Fermi National Accelerator Laboratory

FERMILAB-Conf-98/125

Standardization of Beam Line Representations

David C. Carey

Fermi National Accelerator Laboratory
P.O. Box 500, Batavia, Illinois 60510

June 1998

Published Proceedings of the *5th International Conference on Charged Particle Optics*,
Delft, The Netherlands, April 14-17, 1998

Disclaimer

Distribution

Fermilab-Conf-98/125

Standardization of Beam Line Representations

David C. Carey
Fermi National Accelerator Laboratory*
Batavia, Illinois 60510 U.S.A.

Abstract

Standardization of beam line representations means that a single set of data can be used in many situations to represent a beam line. This set of data should be the same no matter what the program to be run or the calculation to be made.

We have concerned ourselves with three types of standardization: (1) The same set of data should be usable by different programs. (2) The inclusion of other items in the data, such as calculations to be done, units to be used, or preliminary specifications, should be in a notation similar to the lattice specification. (3) A single set of data should be used to represent a given beam line, no matter what is being modified or calculated. The specifics of what is to be modified or calculated can be edited into the data as part of the calculation.

These three requirements all have aspects not previously discussed in a public forum. Implementations into TRANSPORT will be discussed.

Introduction

The standardization of beam line representations is a significant aid in the making of beam line calculations. It can also take many forms.

The concept of standardization can apply to the input data format for different computer programs. If the data sets for two different programs are identical, then any difference in the output of the two programs is from the programs themselves, and not an artifact of the translation of data from one form to another.

Standardization of data format within a given program can also be useful. A single, versatile, and comprehensive data format can make the use of a given program both easier and more reliable. We illustrate this fact with specific examples.

Finally, standardization of the data set representing a single beam line leads to greater efficiency and fewer errors. All studies on a beam line can be done from a single disk file. Variations and modifications can be edited in as part of the running of the program Details will be given below.

Standardization Between Programs

At a workshop held at SLAC in 1984, the authors of a number of different computer programs agreed to adopt a standardized input format. The input format adopted was that used by the program MAD, authored by F. C. Iselin. Iselin had made a thorough study of the various accelerator design computer programs, and from them, and from his own ideas, he distilled an input format which was versatile, concise, and easy to interpret.

The ease of interpretation may be illustrated by comparison to the original notation of the computer program TRANSPORT. In MAD, a sector bending magnet which is 3.0 meters long and bends through an angle of 10 milliradians is represented as:

```
SBEND,  L = 3.0,  ANGLE = 0.010  ;
```

There is no question as to the nature of the element, how it is parameterized, and what the values of the parameters are. By contrast, in the original numerical notation of TRANSPORT, the same element would be represented as:

```
29.   3.0  22.238  ;
```

Here the initial number (29.0) indicates that this is an SBEND element. The meaning of the parameters which follow depends on the type of element (BEND, QUAD, SEXT, etc.) and their relative position.

The second number (3.0) is the length in meters. The third number has no initial obvious relationship to the value of the bend angle in MAD notation. The number 22.238 in the original TRANSPORT notation is the value in kiloGauss of the magnetic field on the reference trajectory. To evaluate the magnetic field, we have assumed a reference momentum of 200 GeV/c.

By making TRANSPORT able to read MAD notation, it was possible to allow a number of different parameterizations of the various physical elements. The same element could be given as

```
SBEND,  L = 3.0,  ANGLE = 0.010  ;
```

or as

```
SBEND,  L = 3.0,  B = 22.238  ;
```

However, many users had old data sets either in a file on the disk of a computer, or even in the form of an old deck of cards in the drawer of a card cabinet stored in some closet. They were used to the old notation and could work with it without consulting a manual. It was therefore deemed desirable to maintain compatability with the original numerical notation of TRANSPORT.

In the course of incorporation of the MAD reading routines into TRANSPORT, it became clear that the requirement of compatability with the original numerical notation did have its consequences. For example, in the numerical notation, spaces are significant, and are used to separate the numbers. To the MAD reader, spaces were not significant. The difference between the two programs is exemplified in the fact that both programs (MAD and TRANSPORT) can read the word

```
DRIFT
```

However, MAD can also understand the element name

```
DR  IFT,
```

while TRANSPORT cannot.

Another seeming consequence of the need to maintain compatability resulted in the requirement that, in TRANSPORT, nothing may be used before it is defined. No subline may be incorporated into a beam line until the subline is defined. Similarly, no parameter may be used to define an element until that parameter is defined.

This second requirement includes element parameters. The MAD notation allows a parameter of one physical element to be defined in terms of that of another physical element. In TRANSPORT then, the element containing the specification of the value of a parameter would need to precede the element which refers to that value for the specification of one of its own parameters.

I took a small poll to see if this restriction was unreasonable, and got no objections to it. However, the smallness of the population sampled might have been unreasonably small. It was recently explained to me why it might be reasonable to define a beam line, and then define the elements it contains, and, finally, give values to some of the parameters.

The program MAD allows the specification of parameters after their use and the definition of sublines also after their use. However, there was a definite reason that this facility could not have simply been left in the MAD data reading package when it was incorporated into TRANSPORT.

The program MAD allocates a literal mnemonic to each possible parameter of each physical element That means that even if a parameter is not specified, but instead is left to have its default value, it is still assigned a storage space for its value and a literal mnemonic. This situation is possible because accelerator designers have good enough sense to use a single parameterization for each of the various types of magnet (BEND, QUAD, SEXT, etc.), and a single set of units in which to express the values of the parameters.

The program TRANSPORT is used in many contexts and the various magnetic elements have a variety of ways in which they can be parameterized. For example, an RBEND can be specified in terms of its length and bend angle, its length and magnetic field, its radius

of curvature and bend angle, its length and radius of curvature, or its magnetic field and bend angle. Only the parameters used are stored. An indexing system keeps track of which parameters are used.

This procedure is more efficient than would be the allocation of storage for each possible parameter. However, it makes it more difficult to keep track of all the references to beam sublines or parameters of other elements.

Nevertheless, the need was considered sufficiently compelling that the reading procedures in TRANSPORT have been enhanced to allow the reference to sublines or parameters not yet defined.

TRANSPORT still keeps only those parameters which are actually used. However, when it comes to a parameter which has not yet defined, it saves a pair of new elements which together can be defined as a "parameter as yet undefined." The pair of elements is necessary as one element is used to store the element reference and the other is used to store the parameter reference. After the entire beam line has been read in, all dummy references to "parameters yet undefined" are found and replaced by the actual parameter referred to. The reference was meanwhile kept in the dummy. A similar procedure is used for "sublines yet undefined," but the procedure requires only one new element.

An example is shown below. In this example, we have sublines defined both before and after the lines in which they are to be incorporated. Similarly, we have parameters defined both before and after the elements in which they are used. Some of these parameters are defined in terms of algebraic expressions involving other parameters. The algebraic expressions are still evaluated correctly, as are their partial derivatives when the program is fitting. When fitting, the program must go through the beam line several times, evaluating only those algebraic expressions whose components have already been evaluated. The program loops until all such algebraic expressions have been evaluated, then it runs through the beam line itself.

```
'SAMPLE DATA SUPPLIED BY AL RUSSELL'
  0
!  Example of poorly ordered file for Dave Carey.
!  Unit for magnetic fields is TESLA.
CELL: LINE = (QF2, QEND, D1FT, DRX, EPB, DRX, D1FT, QEND, QD2, &
              QD2, QEND, D1FT, DRX, EPB, DRX, D1FT, QEND, QF2  )
QF2:     QUADRUPOLE, TYPE = PM, L = QL/2, K1 = KF
QD2:     QUADRUPOLE, TYPE = PM, L = QL/2, K1 = KD
QEND:    DRIFT, L = LEND
EPB_FLD: RBEND, TYPE = EPB, L = LEFF, ANGLE = ANGEPB
EPB_END: DRIFT, L = LEND
EPB:     LINE = (EPB_END, EPB_FLD, EPB_END)
D1FT:    DRIFT, L = 0.3048
DRX:     DRIFT, L = 6.3912
!  Permanent magnet quadrupoles.  Nominal parameters.
```

```
QL    := (1 + 8.0/12)*0.3048
GRADF := 2.91469875
GRADD := 2.91469875
KF    := SQRT(GRADF/BRHO)*QL
KD    := -SQRT(GRADD/BRHO)*QL
LEFF   := 3.048
LEPB   := 3.198
LEND   := (LEPB - LEFF)/2
BEPB   := 1.
ANGEPB := BEPB*LEFF/BRHO
! Define 8 GeV kinetic energy beam
TBEAM   := 8.
PBEAM   := SQRT(TBEAM*(TBEAM + 2.*PMASS))
BRHO    := PBEAM*1.E9/CLIGHT
! ----------------------END LATTICE DESCRIPTION -------------------------
!  Procedure portion follows
USE, CALL
PRINT, BEAM, TRANS
SENTINEL
```

In the preceding example, elements depend on parameters whose definition may precede or follow the element itself. However, the elements can be reordered so that nothing depends on an element or parameter not yet defined.

In the following example this property no longer pertains. There is no reordering of elements so that all parameters are defined before they are used. The example involves a quadrupole triplet used to give a point-to-point focus in both planes.

```
'  USE TRIPLET TO FOCUS BEAM FROM SOURCE ONTO TARGET'
  0
UNIT,  X,  IN ;
UNIT,  L,  FT ;
BEAM:  BEAM,  X=0.5, XP=17.5,  Y=0.5,  YP=17.5,  DEL=3.0,  P0=1.05  ;
D1:   DRIFT,  L=12.0  ;
Q1:   QUAD, L=1.5, B=Q3[B], APER=Q2[APER]  ;
D2:   DRIFT,  L=0.5  ;
Q2:   QUAD,  L=2.0*Q1[L],  B = BQF,  APER=4.0  ;
D3:   DRIFT,  L=0.5  ;
Q3:   QUAD,  L=Q1[L],  B=-1.0,  APER=Q2[APER]  ;
D4:   DRIFT,  L=7.0  ;
BQF:  = 1.0*BQFS  ;
BQFS: = 1.0*BQFT  ;
BQFT: = 1.0  ;
FIT1:   FIT,  R12 = 0.0,  TOLER=.001  ;
```

```
FIT2:   FIT,  R34 = 0.0,  TOLER=.001  ;
PRB:  PRINT,  BEAM  ;
PRT:  PRINT,  TRANS  ;
VARY  BQFT  ;
VARY  Q3[B] ;
SENTINEL
```

Here the length of the first quadrupole is given explicitly, and the length of the two other quadrupoles are defined in terms of the length of the first. The field of the third quadrupole is given explicitly, while the field of the first quadrupole is defined in terms of the field of the third. The field of the second quadrupole is defined in terms of a parameter defined later. There is a sequence of parameters involving algebraic expressions before one gets to a numerical value. Finally, the apertures of the first and third quadrupoles are defined in terms of that of the second.

If this description is complicated and confusing, that is quite deliberate. It is, nevertheless, well defined, and TRANSPORT can now read this data set and perform the fit.

Standardization of Data for a Single Program

At the workshop in 1984 it was decided that the standardization of beam line representation should apply only to the physical lattice. The MAD or MAD-like notation would be used for all the physical elements. The operations were regarded as being specific to a particular program and therefore not subject to standardization.

In the year or so after the workshop, the MAD notation for physical elements was incorporated into TRANSPORT. Some non-MAD elements, such as the BEND (a bending magnet without fringing fields) were also included in TRANSPORT in MAD-like notation. In addition, by simply expanding the list of element parameter keywords, it was possible to allow a variety of parameterizations of various elements.

The three categories of element which were not part of the physical description of the lattice were the beam phase space description, preliminary specifications, and operations. The list of preliminary specifications included units specifications, random errors on physical parameters, and limits on varied quantities in fitting. The operations included printing, plotting, fitting, and misalignments of magnets or sections of a beam line. Some of these non-lattice elements could easily be put in a MAD-like notation, while others did not lend themselves so well to such an implementation.

After the modifications were completed, TRANSPORT could read the description of the physical lattice in MAD or MAD-like notation. This description would be part of a data set which would also include the preliminary specifications and the operations. A lattice description could be taken from a set of MAD data and incorporated into a TRANSPORT deck and run.

At this point, the authors of TRANSPORT decided that it would be worthwhile to revise the documentation of the program. Since some of the elements could still be expressed only in the original numerical notation, it was necessary to describe this notation. In addition, there were many users who had existing data sets in the old notation. The new manual then described for each element both the original numerical notation and the new MAD or MAD-like notation.

All the required information was in this new manual. However, the description of both notations for each element proved to be very confusing for users. In preparing data, users unused to the new notation would incorporate aspects of one notation in an element otherwise expressed in the other notation.

The obvious solution then was to standardize on the MAD or MAD-like notation. The use of the keyword notation could not be limited to the lattice specification. For the program to be straightforward to use, all elements must be expressable in a similar notation. Thus we arrive at the need for a second type of standardization of beam line representation. The input for a single program should be stylistically consistent.

Keyword notations for the remaining elements were then worked out. For some of the elements, this procedure allowed a significant enhancement of the capabilities of TRANSPORT. Here we will describe the development of new capabilities for a few of the elements.

A constraint used in fitting lends itself well to at least the pattern of a MAD-like representation. For example, a constraint to the value zero on the R_{12} matrix element can be expressed as

```
FIT,  R12 = 0.0,  TOLER = 0.001  ;
```

The tolerance is used in calculating the chi squared. The fitting process tries to find the minimum of the chi squared by locating the point in a several dimensional space where its derivatives are zero.

The old numerical notation is still used as the representation of the constraint internally to TRANSPORT. The difference between the FIT specification and that of a typical physical element is that what appear to be the keywords are not a fixed set of literals stored in a dimensioned array. With the second-order matrix elements T_{ijk} and the third-order matrix elements $U_{ijk\ell}$, there are too many possible "keywords" representing the quantity to be constrained. Instead TRANSPORT recognizes the various matrix elements as being a letter followed by one or more integers, and then converts to an internal notation. For some of the quantities, like the floor coordinates, a simple name, with no numbers, specifies the quantity being constrained.

The beam phase space can be expressed in terms of a six-dimensional sigma matrix. If the two transverse planes are separated, it can also be expressed in terms of alpha and beta parameters and the emittances in both planes. With the old numerical notation, there was no natural way to distinguish between the two possibilities. With the MAD-like notation

either type can be specified, simply by expressing the BEAM element in terms of the appropriate keywords. If the parameters on the element are to indicate the dimensions of the six-dimensional phase ellipse, then they are `X, XP, Y, YP, DL, and DEL`. If the beam phase space is to be expressen in terms of accelerator parameters, then the keywords to be used are `BETAX, ALPHAX, EPSX, BETAY, ALPHAY, and EPSY`. The beam phase space dimensions are stored internally in both cases in sigma matrix notation. The format for input of the beam phase space dimensions does not dictate the form of the output. The two can be treated independently. If the output of the beam phase space is to be in sigma matrix form the command

```
PRINT,  BEAM  ;
```

is used. If the output of the beam phase space is to be in accelerator parameters, then the command

```
PRINT,  TWISS  ;
```

is used.

A third example of a MAD-like format for an element which is not quite exactly in the strict MAD style lies in the units specifications. Since accelerator designers have enough good sense to work in consistent units, there is no need for units conversions in MAD. However, TRANSPORT was designed for a situation where longitudinal units might be in feet or meters, transverse units might be in cm or inches, and transverse angles might be in milliradians. A milliradian is in itself a self-inconsistent unit, since it is (to second order) the ratio of two lengths expressed in different units.

The UNIT element cannot be expressed in terms of a fixed set of keywords because the unit name must be part of the input. This requirement occurs since TRANSPORT prints out the physical parameters describing the element and the results of the calculations, including the units. Since new units, such as Webers per square foot, can be introduced, it is necessary to have the list of possible "keywords" be open ended and allow new possibilities.

With the old numerical input, the only place where a name could be specified is in the element label. With the new keyword notation, it is possible to express the unit name as one of the keywords. One nice feature of the keyword notation in MAD is that the keywords do not need to be assigned values. The simple occurance of a keyword, followed by a comma or semicolon, may indicate something to the program. The `UNIT` element is an example of this feature.

A conversion of the transverse unit to inches might then look like:

```
UNIT,  X,  IN  ;
```

TRANSPORT recognizes the names of many commonly used units and will supply conversion factors. If the keyword `IN` were not recognized by TRANSPORT, then it would be necessary to supply a conversion factor, and the element would look like:

```
UNIT,  X,  IN,  SIZE = 0.0254  ;
```

Here the default transverse unit is taken to be meters. The default set of units is specified on a separate element.

The final example of the adaption of the keyword notation occurs in the use of the `ALIGN` element. The `ALIGN` element is used to specify misalignments of magnets or portions of a beam line. There are many options in the imposition of the misalignment element.

The results of the misalignment can be exhibited in the beam (sigma) matrix, or in a separate misalignment table. If the results are displayed in the beam matrix, then the effect of the misalignment on the transfer (R) matrix is shown also. If the results are displayed in the misalignment table, then the results of the misalignment of ten elements in each of six degrees of freedom can be shown separately.

The misalignment can be uncertain, known, or random. If the misalignment is uncertain, then the parameters on the misalignment represent the rms values of an envelope of possible deviations from the aligned position. The results of the independent misalignment of different magnets are added in quadrature.

If the misalignment is known, then the parameters indicate the exact deviation in each of six degrees of freedom (three displacement and three rotation) of the misaligned magnet or beam line section. A random misalignment is like a known misalignment, except that each of the six deviations is multiplied by a random number which can range from -1.0 to 1.0.

Finally, the entity to be misaligned can be specified in several different ways. A single magnet, a section of a beam line, or all magnets of a given type (quadrupoles, bends, etc.) can be misaligned.

In the original numerical input for TRANSPORT, all of these characteristics were specified in a single (and diffucult to interpret) numerical code. The use of the keyword notation allows the (much clearer) specification of the characteristics of a misalignment. Keywords for the display of the results are `BEAM` or `TABLE`. Those for the type of misalignment are `UNCERTAIN`, `KNOWN`, or `RANDOM`. There are additional keywords to specify the magnet or section of the beam line to be misaligned. One of the possibilities is to use the label of the magnet to be misaligned as a keyword.

Having adopted the keyword notation opens up another range of possibilities without having to use another obscure numerical code. The origin of the coordinate system in which the misalignment is given, and the orientation of the coordinate axes can also be specified. The origin can be at the entrance face, the midpoint, or the exit face of the misaligned magnet. Similarly, the orientation of axes can be the same as those for the beam coordinates at the location of the origin. If the misaligned magnet is a bending magnet, then the z axis for the misalignment coordinates can be along the chord of the magnet.

The use of the keyword notation then not only renders consistent the style of the input

for the program TRANSPORT, but opens up a number of new possibilities also.

Standardization of Representations for a Single Beam Line

Perhaps the least obvious need for standardization occurs with the data for a particular program used to represent a single beam line. Yet, the fact that the need for standardization is not obvious may mean that the actual need is significant.

The problem arises when several people are working on a beam line, or when one user is studying several aspects of a given beam line. A user may start with a single set of data to represent the beam line. He may make some runs to try out variations on the original design, or he may make some studies of the effect of errors. To do these studies he may create copies of the original data set and then make appropriate modifications. Other involved parties may do the same.

The result is that there gets to be a number of data sets all representing the same beam line. Eventually, it may not be clear which studies were done on which version of the beam line, and which data set should be regarded as the one either representing the final design or corresponding to the beam line as actually built.

The cause of the problem is that editing and calculating are considered to be two separate operations. Once a data set representing a beam line has been modified, a new data set has to be created in order to make a run through the beam line.

The program MAD (and now TRANSPORT) has a marker element. The marker element refers to a given location in the beam line. All of the operations (printing, ploting, misalignment of magnets, constraints, etc.) can be placed together at the end of the data set. The lattice can be then separated from the set of operations so that there is no overlap.

The use of the marker element solves part of the problem. However, to anticipate all possibilities, it would often be necessary to place markers in the beam line with such high density that the structure of the beam line itself would become obscured. The marker option also does not allow revisions to be made to the beam line itself on a temporary basis.

What is needed is to establish a master copy of the data set representing the beam line. From there, one needs to be able to make temporary modifications and run as part of the same procedure.

TRANSPORT has always had the ability to make such temporary modifications by means of the labels on the elements. However, any such temporary modification had to be placed in the original data set and marked as inactive with a minus sign before the mnemonic or numerical type code representing the element. Clearly not all possible studies could be anticipated in advance. A data which contained such a proliferation of deactivated elements would also be such a clutter that it would be just about impossible to work with.

With TRANSPORT, the master copy can first be created as a data set containing literals and numerical values. The procedure is described in detail in the TRANSPORT manual.

TRANSPORT then has a command by which a binary file can be created containing the needed common blocks which contain the data.

The second line in the TRANSPORT data set is called the indicator line, which indicates to the program whether the following data is a new data set, or modifications (by means of the element labels) of a preceding data set. Following the indicator number, there may be certain commands indicating various forms of output or input. One of these commands is BWRITE, which is the command which writes out the common blocks to a binary disk file. This disk file can then serve as the master copy for further studies. The advantage here is that this file cannot easily be modified directly, but must be created anew by passing the original data through the reading routines of TRANSPORT.

This file can then be read by the placement of the word BREAD on the indicator line. If this command is used, then there need be no more data giving the new data set. TRANSPORT can be run with the only element data occurring in the binary file.

Because the computer programs TURTLE and TRANSPORT have the same reading routines, the data for TURTLE can also be read from the same binary file. The data set for the program TURTLE has a similar indicator line where a second integer is to be found, indicating the number of trajectories to be run through the optical system. The command BREAD can be placed on this line, and the TRANSPORT data set in binary will be read and processed by TURTLE. At this point is the description, it would be necessary to have any histogram specifications already exist in the binary file, and therefore in the original TRANSPORT deck.

After having read the binary data file, TRANSPORT is capable of editing the data. An editor typically had two commands. The first is to remove data which is unwanted. The second is to insert new data. Often there is also a combination of the two, which is a command to replace data. The "replace" command is most often used to replace a line of data with a similar line.

The replace command, as cited above, has always existed in TRANSPORT through the identification of elements by their label. The remove and insert commands are new and previously not described.

The "insert" command is split into commands, `BEFORE` and `AFTER`. The meaning of the two commands is as expected. The `BEFORE` command is used to insert new data before the element with the label indicated on the `BEFORE` command. The `AFTER` command is used to insert new data after the element with the label indicated.

For example, a command, with label, `PR1` which is used to print the beam matrix, might appear as

```
PR1:   PRINT,  TRANS  ;
```

To insert two fitting constraints for the beam matrix after the print command, one would include in the editing data the lines:

```
PR1:   AFTER  ;
FIT,  S12 = 0.0,  TOLER = 0.001  ;
FIT,  S34 = 0.0,  TOLER = 0.001  ;
ENDINSERT
```

There is good reason why it is useful to have both `BEFORE` and `AFTER` commands. In the MAD mode, physical elements are defined first, then incorporated into a beam line by means of the `LINE` command. In making up a beam line or subline an element is referred to by its label.

TRANSPORT also has another mode, which is its original method of operating. Here the beam line elements are simply listed in sequence. TRANSPORT runs through the elements as listed. It is then not necessary for each element to have a label. However, it is likely that any point in the beam line would be either at the upstream or downstream of a magnetic element. The magnetic element would be likely to have a label. Consequently, any point in the beam line could be indicated by a `BEFORE` command or an `AFTER` command referring to a label.

The `ENDINSERT` command is necessary only when two or more new lines are being inserted. Otherwise, the new line can be included in the data as a continuation of the `BEFORE` or `AFTER` command. If only one `FIT` command is to be inserted in the situation described above, the following command is adequate.

```
PR1:   AFTER,  FIT,  S12 = 0.0, TOLER = 0.001  ;
```

No `ENDINSERT` command is necessary.

If it is desired to remove a line from the data set then the `REMOVE` command is used. An element with the label `PR1` can be removed by the command

```
PR1:   REMOVE  ;
```

The editing commands work exactly as if one were working from the original data set. The `REMOVE` command does not just deactivate the element to which it pertains. It actually removes it from the data, so that if it is needed later, it must be reinserted.

There are also commands for suppressing the printing of the data, for suppressing the printing during the run through the beam line, and for eliminating any fitting procedure. The step which reads the data from the binary file then effectively shows no output, and the calculation can begin only after the editing commands have been executed.

The master file and its binary derivative can be kept intact. The data contained in it can be temporarily modified and the data can be run through TRANSPORT. The changes can be documented simply by allowing TRANSPORT to print the editing commands.

After the editing changes have been made, another binary file can be written on the disk which can be read by TURTLE. The data can then be assimilated by TURTLE and one- and two- dimensional histograms can be created. The histogram specifications can be among the commands edited into the data, and therefore do not have to be anticipated in the master file.

Starting with the master file, we can do optimization on the beam line with TRANSPORT, and histogramming with TURTLE. A wide variety of studies and design alternatives can be explored while retaining a single standardized master file representing the beam line.

References

* Operated by Universities Research Association, Inc., under contract with the United States Department of Energy.

1. Al Russell, private communication.

2. D. C. Carey and F. C. Iselin, *A Standard Input Language for Particle Beam and Accelerator Computer Programs.* 1984 Summer Study on the Design and Utilization of the Superconducting Super Collider, Snowmass, Colorado.

3. H. Grote and F. C. Iselin, *The MAD Program, User's Reference Manual.* CERN/SL/90-13 (AP) (Rev. 4).

4. D. C. Carey, K. L. Brown, and F. Rothacker, *Third-Order TRANSPORT: A Computer Program for Designing Charged Particle Beam Transport Systems*, SLAC Report SLAC-R-95-462 (1995).