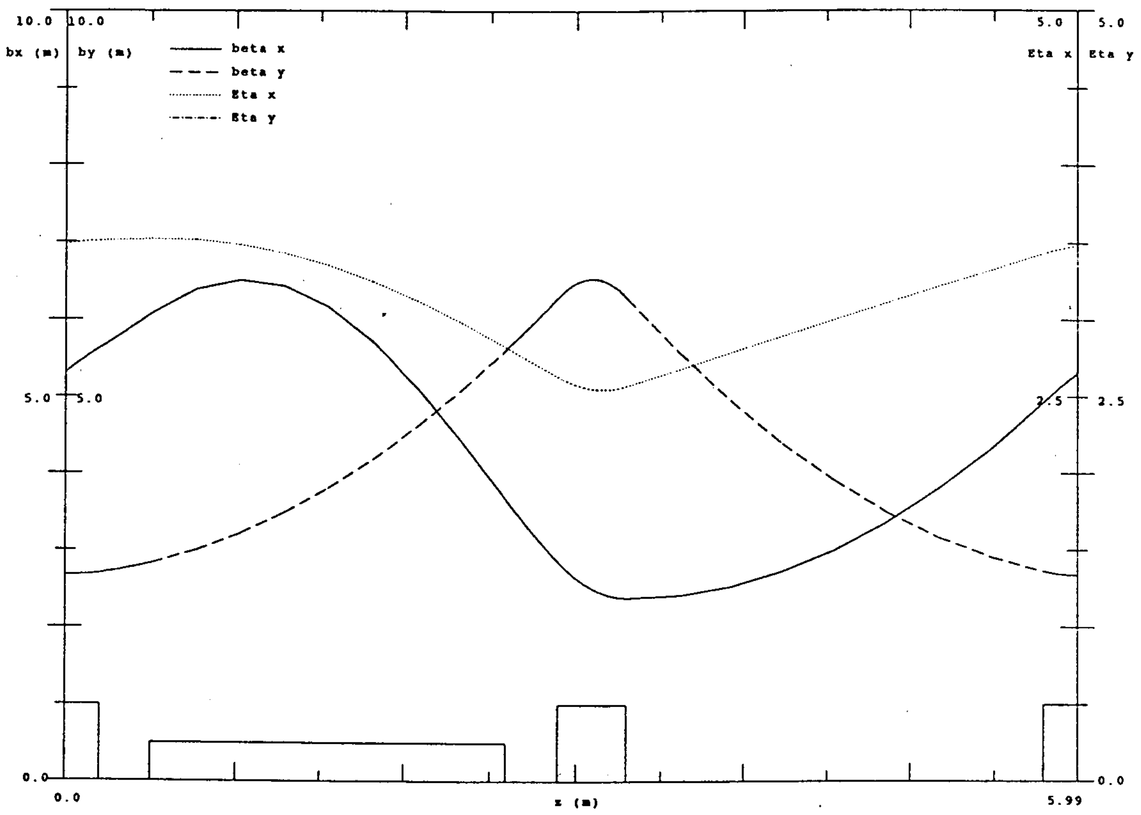


Figure 2. Betatron amplitudes in one period.

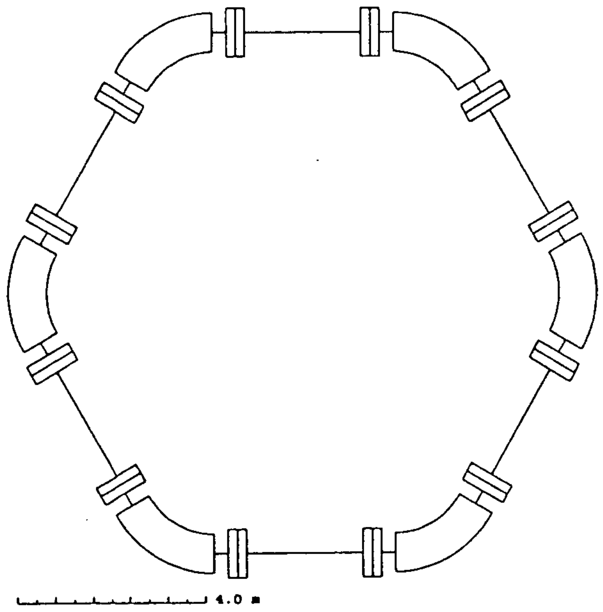


Figure 3. Lattice layout.

Section 3 Sample Transport Calculation

In this section a simple transport problem is demonstrated. The beam line consists of a 1 meter drift, a quadrupole doublet and a 5 meter drift. We will introduce some new commands in this section.

```
run lattice
  Lattice program running...
```

(News items appear in this place)

```
Next
b 1 1 1                              (beam command)
Beam rigidity =     1.0000 t-m
x,y emittance =     1.0000   1.0000 cm-mrad
dp/p =  0.000%
Next
e                                    (element entry command)
Enter elements and terminate with an 'end'
l1 d 0 1
qf q 1.2 .5 1
s d 0 .1
qd q 2.2 .5 -1
l2 d 0 5
end
element list currently contains          5 definitions
Next
l                                    (lattice entry command)
Enter lattice elements and terminate with 'end'
l1 qf s qd l2
end
  5 elements in beam line
Next
```

The default mode for LATTICE is the *lattice* mode, where the matched beam functions are determined. In the *transport* mode, the initial beam parameters are introduced at the beginning of the beam line. Entering these parameters automatically places the program in the *transport* mode.

```
t 10 0 10 0                                    (transport command)
Transport mode
        betax =  10.0000   alphax =   0.0000
        betay =  10.0000   alphay =   0.0000
        etax  =   0.0000   eta'x  =   0.0000
        etay  =   0.0000   eta'y  =   0.0000
Next
g                                              (go command)
*
Total length =    7.100  Total bend =    0.000,    0.000 degrees
betax =   6.0861 meters    alphax =  -0.7461
betay =   8.3070 meters    alphay =   0.5011
eta x =   0.0000 meters    eta' x =   0.0000
eta y =   0.0000 meters    eta' y =   0.0000
x     =   0.7801 cm        x'     =   1.5993   r12    =  0.5980
y     =   0.9114 cm        y'     =   1.2272   r34    = -0.4480
m(1,6)=   0.0000           m(2,6) =   0.0000  m(5,6)  =  0.0000
m(3,6)=   0.0000           m(4,6) =   0.0000
Next
```

We see here that the **go** command produces a different sort of output. We would like to tune the two quadrupoles to produce a beam whose radius is 1 cm in both planes. We have already entered vary codes for the two quadrupoles in the original data, so all we need to do here is to specify the fitting parameters and invoke the optimizer.

```
f                                              (fit command)
Enter fit functions, target value:
x 1
y 1
end
Fit parameters for a machine of  1 periods:
variable  target value
    x          1.0000
    y          1.0000
Next
i                                              (iterate command)
```

(The convergence information is omitted here)

```
converged in   3 iterations
variable    initial value    final value   target value
    x           0.78013        1.00003       1.00000
    y           0.91143        1.00001       1.00000

initial and final values of element parameters
element   par  vary code  initial value  final value
 qf       b'      1.2        1.000000     0.687450
 qd       b'      2.2       -1.000000    -0.729337
Next
```

```
g
*
Total length =     7.100  Total bend =     0.000,    0.000 degrees
betax =  10.0007 meters     alphax =  -0.7160
betay =  10.0002 meters     alphay =   0.2146
eta x =   0.0000 meters     eta' x =   0.0000
eta y =   0.0000 meters     eta' y =   0.0000
x     =   1.0000 cm         x'     =   1.2298   r12    =  0.5822
y     =   1.0000 cm         y'     =   1.0228   r34    = -0.2099
m(1,6)=   0.0000            m(2,6) =   0.0000  m(5,6) =  0.0000
m(3,6)=   0.0000            m(4,6) =   0.0000
Next
```

Let us produce a crude terminal-independent printer plot of the beam size along the transport system. The name of the element is shown vertically along the top of the graph, and the element type along the bottom. We have entered two parameters after the **gr** command to set the x- and y-scales to 2 cm.

```
gr 2 2                                          (graph command)
|1111111111qqqqqsqqqqqlllllllllllllllllllllllllllllllllllllllllllllllllllll|
|1111111111fffff ddddd22222222222222222222222222222222222222222222222222222|
|                                                                          |
|                                                                          |
|                                                                          |
|                                                                          |
|                                                                          |
|                                                                          |
|               yyyyyyyyyyyyyyyyyyyyyyyyyyyyy                              |
|          yyyyyy                            yyyyyyyyyyyyyyyyyyyyyy        |
|yyyyyyyyyyxxxxx                                            xxxxxxxyyyyyyyy|
|               xxxxxx                       xxxxxxxxxxxxxxxx              |
|                     xxxxxxxxxxxxxxxxxxxxxxxx                             |
|                                                                          |
|                                                                          |
|                                                                          |
|                                                                          |
|                                                                          |
|                                                                          |
|                                                                          |
|----------qqqqq-qqqqq-----------------------------------------------------|
 x, y, eta scales :  2.00, 2.00, 5.00, 5.00, lth =     7.10, dp/p =  0.00%
Next
```

If you have a Tektronix 4010/4014 compatable terminal, you can get a medium-resolution plot by typing **gt**. The screen will be cleared and a graph written. To continue, hit the **<cr>**. Some brands of terminals will now still be in the Tektronix non-scrolling mode and must be placed back into text mode at this point by the user.

Next, let us change the length of the final drift 12 from 5 to 6 meters. We will then recalculate the quadrupole strengths so that the beam size is 1 cm radius in both planes at the end of the modified beam line.

The alter command takes three arguments: the first one is the name of the element to be altered, the second one the parameter, the length in this case, and the third one is the value which replaces the old one.

```
a 12 1 6                                  (alter command)
Next
i                                         (iterate command)
converged in   2 iterations
variable    initial value    final value   target value
    x             1.07628       1.00001       1.00000
    y             0.98364       1.00000       1.00000

initial and final values of element parameters
element  par  vary code  initial value  final value
 qf      b'       1.2       0.687450     0.754555
 qd      b'       2.2      -0.729337    -0.776568
Next
g                                         (go command)
*
Total length =     8.100  Total bend =    0.000,    0.000 degrees
betax =  10.0003 meters    alphax =  -0.8098
betay =  10.0000 meters    alphay =   0.1353
eta x =   0.0000 meters    eta' x =   0.0000
eta y =   0.0000 meters    eta' y =   0.0000
x     =   1.0000 cm        x'     =   1.2867   r12   =  0.6293
y     =   1.0000 cm        y'     =   1.0091   r34   = -0.1341
m(1,6)=   0.0000           m(2,6)  =  0.0000  m(5,6)  =  0.0000
m(3,6)=   0.0000           m(4,6)  =  0.0000
Next
```

One of the simplest lattice editing function is reversing the order of the entire lattice. We do this here. After reversing the lattice, we print out the order of elements in the new lattice.

```
l r                                       (lattice reverse command)
Next
d                                         (data list command)
      name       type  vcode   lth,angle    b,b',s      n,gap   xaper   yaper
      ----       ----  -----   ---------    ------      -----   -----   -----
   1 l2         drift  0.0     6.000000    0.000000    0.0000
   2 qd         quad   2.2     0.500000   -0.776568    0.0000
   3 s          drift  0.0     0.100000    0.000000    0.0000
   4 qf         quad   1.2     0.500000    0.754555    0.0000
   5 l1         drift  0.0     1.000000    0.000000    0.0000
Next
g                                         (go command)
*
Total length =     8.100  Total bend =    0.000,    0.000 degrees
betax =  21.7511 meters    alphax =   0.2131
betay =   8.7439 meters    alphay =   0.0154
eta x =   0.0000 meters    eta' x =   0.0000
eta y =   0.0000 meters    eta' y =   0.0000
x     =   1.4748 cm        x'     =   0.6933   r12   = -0.2084
y     =   0.9351 cm        y'     =   1.0695   r34   = -0.0154
m(1,6)=   0.0000           m(2,6)  =  0.0000  m(5,6)  =  0.0000
m(3,6)=   0.0000           m(4,6)  =  0.0000
Next
```

As we have reversed the order of the beam line elements, the value of the beam parameters at the end of the beam line has changed. To test the optimizer, we will ask it to retune the quadrupole gradients to give the same beam size (x = y = 1.0 cm radius) at the end that we had with the original beam line. The fitting functions have already been entered and the quadrupole gradients have already been specified as variables.

```
i
converged in   3 iterations
variable    initial value    final value   target value
    x           1.47483       0.99997       1.00000
    y           0.93509       0.99998       1.00000

initial and final values of element parameters
element  par  vary code  initial value  final value
 qf      b'      1.2       0.754555      1.702501
 qd      b'      2.2      -0.776568     -1.184664
Next
g
*
Total length =    8.100  Total bend =    0.000,   0.000 degrees
betax =   9.9994 meters    alphax =   5.1559
betay =   9.9995 meters    alphay =  -1.5627
eta x =   0.0000 meters    eta' x =   0.0000
eta y =   0.0000 meters    eta' y =   0.0000
x     =   1.0000 cm        x'     =   5.2521   r12    = -0.9817
y     =   1.0000 cm        y'     =   1.8554   r34    =  0.8423
m(1,6)=   0.0000           m(2,6) =   0.0000  m(5,6) =  0.0000
m(3,6)=   0.0000           m(4,6) =   0.0000
Next
gr

|lllllllllllllllllllllllllllllllllllllllllllllllllllllllqqqqsqqqqqlllllllll|
|2222222222222222222222222222222222222222222222222222222dddd fffff111111111|
|                                                                          |
|                                                                          |
|                                                            xxxxxxx       |
|                                                           x       x      |
|                                                         xxx        x     |
|                                                       xx            xx   |
|                                               yyyyyyyyyy              x  |
|                       yyyyyyyyyyyyyyyyyyyyyyyyy         yy             xx|
|yyyyyyyyyyyyyyyyyyyyyyyy                                   yyyyyyyyyyyyyyy|
|                                                                 y        |
|                                                                          |
|                                                                          |
|                                                                          |
|                                                                          |
|                                                                          |
|                                                                          |
|                                                                          |
|                                                                          |
|-------------------------------------------------------qqqq-qqqqq---------|
 x, y, eta scales :  2.00, 2.00, 5.00, 5.00, lth =     8.10, dp/p =  0.00%
Next
quit
End of program
```

We have introduced the concept of altering element values, and of editing the lattice definition. A crude terminal independent graphical display is demonstrated.

Section 4 Description of Program

In this section we describe LATTICE. Its basic structure is outlined and a little theory is given.

Units

The units used in this program are MKSA. There are a few exceptions however. The following table indicates the standard units:

Input Parameters

length	meters
magnetic field	Tesla
quadrupole gradient	Tesla/meters
magnet edge angles	degrees
rigidity	Tesla-meters
emittance	π cm-mrad, unnormalized
momentum spread	±%
betatron amplitude	meters
dispersion vector	meters

Output Parameters

beam radius (x,y)	cm
beam divergence (x′,y′)	mrad
phase advance	degrees

If the rigidity is entered in kGauss-meters or Gauss-meters, then all magnetic fields are entered in a consistent set of units. However, for the electron storage ring functions to be calculated correctly, Tesla-meters should be used.

Beam Line Definition

A synchrotron or beam transport system consists of a *lattice* of *elements,* that is, a sequence of magnetic elements, drift spaces, and other devices which act on the beam. In a beam transport system, the beam starts at one end of the lattice (beam line) and traverses each element once until it emerges from the other end. The configuration of elements can be more or less arbitrary although only certain combinations of elements will result in efficient transport of the initial beam to the end of the beam line.

In a synchrotron, however, the beam recirculates periodically through the lattice of elements. In order that the beam orbit remain stable, additional constraints are placed on the configuration on the elements so the amplitude does not grow continuously. If these constraints are met, a unique beam envelope function around the ring is defined which is called the *matched function* for the lattice. The matched functions exist in both transverse planes, so no beam information other than the rigidity, emittance and momentum spread need be supplied to determine the beam size.

To enter elements into the element dictionary, the following procedure is used. For example, three elements are to be used, two drifts called `l1` and `l2` and a quadrupole called `q1`. The element dictionary entry is opened, the elements read in, and the dictionary closed by entering

```
e                              (opens the element dictionary)
l1 d 0 1
l2 d 0 3.4
q1 q 0 2 1.2
end                            (closes the element dictionary)
```

Each element has a name, followed by its type, a vary code and three parameters. The element types, which is usually abbreviated by its first letter, are as follows:

`drift`	(drift length)
`quad`	(quadrupole magnet)
`bend`	(horizontal bend magnet)
`edge`	(edge of a horizontal bend magnet)
`lens`	(arbitrary linear lens)
`matrix`	(arbitrary first-order matrix)
`sextupole`	(a drift to first order, controls chromaticity)
`BEND`	(vertical bend magnet)
`EDGE`	(edge of a vertical bend magnet)

The detailed definition of each element is found in section 5. The vertical bend magnet adds to the total vertical angle. It and the vertical edge element are the vertical bending counterparts of the normal horizontal magnet and edge elements. More elements can be added to the element dictionary at any time. The dictionary can hold up to 100 definitions.

The order that the elements are ordered into a lattice is then entered. If, for example, the lattice consisted of four elements, consisting of the first drift, then the quad, then the second drift and then the quad again, one would enter

```
l                       (opens the lattice list for entry)
l1 q1 l2 q1
end                     (closes the lattice list)
```

which defines a lattice to comprise a sequence of four elements. Notice that the quadrupole is used more than once.

The beam rigidity, emittance and relative momentum spread are entered. The `beam` command has the following parameters:

```
beam  Rigidity   x-emittance   y-emittance   momentum spread
```

Printing the Lattice List

The contents of the lattice list can be shown with the

```
d   or  data
```

command. This shows the elements of the lattice in order, along with a sequence number which is useful in lattice editing.

A short printout of the lattice list or of the element dictionary is had by typing

```
    l l
```

or

```
    e l
```

Modes

LATTICE operates in one of two modes, the *lattice mode* , the default mode which requires no definition of initial beam parameters, and the *transport mode* in which the initial beam betatron parameters are specified. In the lattice mode, the matched beam functions are determined for the beam line which is contained in the working registers. It is easy to switch between the two modes to calculate synchrotron insertions, for example, or to optimize the parameters of a long periodic transport line.

To operate in the *transport* mode instead of the default *lattice* mode, the initial set of betatron parameters is entered with the command

t $\beta_x \quad \alpha_x \quad \beta_y \quad \alpha_y \quad \eta_x \quad \eta'_x \quad \eta_y \quad \eta'_y$

An alternative way to enter beam size information is discussed in Section 5.

Working Area

The program operates by defining a series of elements, which are contained in the *element dictionary* and the order of these elements contained in the *lattice list*. Some additional information is required, such as the rigidity Bρ, the x- and y-emittances and the relative momentum spread. Figure 4 shows the configuration of registers in the working area.

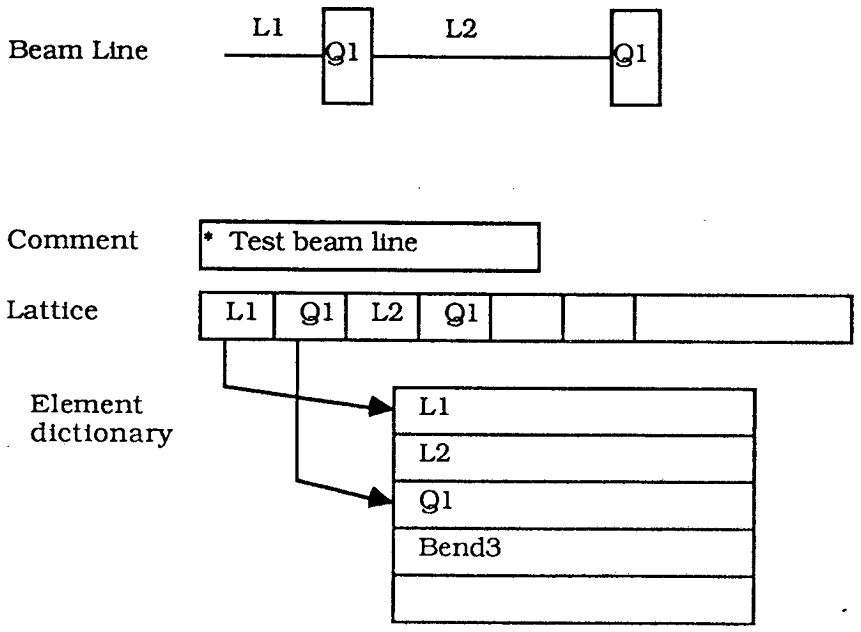


Figure 4. Register configuration in working area.

Save Registers

LATTICE is organized into a *working area* which contains the element dictionary, the lattice list, and six other registers which include the additional beam information, graph scales, fitting parameters and the like. In addition, each of these registers has six counterpart *save registers* which can be used to save intermediate results. Data can be move between the working area and any of the save registers selectively or as an entire group. It is possible to store six completely independent problems in the program with a seventh in the active working area. Figure 5 shows the relationship between the element dictionary and lattice list with their save registers.

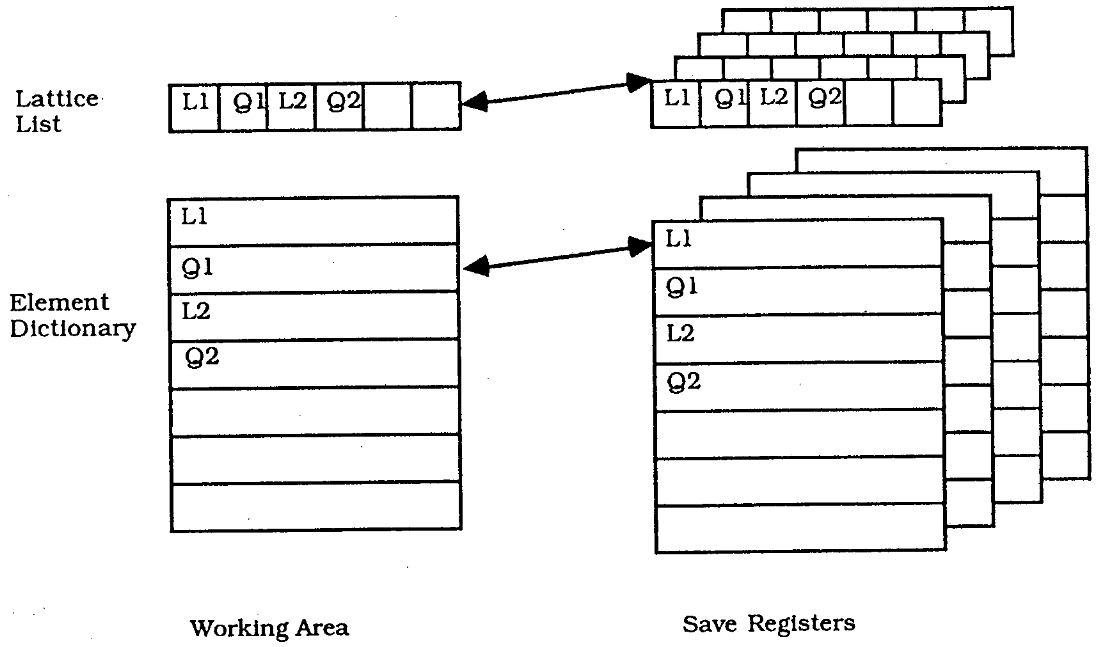


Figure 5. Working area and save registers.

To move the working element dictionary, for example, to the element dictionary save register number 4 (out of 6), execute a save command with a suffix **e** by typing

```
s 4 e
```

There are counterpart save and recall commands for each of the eight registers in the working area. All 8 working area registers can be simultaneously saved into save registers group 4 in one operation by typing

```
s 4 a
```

where the suffix **a** refers to "all" the registers simultaneously. Similarly, all eight registers can be returned to the working area from the fourth group of save registers simultaneously by typing

```
r 4 a
```

Calculations

Once the lattice and element definitions and the beam rigidity are entered, the beam envelope can be tracked through the lattice. The program calculates the first order matrix for each element in the beam line and propagates the initial description of the beam in terms of its betatron parameters through each element of the lattice.

To calculate the beam parameters just at the end of the beam line, type

```
g    or  go
```

To track the betatron functions through every element of the lattice, type

```
c    or  cycle
```

or to track the beam size, type

```
x    or  xysize
```

Other commands calculate the chromaticity of a synchrotron (and additional functions for electron storage rings), the sextupole strengths to set the chromaticity of synchrotrons to zero, and calculate and store the matrix of sections or of the entire beam line. These stored matrices may themselves be used as beam line elements, or arbitrary matrices may be entered as data.

Acceptance Polygon

Any element in the beam line may be given an aperture radius in the x- and the y-plane. Recall that the format of the element description is:

```
name      type    vary code    p1    p2    p3    x-aper   y-aper
```

The last two parameters are the aperture radius in meters. Any element, including a drift, may be given a finite aperture. If the value is zero or absent, the element is not considered to be a potential aperture stop.

The geometrical acceptance polygon of a beam line may be calculated with the

```
po or polygon
```

command. The aperture is considered to exist as a slit at each end of each element that has a defined aperture. The aperture at each end is transformed to the beginning of the transport line and the smallest area that is formed by the intersections of all the transformed aperture limitations forms the *acceptance polygon* for the beam line. Its area is calculated and the vertex points are printed out, along with an identification of which elements contribute to the minimum geometrical acceptance.

Optimization (Fitting)

Optimization is the process of varying the parameters of some of the lattice elements so that certain conditions or constraints are satisfied. For example, the quadrupole gradients in a synchrotron lattice may be varied to set the tune to a certain value. LATTICE includes an optimizer that will accept up to ten variables with ten constraints.

The idea is to indicate which parameters in the element list are to be made variable, and to introduce a list of constraints. Some constraints, such as synchrotron tune or transition gamma apply to the entire beam line. The rest of them, such as beam size or matrix elements, apply at the end of the beam line. A total of 38 different types of constraints may be applied: they are listed under the `fit` command in Section 5.

Once the element parameters have been indicated as being variables and the fitting constraints have been introduced, the `iterate` command causes a search routine to replace the old element parameter values with new ones so that the beam satisfies the constraints. If no solution is found that satisfies the given error criteria, the original parameters are unchanged.

There are three ways of setting the *vary code* of the variables in the element dictionary. The vary code consists of an integer and a fractional part **n.m**. The integer part is a sequence number. Each element in the dictionary is permitted to have at most one of its parameters variable. If **p** variables are being optimized, then for each element with a variable parameter (length, magnetic field, etc) the integer parameter **n** of its vary code must range over 1..p. That is, each element must have a distinct value of **n** ranging from 1 to p. This sequence number is used to link elements together when constrained variation is desired (see below).

The fractional part **m** specifies which parameter of the element is to be made variable. One of the three parameters following the vary code may be made variable. The fractional part **m**, which will be the digit 1, 2, or 3, refers to the first, second, or third parameter. The first parameter of an element is usually the length, the second usually the field. Refer to the `elements` command in Section 5 for the meaning of the parameters of the 9 element types supported by LATTICE.

For example, if a quadrupole element has the vary code `1.2` as in

```
q1  q  1.2  .5  3.4
```

then the vary code would have a sequence number of 1 and the second parameter, the gradient, 3.4 T/m in this case, would be made a variable. If another element in the beam line is independently variable, its sequence number **n** would be 2. The order of sequence numbers is unimportant: but they must range from 1 to p, the total number of variables.

It is possible to lock several elements together so their parameters are varied together as one variable. Different elements with the same sequence number are counted as one variable and the parameter being varied is shifted the same additive amount during the search procedure. If one sequence number is negative, the search subtracts, rather than adds the correction to that parameter. This is useful, for example, in moving an element in a beam line while keeping the total length of the beam line constant. A situation where a quad is to be varied and slid along a constant length beam line might look like:

```
l1  d  1.1  5
q1  q  2.2  .5  3.4
l2  d -1.1  5
```

where the lengths of the two drifts are anticoupled, and the quadrupole gradient is the other variable.

As mentioned above, there are three ways to set the vary codes: by including them in the element definitions when entering them; by using the `vary` command; or by using the `alter` command to change the vary code of the element by hand once entered. The `vary` command, described in Section 5, is a quick way of specifying variables. To link variables as shown above, the `alter` command must be used.

The constraints (fitting parameters) are entered with the `fit` command. The number of constraints must equal the number of variables. The fit list is opened and the constraints are entered, along with their numerical values. The list is then closed. For example, to constrain the tunes of a synchrotron lattice to 2.7 and 3.7, type

```
fit
nux  2.7
nuy  3.7
end
```

A total of 38 constraint types are provided. See Section 5 under `fit` for details.

Once the variables have been set and the constraints entered, the fitting process is invoked with the `iterate` command. If the process finds a solution, the variables will be changed. It is sometimes advisable to save the old solution in one of the storage registers (see `save n all` in Section 5) in case the solution is not the desired one. The solver checks for singularity or divergence in the optimization process.

Section 5 Commands in Detail

Alphabetical List of Commands. The commands grouped by function can be found in Section 8.

```
*  comment line <blank>

a(lter  element name   element parameter   new value
```

b(eam Rigidity<1 T-m> ε_x<10π cm-mrad)> ε_y<10π cm-mrad> dp/p<0%>

```
c(ycle number of subdivisions per element<1>  to element number<all>

ch(romaticity  number of subdivisions per element<8>

cs(ex    name of first sextupole     name of second sextupole

d(ata  through element number<all>

e(lements
    (list of element descriptors)
end

f(it
    (list of fitted function and parameter value)
end

g(o  dp/p<0%>
gr(aph  x-scale<5 cm>  y-scale<5 cm>  ηxscale<5 m>  ηyscale<5 m>

gt

gx  number of subdivisions per element<8>

h(elp

init

i(terate  error criterion<0.001>   relaxation parameter<1>
i(terate  a
    through
i(terate  f

l(attice
    (list of element names which form the order of the lattice)
end

l(attice  l(ist
l(attice  i(nsert  where   what
l(attice  d(elete  start deletion   finish deletion
l(attice  a(lter   where   new element
l(attice  r(everse
l(attice  e(lement

m(atrix   power<1>  through element number<all>

ma  power<1>   through element number<all>   store register

ma(trix  n  lth<m>  m11...m56 (30 matrix elements in order)

o(ffline  on
o(ffline  off
o(ffline  purge

p(eriods  number of periods<1>

po(lygon
po(lygon  g
po(lygon  dp/p<0%>
po(lygon  dp/p<0%>  g

quit

re(ad  filename
```

r(ecall number register
r(ecall number register **t(ransport**
r(ecall number register **f(it**
r(ecall number register **b(eam**
r(ecall number register **e(lements**
r(ecall number register **a(ll**
r(ecall list of save registers

s(ave number register
s(ave number register **t(ransport**
s(ave number register **f(it**
s(ave number register **b(eam**
s(ave number register **e(lements**
s(ave number register **a(ll**

sc(ale s_x<5 cm> s_y<5 cm> $s\eta_x$<5 m> $s\eta_y$<5 m>

t(ransport
t(ransport off

t(ransport β_x<m> α_x β_y<m> α_y η_x<m> η'_x η_y<m> η'_y
t(ransport x<cm> r_{12} y<cm> y_{34} **xy**
t(ransport x<cm> r_{12} y<cm> y_{34} η_x<m> η'_x **xy**
t(ransport x<cm> r_{12} y<cm> y_{34} η_x<m> η'_x η_y<m> η'_y **xy**
t(ransport s
t(ransport s-

v(ary
(list of element names and parameter names)
end

w(rite
w(rite w

x(ysize number of subdivisions per element<1> to element number<all>

`*   comment line <blank>`

A comment line can be entered which will be printed out after various commands. The comment is separated from the `*` by at least one space. The comment line is useful to distinguish among several beam lines saved in the registers.

`a(lter` `element name   element parameter   new value`

Any parameter in the element dictionary in the working area can be altered with this command. The first parameter refers to the name of the element, the second to which parameter is to be varied, and the third parameter is the new value for that parameter. The allowable values of the `element parameter` are as follows:

`type`	type element type (quad, drift, etc.)
`vc`	vary code
`lth,l`	length of an element that has a length
`angle,a`	edge angle of a magnet
`b`	field of a dipole
`b',b`	gradient of a quadrupole
`n`	field index of a magnet
`gap,g`	gap half-height of a magnet
`s`	strength of a lens
`x`	half-width aperture of any element
`y`	half-height aperture of any element

An example is

```
a  q3  b'  3.4
```

which would change the gradient of the element named **`q3`** to 3.4.

`b(eam` `Rigidity<1 T-m>` ε_x`<10π cm-mrad)>` ε_y`<10π cm-mrad>  dp/p<0%>`

The basic beam parameters are given with this command. The rigidity is used by any magnetic element acting on the beam. If the rigidity is given in Tesla-meters, the magnetic fields and gradients are in the units of Tesla and Tesla/meter, and the lengths given in meters. The emittances are used when graphing the beam size or printing out the beam size with the **`x`** or **`xysize`** command. The momentum spread is also used when graphing or printing out beam size when the dispersion is non-zero.

---

c(ycle `number of subdivisions per element<1>  to element number<all>`

This command prints out the betatron functions at the end of each element in the beam line. In the *lattice* mode, the matched functions are first calculated. in the *transport* mode, the betatron functions are provided by the **transport** command.

Each element (except magnet edges) can be more finely divided by supplying a second parameter indicating into how many divisions each element is to be divided. Rapidly changing beam sizes along the length of an element can then be observed.

The third parameter restricts the calculation to the first n elements.

---

ch(romaticity `number of subdivisions per element<8>  freq<0 MHz>  V<0 MV>`

The tune of a synchrotron lattice varies with momentum. The chromaticity in each plane is defined as

$$C_x = \frac{\delta \nu_x}{\delta\left[\frac{\Delta P}{P}\right]}, \quad C_y = \frac{\delta \nu_y}{\delta\left[\frac{\Delta P}{P}\right]}$$

The chromaticity of the lattice is calculated if the matched functions exist. The chromaticity integrals are approximated by subdividing each element into 8 pieces. The second parameter permits this subdivision to be changed if more accuracy is required at the expense of calculation time.

In addition, several functions are calculated for electron storage rings, in particular the partition function, the natural emittance and energy spread, the radiation loss per turn and the three damping times. If the second and third parameters contain the r.f. frequency in MHz and the total cavity voltage in MV, the phase oscillation frequency, energy aperture and quantum lifetime are also calculated.

---

cs(ex `name of first sextupole    name of second sextupole`

The chromaticity of a synchrotron is set to zero with this command. The synchrotron must be provided with two families of sextupole elements which will provide a way of changing the first order chromaticity in each plane. The strength of these elements in units of Tesla/m^2 will be determined which sets the first order chromaticity to zero.

As an example, let the lattice contain two sextupole families called **sx** and **sy** which are respectively near focusing and defocusing quadrupoles where the dispersion is non-zero. Then, the command

```
cs  sx  sy
```

will set the gradient of the two sextupoles so that the first order chromaticity is set to zero.

```
d(ata  through element number<all>
```

This command lists all elements in the beam line, along with a sequence number which is useful in lattice editing. If a parameter is included, that number of elements are listed.

```
e(lements
      (list of element descriptors)
end
e(elements  l(ist
```

This command prepares the program to read in a list of elements into the element dictionary. This command can be used as often as necessary to read additional elements into the dictionary. Each entry in the dictionary has a unique name, no longer than six characters. A new element with the same name as one previously existing will overwrite the old one.

Nine different types of elements are defined. They include a drift, horizontal and vertical magnets and their corresponding edge elements which are used to define non-normal entry and exit angles and gap fringe field corrections, a quadrupole, a generalized lens of strength s (inverse meters), a sextupole and a general matrix with the coefficients either read in with the **ma** command or extracted from a given beam line segment.

The list of element list descriptors is read in, one element per line. The list is terminated with the **end** command. The element and their parameters is as follows:

name	**d (drift)**	n.m	length	-	-	x	y
name	**b (bend)**	n.m	length	B	n	x	y
name	**e (edge)**	n.m	angle	B	gap/2	x	y
name	**B (vertical bend)**	n.m	length	B	n	x	y

name	**E (vertical edge)**	n.m	angle	B	gap/2	x	y
name	**q (quadrupole)**	n.m	length	B′	-	x	y
name	**l (lens)**	n.m	length	s	-	x	y
name	**s (sextupole)**	n.m	length	B″	-	x	y
name	**m (matrix)**	0	n	-	-	-	-

The vary code n.m indicates which parameter of the element, if any, is made a variable for optimization. If N parameters are being varied, then the integer part n of the vary code must range from 1 to N for each element with a variable parameter. The parameters of two or more elements may be linked together as a single variable by making n the same for them. Parameters can be linked and be made to move in equal but opposite directions by making the sign of one of the vary codes negative. The fractional part of the vary code is the digit 1 through 3, indicating which parameter, the first through the third, is to be varied. Only one parameter per element can be variable.

The aperture x or y is the distance in meters from the central orbit to an aperture restriction for that element. If a value is entered for either or both parameters, it will be used in calculating the geometric acceptance of the beam line with the **polygon** command.

The bend and vertical bend dipole elements have normal entry and exit edge angles and a zero height gap. For non-zero end angles, the magnet edges are defined with the **edge** element type and are used on either or both ends of a magnet where appropriate. The first parameter is the angle in degrees, with a rectangular magnet having positive entrance and exit angles. The field parameter must be set to the same value as the accompanying magnet. The gap half-height includes a small correction for field fall-off at the edge. The vertical edge element must be used with the vertical bend.

The lens element has a length l and a strength s. The element has no physical counterpart but can be thought of as like a solenoid that does not rotate the beam round the axis, an einzel lens or a quadrupole that is converging (or diverging for negative strength) simultaneously in both planes. The strength s is equivalent to $B_{sol}/2B\rho$ or to $(B'_{quad}/B\rho)^{1/2}$. The strength parameter has units of inverse length.

The sextupole is considered to be a drift element, as LATTICE is a first-order program. However, it contributes to the chromaticity of a synchrotron lattice. When used as a first-order drift element, only its length may be made variable. Another command, the **cs** command, is used to tune the sextupole strengths to set the chromaticity of a synchrotron to zero.

The matrix type of element uses a matrix previously stored in one of the six matrix registers. The parameter n of this element type refers to one of the six storage registers.

As an example, to enter the definitions for two drift lengths and one quadrupole, type

```
e
l1 d 0 1
l2 d 0 2.5
q1 quad 0 1 3.4
end
```

which would enter two drifts 1 and 2.5 meters long, and a 1 meter quadrupole with a gradient of 3.4 T/m.

To list the contents of the element dictionary, type

```
   e  l
```

or

```
   elements  list
```

```
f(it
      (list of fitted function and parameter value)
end

f(it  l(ist
f(it  n
```

This command is used to enter a list of constraints for the optimizer. Up to 10 constraints may be entered. Each line in the constraint list consists of the name of the parameter to be constrained, followed by its numerical value. See the end of Section 4 for a discussion of fitting.

The keywords and parameters definitions are shown in the following table.

`totlength`	length<meters>	
`betax`	β_x<meters>	
`alphax`	α_x	
`etax`	η_x<meters>	
`eta'x`	η'_x	
`nux`	ν_x	
`trgamma`	$\gamma_{transition}$	
`x`	x<cm>	(transport variable)
`x'`	x'<mrad>	(transport variable)
`r12`	r_{12}	(x-x' transport correlation)
`totangle`	θ_x<degrees>	
`m11`	m_{11}	
`m12`	m_{12}<meters>	
`m21`	m_{21}<meters^{-1}>	
`m22`	m_{22}	
`m16`	m_{16}<meters>	
`m26`	m_{26}	
`m16rev`	$m_{12}m_{26}$-$m_{16}m_{22}$<meters>	
`m26rev`	$m_{21}m_{16}$-$m_{11}m_{26}$	
`tracex`	m_{11}+m_{22}	
`betay`	β_y<meters>	
`alphay`	α_y	
`etay`	η_y<meters>	
`eta'y`	η'_y	
`nuy`	ν_y	
`y`	y<cm>	(transport variable)
`y'`	y'<mrad>	(transport variable)
`r34`	r_{34}	(y-y' transport correlation)
`sumthy`	θ_y<degrees>	
`m33`	m_{33}	
`m34`	m_{34}<meters>	
`m43`	m_{43}<meters^{-1}>	
`m44`	m_{44}	
`m36`	m_{36}<meters>	
`m46`	m_{46}	
`m36rev`	$m_{34}m_{46}$-$m_{36}m_{44}$<meters>	
`m46rev`	$m_{43}m_{36}$-$m_{33}m_{46}$	
`tracey`	m_{33}+m_{44}	

For, example, if the x-tune of a synchrotron were to be set to 2.4 and the total bend angle around the machine were to be set to 360 degrees, the following would be typed:

```
f
 nux 2.4
 totangle 360
 end
```

The contents of the current fit list can be printed with the

```
 f 1
```

command. If only the *n*th element of the fit list is to be changed, rather than re-entering the whole list, type

```
 f n
```

where n is replaced by a number which indicates the variable's position in the fit list. Then enter the constraint line.

---

g(o dp/p<0%>

This command calculates the beam parameters at the end of the beam line. In the *lattice* mode, the matched functions are displayed if they exist, or the trace of the matrix if they do not. In the *transport* mode, the initial beam is transformed by the transport line and displayed.

If the parameter is non-zero, the momentum (or rigidity) is shifted by the value of the parameter in percent. The rigidity is returned to the nominal value after the calculation. In this way, the effect of changing the rigidity of the beam can be seen on the parameters of the beam at the exit of the beam line.

---

gr(aph x-scale<5 cm> y-scale<5 cm> $\eta_x$scale<5 m> $\eta_y$scale<5 m>

This command draws a crude printer plot on the terminal of the beam size functions x and y, along with the dispersion functions η_x and η_y. The beam size in each plane is defined by

$$x = \sqrt{\beta\,\varepsilon} + \eta\frac{\Delta p}{p}$$

The emittances and momentum spread are given on the **beam** command.

The scale factors may be changed and they will retain their new value for subsequent graph calls. If a scale factor is set to zero, its associated plot is omitted.

---

gt

If your terminal emulates Tektronix 4010/4014, this command will draw a medium-resolution plot of the beam size and dispersion on the screen using the same scales as the **graph** command. The effect of dispersion will be included in the beam size, as with the **graph** command. To continue processing, hit the **<cr>**. Some terminals are left in the Tektronix non-scrolling mode and must be returned to the text mode manually.

---

gx `number of subdivisions per element<8>`

A high-resolution annotated plot can be generated off-line with the support program LATGRAPH using data generated by this command. A ''gx'' file will be generated in the user's directory containing the plot information. The command will ask for a file name to be given to this ''gx'' file. Any number of these files can be generated. See Section 6 for use of the LATGRAPH graphics support processor.

---

h(elp

Two screens of information will be printed listing all the commands that the program understands.

---

init

All registers in the program are cleared. This command does not really ever need to be used.

```
i(terate  error criterion<0.0001>   relaxation parameter<1>
i(terate  a
      through
i(terate  f
```

This command initiates the optimization process. First, fit conditions are entered with the **fit** command, and the same number of element parameters are made variable with the **vary** command. Then the iterate command initiates the search routine that changes the variables so that the constraints are satisfied. If the program cannot find a solution, the original parameter values are left unchanged. See the end of Section 4 for a discussion of fitting.

Some control over the convergence process is provided by the two parameters. The error criterion determines the maximum allowable difference between any constraint and the calculated value. If the value is increased, a looser fit is permitted.

The amount of change of the variables per step is determined by the relaxation parameter. If this parameter is reduced in size, the program will undershoot the correction, which helps convergence in difficult situations. The value of the relaxation parameter increases toward unity after each iteration step.

Several values of the fitting and relaxation parameters are built into the program and can be invoked by typing the letter ''a'' through ''f'' as the second parameter. The values of these parameters for various letters are as follows:

letter	fit parameter	relaxation parameter
(none)	0.0001	1.0
a	0.0001	0.1
b	0.001	0.1
c	0.01	0.1
d	0.01	0.01
e	1.0	0.01
f	5.0	0.01

The closer the parameter is toward ''f'', the looser the fitting criterion and smaller the convergence steps. No parameter at all requires the tightest fit and converges fastest for well-conditioned systems.

It is frequently useful to save a beam configuration in one of the registers before optimizing it.

```
l(attice
      (list of element names which form the order of the lattice)
end
```

```
l(attice  l(ist
l(attice  i(nsert  where   what
l(attice  d(elete  start deletion    finish deletion
l(attice  a(lter   where   new element
l(attice  r(everse
l(attice  e(lement
```

The **lattice** command prepares the program to receive the list of names that orders the elements in the beam line. The list of names can span several input lines, and must end in an **end**. Note that no element can have the name "end". The list and the **end** can all be on one line. Three examples of entering the same list of four elements follow:

```
lattice
   l1 q1 l2 q1
end
```

```
lattice l1 q1 l2 q1 end
```

```
l
l1 q1 l2 q2 end
```

The lattice list is entered into the *working register*. The **lattice** command is also used to edit the lattice list. After a list has been entered, along with the definitions of all the elements with the **element** command, the **data** command can be used to display the list along with the sequence number for each element. Several lattice editing commands use these sequence numbers.

```
l(attice  l(ist
```

A short display can be obtained with the **lattice list** or **l l** command.

```
l(attice  i(nsert  n  name
```

An element with name `name` can be inserted into the list just after element `n` with the **lattice insert** or **l i** command. If `n` is zero, the new element will head the lattice list.

```
l(attice  d(elete  n1  n2
```

The elements from `n1` to `n2` can be deleted with the **lattice delete** or **l d** command.

```
l(attice  a(lter  n  name
```

An element with name `name` can replace the **nth** element in the list with the **`lattice alter`** or **`l a`** command.

```
l(attice  r(everse
```

The entire lattice list is reversed with the **`lattice reverse`** or **`l r`** command.

```
l(attice  e(lements
```

Finally, a lattice can be automatically generated directly from the element dictionary, in the same order as entry into the element dictionary, by the **`lattice elements`** or **`l e`** command.

```
m(atrix  power<1>  through element number<all>

m(atrix  power<1>  through element number<all>   store register
```

The **`matrix`** command calculates the 6 × 6 matrix for the entire lattice and displays it. If the **`power`** parameter is greater than 1, the matrix to that power is displayed. The next parameter limits calculation through that number of elements.

The matrix calculated may be stored in one of 6 matrix storage registers by including a third parameter. The storage register also will include the accumulated length of beam line and bend angles in both planes. This matrix may then be included in the calculation by using an element of matrix type as described under the **`element`** command.

```
ma  n  lth<m>  angle<deg>  m11 m12...m55 m56 (30 matrix elements in order)
```

This command allows an arbitrary matrix element to be entered. The matrix is saved in one of the six matrix storage registers indicated by `n` which ranges over 1..6. The next two parameters are the length and bend angle, if any in the x-plane. The next 30 parameters are the first five rows of the 6 × 6 matrix. (The sixth row is all zero except for the last element, which is always one.) The input can span more than one input line.

Some operations, such as graphing and chromaticity calculations cannot be done with matrix type elements in the beam line: they must use actual element definitions.

```
o(ffline  on
o(ffline  off
o(ffline  purge
```

The **offline** command controls output to an off-line file which saves all interaction with the program for later inspection. All user input and almost all program generated screen images are duplicated in the off-line file.

The **offline on** or **o on** command invokes a request for an off-line file name. The program appends the suffix ".out" to the file name. The file will appear in the user's directory. Writing to that file can be interrupted with the **offline off** command. The next invocation of the **offline on** command will not ask for a file name again. The **offline purge** closes the offline file and the next **offline on** command will request a new file name.

```
p(eriods  number of periods<1>
```

This command indicates how many periods of the sequence in the lattice list comprises the entire synchrotron. In particular, the synchrotron tunes, the length and total bend angles in both planes are affected. This is in effect in both lattice and transport mode.

The command without an argument prints out the current number of periods in the synchrotron or beam line.

```
po(lygon
po(lygon  g
po(lygon  dp/p<0%>
po(lygon  dp/p<0%>  g
```

This command initiates the calculation of the acceptance polygon in both the x- and y-phase planes. Each element that has a non-zero aperture is included, with the aperture applied at both ends of the element (but not in the interior).

The effect of momentum deviation can be calculated by including a percentage momentum shift as the second command. In systems with dispersion, the acceptance polygon will become nonsymmetric and usually move off the origin.

If the literal **g** is included as the first parameter (no momentum shift) or as the second parameter, an external file with the suffix ".po" will be generated which contains information to graph out the acceptance phase space in both planes. This file is used by the program POLYGRAPH to produce graphics plots and listings of the acceptance polygon. See Section 6 for more information.

`quit`

The program ends execution.

```
re(ad  filename
```

The program reads a data file and executes it. This file is usually generated by the program itself with the **`write`** command, but may also be prepared off-line with any text editor and may include any legal LATTICE command except another **`read`** command.

If the filename is not given as a parameter, the user will be prompted for it.

```
r(ecall  n1  n2  n3  n4 . . .
r(ecall  n  t(ransport
r(ecall  n  f(it
r(ecall  n  b(eam
r(ecall  n  e(lements
r(ecall  n  a(ll
```

LATTICE has six storage registers that duplicate every register in the working area (see Section 4). Data can be exchanged between the working area and these registers selectively or as a group.

The command without a literal **`t,f,b,e,a`** as the second argument is used to move a lattice list from one of the registers into the working area. If multiple arguments are used, a lattice is built up by concatenating the lists in the various registers called. For example,

```
  r 1 2 3 2 1
```

would build a lattice in the working register by concatenating the lattice lists in registers 1,2,3,2,1 in order. The total number of lattice elements must be no more than 250.

The other file commands load the appropriate working register from the storage register `n`. In particular, the **`recall`** `n` **`all`** or **`r`** `n` **`a`** command, where `n` refers to one of the six storage registers, loads all the various working registers simultaneously. Using this command in conjunction with the **`save`** `n` **`all`** command permits saving up to six completely independent problems simultaneously and calling any of them back.

```
s(ave   number register
s(ave   number register   t(ransport
s(ave   number register   f(it
s(ave   number register   b(eam
s(ave   number register   e(lements
s(ave   number register   a(ll
```

These commands are the counterparts to the recall commands described above. The command with only one argument saves the current lattice list in one of the six storage registers. See Section 4 for a discussion of the registers.

---

sc(ale s_x<5 cm> s_y<5 cm> $s\eta_x$<5 m> $s\eta_y$<5 m>

This command permits the graph scales to be entered with actually creating a graph. Its only use is to restore the graph scales by reading back in a previously created off-line file produced by the **write** command.

---

t(ransport
t(ransport **off**

t(ransport β_x<m> α_x β_y<m> α_y η_x<m> η'_x η_y<m> η'_y
t(ransport x<cm> r_{12} y<cm> y_{34} **xy**
t(ransport x<cm> r_{12} y<cm> y_{34} η_x<m> η'_x **xy**
t(ransport x<cm> r_{12} y<cm> y_{34} η_x<m> η'_x η_y<m> η'_y **xy**
t(ransport **s**
t(ransport **s-**

This command is used to switch between the *lattice* and *transport* modes, to display the current contents of the transport register, and to enter contents to the transport register. The lattice mode causes the matched functions of a synchrotron to be calculated, and the transport mode causes a user-specified initial beam to be propagated along a transport line.

The command without any parameters puts the program in the transport mode and lists the contents of the transport register if they have been previously entered, or sets them to default values of 1 meter, no tilt, and no dispersion if they have not been.

The command `transport off` returns the program to the lattice mode.

The command with parameters sets the transport mode and enters the initial beam betatron parameters. If the literal `xy` is present, transport-type variables (x, r_{12}, etc.) may be entered instead.

The transport variables may be set automatically with the `transport save` or `transport save-` commands. If the functions at the end of the beam are first calculated with a `go` command, these may be captured and placed in the transport register with the `t s` command. Another beam line may then be recalled from a storage register for which these initial transport variables are appropriate. Beam lines can be segmented in this way.

The `t s-` command does the same thing, but reverses the slope of the beam and dispersion vectors which is the appropriate starting beam for the present beam line with the lattice reversed. This command is frequently used in conjunction with the `lattice reverse` command when optimizing a beam line from both ends toward the middle.

```
v(ary
      (list of element names and parameter names)
end
```

This command provides a quick way of automatically setting the vary codes of the elements when optimization is required. Instead of using the `alter` command to set each vary code individually, the `vary` command is followed by a list of element names and the parameter to be set variable. The string is terminated by an `end`.

The parameters to this command are pairs consisting of an element name followed by a parameter type. The allowable parameter types are:

length	`l`
angle	`a`
field	`b`
gradient	`b`
lens strength	`s`
field index	`n`
magnet gap	`g`
x-aperture	`x`
y-aperture	`y`

and the list is terminated by `end`.

For example, to set the length of element `l1` and the gradient of element `q1` variable, type (with the number of spaces arbitrary)

```
v   11 1   q1 b   end
```

```
w(rite
w(rite  w
```

The **write** command dumps the program registers to an external file. This file may then be read back in with the **read** command to continue processing. The program will prompt for a file name.

All registers including all the save registers are written unless the **w w** command is given which will write only the registers in the working area.

```
x(ysize number of subdivisions per element<1>  to element number<all>
```

This command is similar to the **cycle** command, except that the beam size, both with and without dispersion, is given for every element instead of the betatron functions. The beam size with dispersion depends on the momentum spread given in the **beam** command.

Section 6 Graphics Processing

LATTICE supports three graphics driver programs: LATGRAPH, POLYGRAPH and LATDRAW. The program LATGRAPH produces high-resolution annotated plots of the beam size and betatron functions, with examples shown in Figures 1 and 2 in Section 2. The program POLYGRAPH produces high-resolution plots of the acceptance polygon in phase space for the beam line. The program LATDRAW produces a scale drawing of the actual beam line as shown in Figure 3.

To produce a high-resolution beam line plot, a ''gx'' file must be produced by LATTICE. It is useful to first produce a low-resolution plot on the screen with the `graph` command to check and adjust the scale factors. When the plot is satisfactory, use the `gx` command to produce an output file which contains the information to be used by LATGRAPH. A file name will be requested for this output file, and the suffix ''.gx'' will be appended. Any number of these files may be generated.

The plot images are produced after LATTICE execution has been terminated. The LATGRAPH program is started (see Section 7) and a help page is presented on request. The program will read any number of ''gx'' files and convert them to Tektronix plot images. For each ''gx'' file, the user can select beam size or betatron plots, and change the scales or momentum spread. The plot is generated with the `continue` command. Several copies of the same ''gx'' file can be generated, and several ''gx'' files can be read in.

When the program is exited, the file LATGRAPH.TEK is placed in the user's directory. This file contains a series of high-resolution Tektronix images to be sent to the printer. They may be also sent to a Tektronix compatible terminal for observation.

The file may be send to the Imagen printer with the

```
ipr -pip2 -LTektronix latgraph.tek
```

command. To send the file to the Talaris printer, type

```
@sys_utilities:talaris_tektronix latgraph.tek
```

The program POLYGRAPH plots the acceptance polygon derived from the aperture information provided for the elements. While running LATTICE the command `po g` prompts for a file name and produces a ".po" file with that name. POLYGRAPH reads these files and produces the output files POLYGRAPH.TEK and POLYGRAPH.OUT. The first file contains Tektronix plot images of the acceptance polygons and the second a listing similar to that produced by the `po` command in LATTICE. On termination, the graphics file can be sent to the printer by typing either

```
ipr -pip2 -LTektronix polygraph.tek
```

or

```
@sys_utilities:talaris_tektronix polygraph.tek
```

The program LATDRAW prints a scale drawing of the lattice layout. This program uses the usual lattice description file that is generated with the `write` command, which contains enough information to reconstruct the geometrical configuration of the synchrotron or beam transport line.

LATDRAW provides a help page on user request and prompts for the lattice description file. The user is led through a series of questions of the number of synchrotron periods to plot, the scale factor, and whether to produce a detailed listing of the coordinates at the ends of each element.

On termination, the files LATDRAW.TEK and LATDRAW.OUT will be present in the user's directory. The first file will contain Tektronix compatable images to be disposed to the printer and the second file contains dimensioning information for the plots. The printer commands for the graphics file from LATDRAW are

```
ipr -pip2 -LTektronix latdraw.tek
```

and

```
@sys_utilities:talaris_tektronix latdraw.tek
```

Section 7 Running in the VMS Environment

At LBL, the programs LATTICE, LATGRAPH, POLYGRAPH and LATDRAW are present in the Accelerator Code Library. They may be run with by typing

```
run accel_disk:[accellib.exe.lattice]lattice

run accel_disk:[accellib.exe.lattice]latgraph

run accel_disk:[accellib.exe.lattice]polygraph

run accel_disk:[accellib.exe.lattice]latdraw
```

All input and output files will be in the user's directory.

Section 8 Summary of Commands

A listing of all the available commands is given here. The commands are divided into nine categories. Some appear in more than one. For a detailed description of each command with examples of its use, refer to Section 5.

Each command starts with a keyword and is followed by optional parameters. In most cases the keyword can be abbreviated by its first letter. Items in **`Courier Bold`** are to be entered exactly as shown. Items in `Courier Listing` are parameters which contain the data. Some parameters have defaults and are indicated by values within carets < >.

If any one parameter of a list of parameters is to be changed, previous members of the list may be skipped over. For example,

```
b 1 2 3
```

is a beam entry command which sets the rigidity to 1 T-m, and the x- and y-emittances to 2π and 3π cm-mrad. To change the second parameter, the x-emittance to 4π, the command

```
b * 4
```

is all that is necessary. Any non-numeric character may be used in place of the **`*`**.

A numerical value may be repeated by suffixing it with an **`r`**`n`. For example,

```
t 1 2r2 3r4 5
```

is equivalent to

```
t 1 2 2 3 3 3 3 5
```

The input is completely free-format. At least one space separates the parameters. All commands are in lower case. Below, the full command keyword is indicated, as well as its abbreviation. For example, the command **`b(eam`** can be entered either by **`b`** or by **`beam`**.

Data Entry Commands

***** comment line <blank>

b(eam Rigidity<1 T-m> ε_x<10π cm-mrad)> ε_y<10π cm-mrad> dp/p<0%>

e(lements
(list of element descriptors)
end

l(attice
(list of element names which form the order of the lattice)
end

p(eriods number of periods<1>

t(ransport β_x<m> α_x β_y<m> α_y η_x<m> η'_x η_y<m> η'_y
t(ransport x<cm> r_{12} y<cm> y_{34} **xy**
t(ransport x<cm> r_{12} y<cm> y_{34} η_x<m> η'_x **xy**
t(ransport x<cm> r_{12} y<cm> y_{34} η_x<m> η'_x η_y<m> η'_y **xy**
t(ransport **s**
t(ransport **s-**

sc(ale s_x<5 cm> s_y<5 cm> $s\eta_x$<5 m> $s\eta_y$<5 m>

ma n lth<m> m11...m56 (30 matrix elements in order)

Calculation Commands

g(o dp/p<0%>

c(ycle number of subdivisions per element<1> to element number<all>

x(ysize number of subdivisions per element<1> to element number<all>

ch(romaticity number of subdivisions per element<8>

cs(ex name of first sextupole name of second sextupole

po(lygon
po(lygon **g**
po(lygon dp/p<0%>
po(lygon dp/p<0%> **g**

m(atrix power<1> through element number<all>

Print/Graph Data

```
d(ata  through element number<all>

gr(aph  x-scale<5 cm>  y-scale<5 cm>  ηxscale<5 m>  ηyscale<5 m>

gx  number of subdivisions per element<8>

gt
```

Data Editing

```
a(lter  element name   element parameter   new value

l(attice  l(ist
l(attice  i(nsert  where   what
l(attice  d(elete  start deletion    finish deletion
l(attice  a(lter   where   new element
l(attice  r(everse
l(attice  e(lement

r(ecall   list of save registers
```

Optimization

```
f(it
      (list of fitted function and parameter value)
end

v(ary
      (list of element names and parameter names)
end

i(terate  error criterion<0.001>   relaxation parameter<1>
i(terate  a
      through
i(terate  f
```

Moving Data

```
s(ave   number register
s(ave   number register   t(ransport
s(ave   number register   f(it
s(ave   number register   b(eam
s(ave   number register   e(lements
s(ave   number register   a(ll

r(ecall  number register
r(ecall  number register   t(ransport
r(ecall  number register   f(it
r(ecall  number register   b(eam
r(ecall  number register   e(lements
r(ecall  number register   a(ll

m(atrix  power<1>   through element number<all>   store register
```

Mode

```
t(ransport
t(ransport  off
```

Input/Output

```
o(ffline  on
o(ffline  off
o(ffline  purge

re(ad  filename

w(rite
w(rite  w
```

Miscellaneous

```
h(elp

quit

init
```