PS/AR Note 94-25 (Min.)

SAP-14

11 November 1994

*Minutes of the Forum of*

# Symbolic Computing for Accelerator Physics

*held on Thursday 3 November 1994*

**Present:** B. Autin (Chairman), J. Bosser, M. Bouthéon, R. Cappi, R. Corsini, E. d'Amico, F. Di Maio, G. Dôme, L. Durieu, M. Giovannozzi, S. Hancock, H. Haseroth, A. Hilaire, J.Y. Hemery, J. Jowett (Deputy), M. Martini (Secretary), D. Manglunki, C. Metzger, H. Mulder, W. Remmer, K. Schindl, J.C. Schnuriger, H. Schönauer, D.J. Simon, F. Thizy, E. Wildner.

## 1 Tutorial on Mathematica language - Atoms and Expressions: B. Autin

Notebook in directory: `g:\home\a\autin\windows\tutor1.ma`.

Since the beginning of the SAP forums many users have been succesfully confronted with *Mathematica*. Sometimes, the object orientation of *Mathematica* was missing, making certain applications hard to read and use. The next SAP Forums will therefore start with a short dedicated tutorial on basic aspects of the *Mathematica* language to encourage people to code their programs more easily and more efficiently.

The first tutorial session is dedicated to *Atoms* and *Expressions*[1].

1. **Atom**: In *Mathematica*, the Atoms are the smallest parts from which the language is built up. The atoms are the symbols, numbers, and strings.
2. **Expression**: The expressions are the entities handled by *Mathematica* (e.g. the formulas and the lists). The building blocks of expressions are the atoms. More precisely, the recursive definition of an expression is: If $e_0$, $e_1$, ... $e_n$ ($n \geq 0$) are expressions or atoms, then

$$e_0[e_1 \ldots e_n]$$

is also an expression. All expressions are constructed by using this rule repetitively.

For instance, `Sin[x^2]` is an expression because once rewritten in *Mathematica* functional notation as `Sin[Power[x, 2]]` it satisfies the above definition of an expression: $e_0$ is the atom `Sin` (symbol), $e_1$ is the expression `Power[x, 2]` built up of the atoms `Power`, `x` (symbols), and 2 (integer).

[1] R. Maeder, Programming in Mathematica, Addison-Wesley, 1991.

In most cases $e_0$ is an atom called the operator of the expression. All the atoms can be characterized by their head (*Head*). This property paves the way to pattern recognition. In addition, an operator can be given attributes (*Attributes*). The attributes act like axioms in set theory, they govern properties such as commutativity or associativity, and affect the way an expression is evaluated. Examples are given in the attached slides for the search of heads and for the functioning of the attributes.

## 2 Automatic injection into the PS: M. Martini

Notebooks in directory:
`g:\home\m\martinim\mathema\micguide.ma` and
`g:\home\m\martinim\mathema\micadops.ma`.

A method is proposed for the automatic injection tuning of particle beams. It is based on the measurement of two successive single turn trajectories and uses the MICADO[2] minimizer algorithm. The method has been coded as a *Mathematica* package and tested for the proton injection in the PS machine.
The package will be called by an application program through the *MathLink* communication protocol and will be integrated in the standard PS control system.

Direct *Mathematica* access to beam trajectory measurement system and corrector magnets being needed for the correction process, customized functions for equipment access have been written:

1. `ReadEquipment[{"Name1","Action1",PLSline},...]` for reading data from equipment,
2. `WriteEquipment[{"Name1","Action1",PLSline,Value1},...]` to send orders to equipment.

A description of the *Mathematica* MICADO functions is given in the Notebook "Micguide".
A prototype program written for the tuning of the injection in the PS can be found in the Notebook "Micadops", in which an example of a correction carried out in the PS is shown.

See the attached slides.

**The next meeting will be held on:**

**Friday 9 September at 16.00 hr in the PS Auditorium - Meyrin, Bldg 6, 2-024**

**The date will be confirmed later**.

M. Martini

## Distribution list

AT, MT, PS and SL Division Leaders and Deputies.
AT, MT, PS and SL Group Leaders and Associates.
SAP list.

[2]B. Autin, M. Arruat, F. di Maio, M. Martini, Beam steering: A test bench for generic algorithms in accelerator controls, CERN/PS 94-09 (AR).

# Atoms & Expressions

**B. Autin**

**3 Novembre 1994**

## Semantics

| *Natural language* | *Mathematica* |
|---|---|
| Paper | Program |
| Title | Opening |
| Abstract | Abstract |
| Sections | Functions |
| Conclusion | Closing |
| | |
| Sentence | Expression |
| Word | Atom |
| Verb | Operator |

## Elements of language

### Functions

f[var_]=definition

### Definition

A definition is a string of expressions.

### Expression

An expression is a structured assembly of words or atoms. A structure is made of an operator acting on arguments which are either atoms or structures.

## Characterizing an atom

### Head

```
Head[1]
Integer
Head[1.2]
Real
Head[1+I]
Complex
Head[a]
Symbol
Head[a+1]
Plus
```

### AtomQ

```
AtomQ[1+I]
True
AtomQ[1+a]
False
```

## ■ Attributes of an operator

### ■ Axioms of an operation

Commutativity: Orderless
Associativity: Flat
Neutal element: OneIdentity
Projectivity: Listable

#### □ No attributes

```
Clear[f]
f[x_,y_]=Log[x+y]
Log[x + y]
f[a,1]
Log[1 + a]
f[a,b,3]
f[a, b, 3]
```

#### □ Flat

Although the definition is limited to two variables, an arbitrary number of arguments may be given if the operator is associative and thus have the Flat attribute.

```
Clear[f]
Attributes[f]={Flat}
f[x_,y_]=Log[x+y]
{Flat}
Log[x + y]
f[a,b,3]
Log[f[a] + Log[f[3] + f[b]]]
```

#### □ Flat and Orderless

If the operation is commutative, then the order of the arguments does not matter and Mathematica orders the arguments according to its internal rules.

```
Clear[f]
Attributes[f]={Flat,Orderless}
f[x_,y_]=Log[x+y]
{Flat, Orderless}
Log[x + y]
f[a,b,3]
f[Log[f[3] + Log[f[a] + f[b]]]]
```

#### □ Flat and Orderless and OneIdentity

In all the previous examples, f remained unevaluated if it has a single argument. If the set on which f operates has a neutral element e, then Mathematica decides that f[x] is the same as f[x,e] and thus the same as x if the attribute OneIdentity is assigned to f.

```
Clear[f]
Attributes[f]={Flat,Orderless,OneIdentity}
f[x_,y_]=Log[x+y]
{Flat, Orderless, OneIdentity}
Log[x + y]
f[a,b,3]
f[Log[3 + Log[a + b]]]
```

#### □ Listable

Another way of introducing an rbitrary number of elements in a function even if the function is not associative is to group the arguments in lists. The evaluation of the function depends on the number of the arguments of definition.

*Single argument*

```
Clear[g]
Attributes[g]={Listable};
g[x_]=x+1;
g[1]
2
g[{1,2,a}]
{2, 3, 1 + a}
```

*Two or more arguments*

```
Clear[g]
Attributes[g]={Listable};
g[x_,y_]=x*y;
g[2,3]
6
g[{2,3,a},2]
{4, 6, 2 a}
g[2,{2,3,a}]
{4, 6, 2 a}
g[{a,b,c},{x,y,z}]
{a x, b y, c z}
```

The lists are "bridged" by the operator g. A basic constraint is that the lists must have the same length or that some of the arguments are single. In other terms g introduces a correlation between the lists.

### ■ Attributes of evaluation

HoldFirst, HoldAll, HoldRest

### ■ Attributes of protection

Protected, ReadProtected, Locked

### ■ Giving attributes to an operator

Attributes[symbol]={att1,att2,...}

```
Attributes[Plus]
{Flat, Listable, OneIdentity, Orderless, Protected}
Attributes[Set]
{HoldFirst, Protected}
Attributes[Rule]
{Protected}
```

## ■ Options of an operator

Some of the arguments whch are expected by an operator may be missing and thus have default values. That arguments are called "optional". Their default values are collected in a list of rules. In contrast with the attributes which must have names built in Mathematica, you are free to give options any value or name.

```
Options[Solve]
```

```
{InverseFunctions -> Automatic, MakeRules -> False, Method -> 3,

 Mode -> Generic, Sort -> True, VerifySolutions -> Automatic,

 WorkingPrecision -> Infinity}
```

```
Options[Bend]={Tilt->0,Edge->Automatic}
```

```
{Tilt -> 0, Edge -> Automatic}
```

## ■ Full form of an expression

```
FullForm[{a,b,c}]
```

```
List[a, b, c]
```

```
FullForm[a+b]
```

```
Plus[a, b]
```

```
FullForm[x^2]
```

```
Power[x, 2]
```

```
FullForm[a->b]
```

```
Rule[a, b]
```

```
FullForm[a=b]
```

```
b
```

```
SetAttributes[FullForm,HoldAll]
FullForm[a=b]
```

```
Set[a, b]
```

## ■ Levels of an expression

The arguments of an expression belong to the level defined by the hierarchical rank of the operator they depend upon.

```
u=f1[x,f2[y,f3[z]]];
Level[u,1]
```

```
{x, f2[y, f3[z]]}
```

```
Level[u,{2}]
```

```
{y, f3[z]}
```

```
Level[u,2]
```

```
{x, y, f3[z], f2[y, f3[z]]}
```

```
Level[u,{3}]
```

```
{z}
```

```
Level[u,3]
```

```
{x, y, z, f3[z], f2[y, f3[z]]}
```

```
Level[u,{-1}]
```

```
{x, y, z}
```

EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH

1 November 1994

# Automatic Injection in the PS

M. Martini

## Injection tuning method

When the tune is known and the closed orbit is unknown, correction of injection oscillations may be done using two successive single turn trajectory measurements and two corrector magnets.

In the PS, the injection septum and a transfer line dipole are used for the horizontal corrections while two transfer line dipoles are used for the vertical corrections.

Trajectory position at monitor $i$ in machine turn number $t$ can be written

$$x_{i,t} = z_i + A\sqrt{\frac{\beta_i}{\beta_{i_0}}}\cos(\mu_i + \phi + 2\pi(t-1)Q)$$

$z_i$ is the closed orbit position at monitor $i$, $\mu_i$, $\beta_i$ are the phase advance and beta-function, $Q$ is the tune, $i_0$ is an arbitrary monitor location so that $A$ is a real amplitude and $\phi$ the phase of the oscillation at that point.

The trajectory difference between two successive single turns is

$$\frac{x_{i,t} - x_{i,t+1}}{2\sin\pi Q} = A\sqrt{\frac{\beta_i}{\beta_{i_0}}}\cos(\mu_i + \phi + 2\pi(t-1)Q + \pi(Q-1/2))$$

Let $Z$ and $X_t$, $X_{t+1}$ be the closed orbit and trajectory vectors in turns $t$, $t+1$, the above expressions may be transformed into

$$||X_t - Z|| = ||X_{t+1} - Z|| = \frac{||X_t - X_{t+1}||}{2\sin\pi Q}$$

Ideal injection tuning would compute the corrector strengths to be applied so that the beam will be injected onto the closed orbit (i.e. cancelling the latter equation).

## Strategy of correction

MICADO strategy has been implemented for the injection tuning in the PS. The correction requires $k$ corrector magnets selected out of $m$ available. The corrector strengths that yield the minimum residual oscillations are

$$\Delta\varphi^{(k)} = -\left(A^{(k)T}A^{(k)}\right)^{-1} A^{(k)T}U$$

with

$$U = \left\{\frac{x_{i,t} - x_{i,t+1}}{2\sin\pi Q}\right\}_{i=1\ldots n}$$

which shows the importance of the use of the correct tune value in the method. The matrix $A^{(k)}$ is composed of $k$ columns $\{i_1 \ldots i_k\}$ of the full matrix

$$A = \left\{\sqrt{\beta_i\beta_j}\sin(\mu_i - \mu_j)\right\}_{j=1\ldots m}^{i=1\ldots n}$$

The indices $i$ and $j$ referring to the monitors and correctors respectively. In the PS, $m = 2$ and $n = 40$.

The norm of the expected residual oscillation after correction is

$$||R^{(k)}||^2 = U^T U + \Delta\varphi^{(k)t}A^{(k)T}U$$

- To achieve reliable corrections it is required to measure the trajectories either in the 1st and 2nd turn or in the 2nd and the 3rd turn after injection because of the phase shift $2\pi(t-1)Q$ may cause the correction system to diverge in the presence of even a small tune error.
- The performance of the correction system may be quoted in terms of the residual oscillation. It is related to the operational environment where monitors and correctors may be either redundant or missing. It is also related to the quality of the trajectory measurement and to the knowledge of the beta-functions, phase advances, and tune.

# MICADO Optimizer

**M. Martini**
**28 October 1994**
**Notebook: Micguide.ma**

## ■ Introduction

## ■ Reading in package

On PS Workstations: load the MICADO Optimizer package as follows:

```
<<"/mcr/mathematica/orbcor.ma"
```

## ■ File management

Files required for orbit corrections with MICADO:

MonitorFile:

Optical parameters of the monitors. The file consists of 9 lists (lists 1-8 are of size equal to the number of monitors, list 9 is of size 2):
- list 1: monitor names (strings),
- list 2: arc length [m],
- list 3: horizontal phases (divided by 2 Pi),
- list 4: horizontal beta-functions [m],
- list 5: horizontal dispersion-functions [m],
- list 6: vertical phases (divided by 2 Pi),
- list 7: vertical beta-functions [m],
- list 8: vertical dispersion-functions [m],
- list 9: horizontal and vertical tunes.

Lists 1-2 are not used by MICADO. Data in MonitorFile are obtained using any lattice program (e.g. BeamOptics package, MAD with OPTICS command, ...).

CorrectorFile:

Optical parameters of the correctors. The file structure is similar to MonitorFile.

ReferenceFile:

Reference positions [mm] at the monitor location. The file consists of 2 lists:
- list 1: pairs of {monitor index, horizontal reference position},
- list 2: pairs of {monitor index, vertical reference position}.

GeometricalFile:

Geometrical errors [mm] at the monitor location. The file consists of 2 lists:
- list 1: pairs of {monitor index, horizontal geometrical error},
- list 2: pairs of {monitor index, vertical geometrical error}.

CalibrationFile:

Calibration factors (e.g. units [A/mrad]) of the correcting magnets. The file consists of a single list:
- list 1: pairs of {corrector index, calibration factor}.

SettingFile:

Present settings (e.g. units [A]) of the correcting magnets. The file consists of a single list:
- list 1: pairs of {corrector index, present setting}.

MatrixFile:

Matrix of corrections for MICADO. The file consists of an array of m lines and n colums (m and n being the number of monitors and correctors). For convenience the corrector index list is added to the matrix. The file ends with the character #.

## ■ Correction matrix

### ■ Theoretical matrix

**MicadoMatrix[m,n,type,plane,options]**

return the matrix of corrections for the chosen m monitors and n correctors. The matrix has m lines and n columns.

The matrix element (i,j) yields the trajectory deviation at the i-th monitor due to a unit kick at the j-th corrector (assumed to be a thin lens).

#### □ Input arguments

*m*

number of monitors or list of monitor index (e.g. 4 or {1,2,3,4} or {43,45,47,50}).

*n*

number of correcting magnets (e.g. 2 or {1,3} or {2,4}). Arguments m and n must be compatible. The integer type (e.g. m=4 and n=2) is convenient when monitors and correctors are simply ordered as the integers (i.e. {1,2,3...}).

*type: Closed, Open*

switch to closed or open orbit formula (compute the correction matrix of a periodic machine or a transport line).

*plane: Horizontal, Vertical*

select the plane of correction.

#### □ Input options

*MonitorFile->"<file_name>"*

optical parameter monitor file name. By default: monitorFile->"/u/martinim/math/fi42mon.opt".

*CorrectorFile->"<file_name>"*

optical parameter corrector file name. By default: CorrectorFile->"/u/martinim/math/fi42cor.opt".

*CalibrationFile->"<file_name>"*

calibration factor file name. By default: CalibrationFile->None (the calibration factors are set to unity).

*DeletedCorrector->{dc1,dc2,...}*

where dc1, dc2, ... are correctors to be eliminated from the correction.
By default: DeletedCorrector->{}.

*DeletedMonitor->{dm1,dm2,...}*

where dm1, dm2, ... are monitors to be eliminated from the correction.
By default: DeletedMonitor->{}.

*CoherentOscillation->True, False*

See Micado function.

*Tune->Qx,y*

where Qx,y is the tune (e.g. the measured tune). By default: Tune->None (Qx,y is the lattice tune in MonitorFile).

*Turn->number*

where number is the turn number from which the trajectory in a periodic machine is measured. By default: Turn->1.

#### □ Output option

*MatrixFile->"<file_name>"*

Correction matrix file name. By default: MatrixFile->"/tmp/matcor.opt".

### ■ Experimental matrix

The matrix of corrections may alternatively be built experimentally and stored in a file. It has the same form as the correction matrix derived by calculation.

## ■ Correction

**Micado[m,n,plane,positions,ncorrectors,options]**

return the list of the selected correctors with their excitations (e.g. in [A]), and the list of the squares root of the residual vector norms [mm].

□ **Input arguments**

m

number of monitors or list of monitor index.

n

number of correcting magnets.

plane: Horizontal, Vertical

select the plane of correction.

positions

list of measured positions at monitors [mm].

ncorrectors

impose a given number of correctors chosen among n magnets.

□ **Input options**

MonitorFile->"<file_name>"

See MicadoMatrix function.

ReferenceFile->"<file_name>"

reference position file name. By default: ReferenceFile->None (no reference).

GeometricalFile->"<file_name>"

geometrical error file name. By default: GeometricalFile->None (no error).

SettingFile->"<file_name>"

present setting file name. By default: SettingFile->None (no present setting).

MatrixFile->"<file_name>"

See MicadoMatrix function.

DeletedCorrector->{dc1,dc2,...}

See MicadoMatrix function.

DeletedMonitor->{dm1,dm2,...}

See MicadoMatrix function.

Reset->True, False

Correction of either the measured orbit or the bare orbit (by resetting the present corrector setting in SettingFile). By default: Reset->True.

CoherentOscillation->True, False

True/False: ignores/subtracts the contribution of the momentum dispersion error to the measured positions. By default: CoherentOscillation->False.

□ **Output option**

OutputFile->"<file_name>"

The file consists of 6 lists:
- list 1: squares root of the residual vector norms [mm] (from 0 to ncorrectors),
- list 2: pairs of {corrector index, corrector strength} (after elimination of faulty correctors),
- list 3: relative momentum error (1 value),
- list 4: pairs of {monitor index, measured position} (after elimination of bad monitors),
- list 5: pairs of {monitor index, filtered position} (after subtraction of the reference positions, the geometrical errors, and the momentum error),
- list 6: pairs of {monitor index, expected position}, (after correction).

By default: OutputFile->"/tmp/output.dat".

# PS injection tuning based on MICADO strategy

M. Martini - 7 November 1994
Notebook: Micadops.ma

## ■ Initialization

## ■ PS optical parameters

## ■ Read present corrector strengths and beam positions

### ■ Read present corrector strengths

```
presentCorrCurrent=
ReadEquipment[
    {"BTP.DVT40",CCV,plsLine},{"BTP.DHZ40",CCV,plsLine},
    {"BTP.DVT50",CCV,plsLine},{"PI.SMH42",CCV,plsLine}]
```

```
{-0.91, -5.31, -0.77, 1911.}
```

### ■ Set calibration factors and present settings of correctors

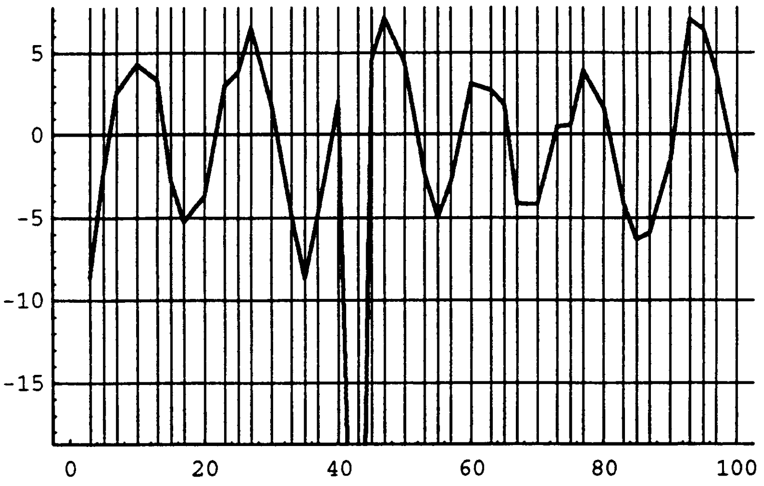
### ■ Read trajectory positions using CODD measurement device

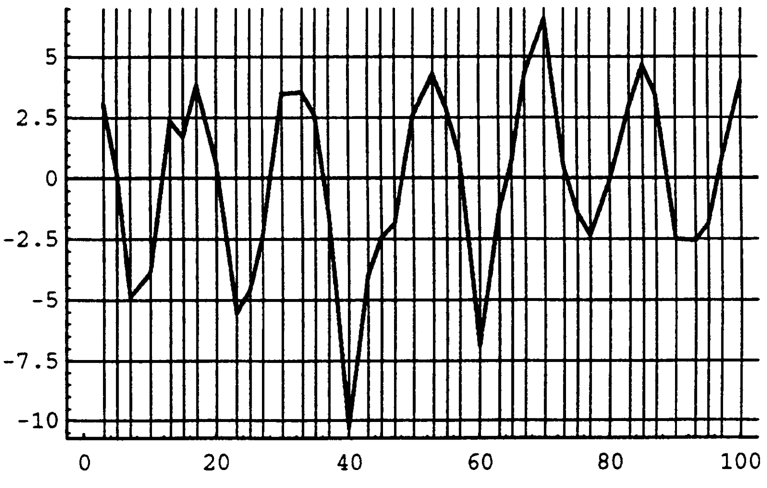
### ■ Plot the horizontal and vertical two turn trajectory differences

#### □ Prepare plots

#### □ Plots

```
Show[pMH,DisplayFunction->$DisplayFunction];
Show[pMV,DisplayFunction->$DisplayFunction];
```





## ■ Elimination of bad monitors

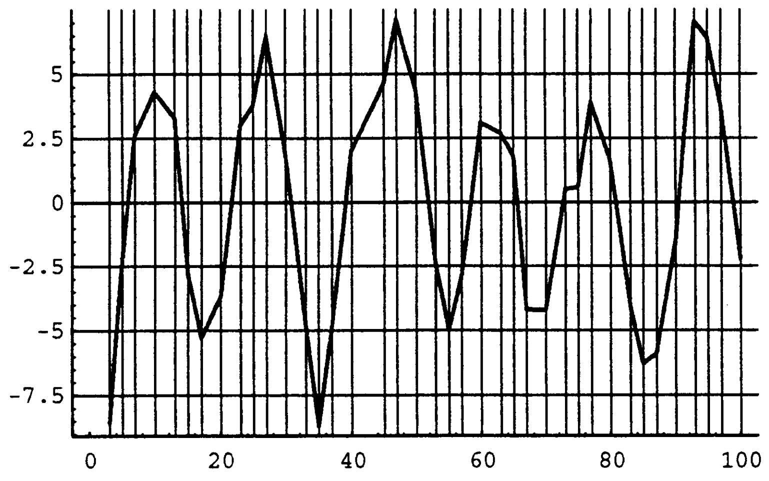
```
DeletedPuH=
Transpose[Cases[puPosition[[planeH]],
                {_Integer,x_/;(Abs[x]>20)}]][[1]]
DeletedPuV=
Transpose[Cases[puPosition[[planeV]],
                {_Integer,x_/;(Abs[x]>20)}]][[1]]
puPositionH=
DeleteCases[puPosition[[planeH]],{_Integer,x_/;(Abs[x]>20)}];
puPositionV=
DeleteCases[puPosition[[planeV]],{_Integer,x_/;(Abs[x]>20)}];
{43}
{}
```

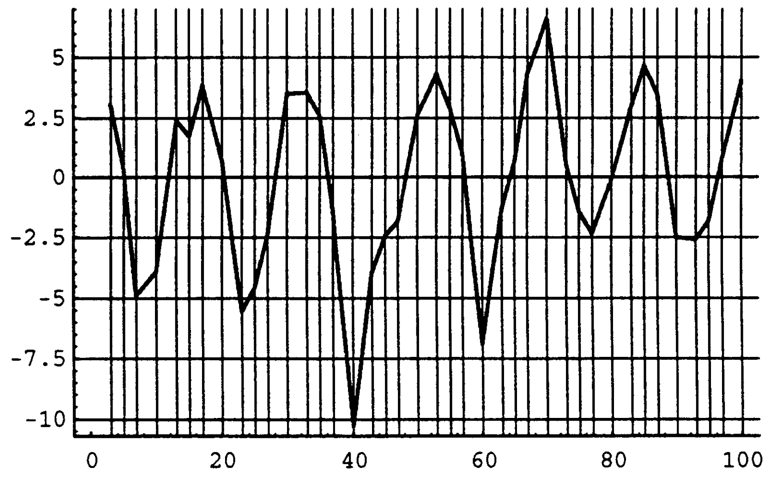
## ■ Plot the horizontal and vertical two turn trajectory differences after elimination of bad monitors

### □ Prepare plots

### □ Plots

```
Show[pMdH,DisplayFunction->$DisplayFunction];
Show[pMdV,DisplayFunction->$DisplayFunction];
```





## ■ Horizontal correction

The 2 corrector magnets are: DHZ40 and SMH42.

### ■ Arrange monitor list: start with the 1st monitor downstream PS injection (PU43)

### ■ Theoretical matrix of correction

### ■ Correction

```
correctorNumber=2;
Micado[Transpose[puPositionHR][[1]],{2,4},Horizontal,
        Transpose[puPositionHR][[2]]/N[2*Sin[Pi*qH]],
        correctorNumber,
        DeletedMonitor->DeletedPuH,
        SettingFile->"/tmp/fi42set.opt",
        CoherentOscillation->True,Reset->False,
        MonitorFile->"/u/martinim/math/fi420mon.opt"
       ]
{{34.4752, 31.515, 12.65}, {2, 4}, {-6.17807, -37.6372}}
```

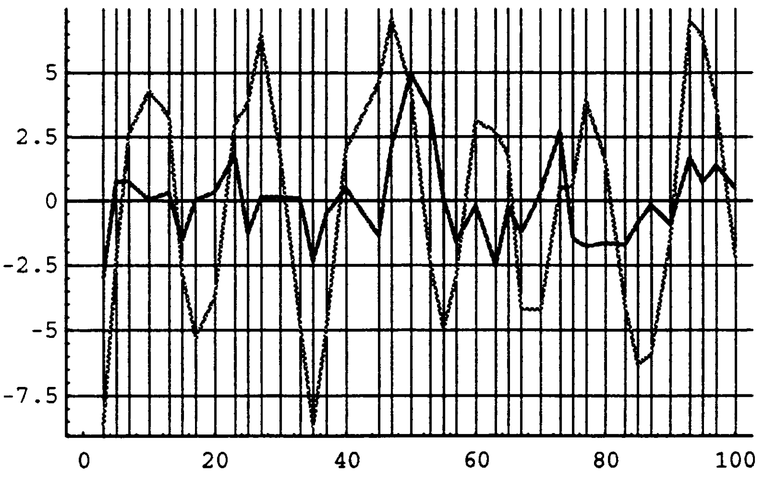
### ■ Arrange monitor list for plots: start with monitor PU3

### ■ Plot the uncorrected orbit and the expected orbit after correction

#### □ Prepare plot

#### □ Plot

```
Show[pMdH,pSH,pEH,DisplayFunction->$DisplayFunction];
```



### ■ Set new horizontal corrector strengths

```
newCorrCurrent=
correctorSetting[FullCorList,nominalCorrCurrent,outputMicado,
                  SettingFile->"/tmp/fi42set.opt",
                  Reset->False
                 ]
{-0.91, -11.4881, -0.7700000000000001, 1873.36}
WriteEquipment[{"BTP.DHZ40",RESERV,plsLine,1},
               {"PI.SMH42",RESERV,plsLine,1},
               {"BTP.DHZ40",CCV,plsLine,newCorrCurrent[[2]]},
               {"PI.SMH42",CCV,plsLine,newCorrCurrent[[4]]},
               {"BTP.DHZ40",RELEAS,plsLine,1},
               {"PI.SMH42",RELEAS,plsLine,1}
              ];
```

## ■ Vertical correction

The 2 corrector magnets are: DVT40 and DVT50.

### ■ Arrange monitor list: start with the 1st monitor downstream PS injection (PU43)

### ■ Theoretical matrix of correction

### ■ Correction

```
correctorNumber=2;
Micado[Transpose[puPositionVR][[1]],{1,3},Vertical,
        Transpose[puPositionVR][[2]]/N[2*Sin[Pi*qV]],
        correctorNumber,
        DeletedMonitor->DeletedPuV,
        SettingFile->"/tmp/fi42set.opt",
        CoherentOscillation->True,Reset->False,
        MonitorFile->"/u/martinim/math/fi420mon.opt"
       ]
```

```
{{13.7902, 8.32953, 4.89858}, {1, 3}, {-4.29609, 3.41701}}
```

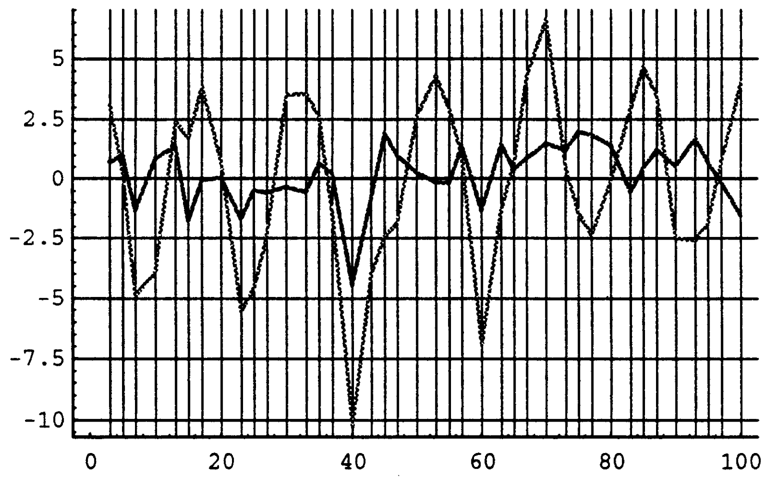
### ■ Arrange monitor list for plots: start with monitor PU3

### ■ Plot the uncorrected orbit and the expected orbit after correction

#### □ Prepare plot

#### □ Plot

```
Show[pMdV,pSV,pEV,DisplayFunction->$DisplayFunction];
```



### ■ Set new vertical corrector strengths

```
newCorrCurrent=
correctorSetting[FullCorList,nominalCorrCurrent,outputMicado,
                 SettingFile->"/tmp/fi42set.opt",
                 Reset->False
                ]
```

```
{-5.20609, -11.48806541594011, 2.64701, 1873.36}
```

```
WriteEquipment[{"BTP.DVT40",RESERV,plsLine,1},
               {"BTP.DVT50",RESERV,plsLine,1},
               {"BTP.DVT40",CCV,plsLine,newCorrCurrent[[1]]},
               {"BTP.DVT50",CCV,plsLine,newCorrCurrent[[3]]},
               {"BTP.DVT40",RELEAS,plsLine,1},
               {"BTP.DVT50",RELEAS,plsLine,1}
              ];
```

## ▪ Re-read beam trajectory positions for checking

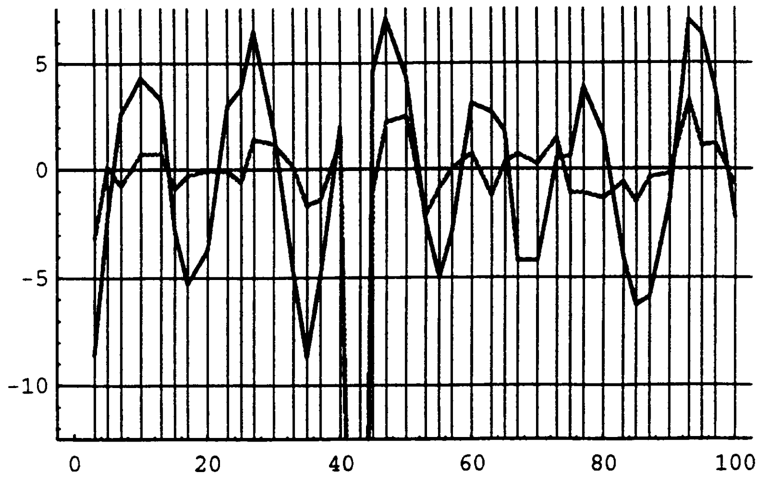
### ■ Read trajectory positions using CODD measurement device

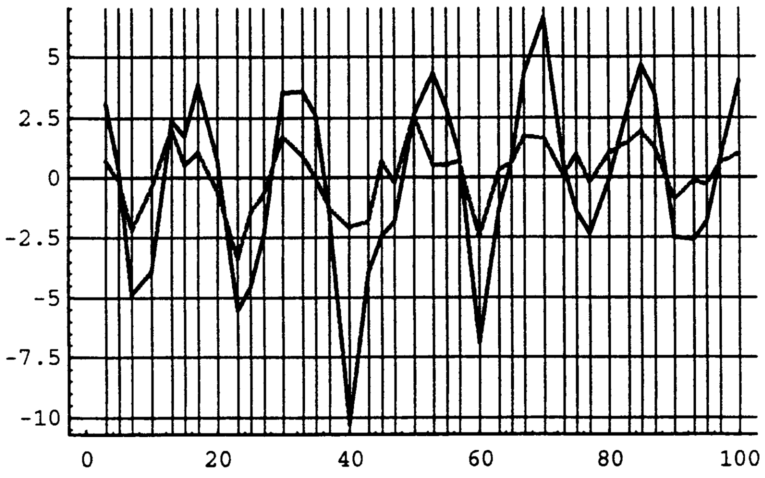
### ■ Plot the horizontal and vertical two turn trajectory differences

□ **Prepare plots**

□ **Plots**

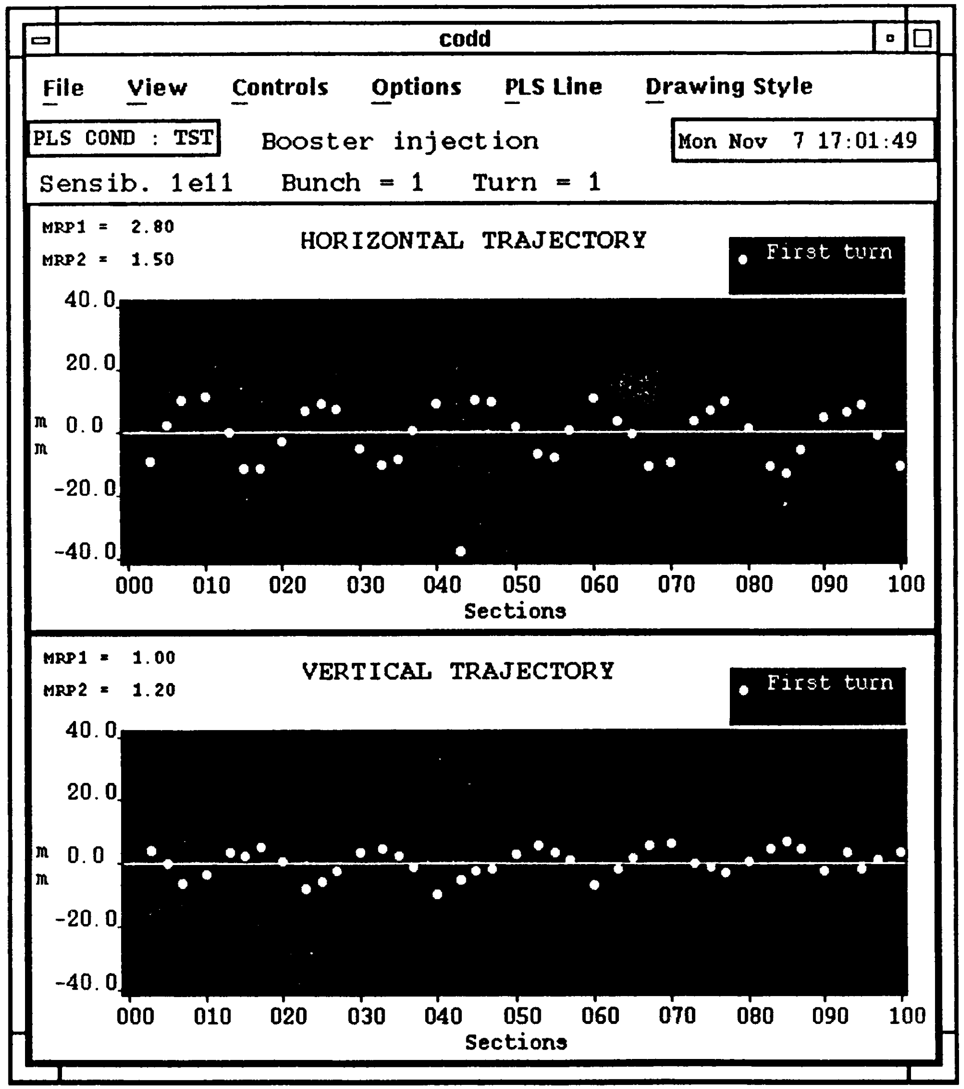
```
Show[pMH,pH,DisplayFunction->$DisplayFunction];
Show[pMV,pV,DisplayFunction->$DisplayFunction];
```





### ■ Back to initial corrector strength values

### ■ Update present settings of correctors



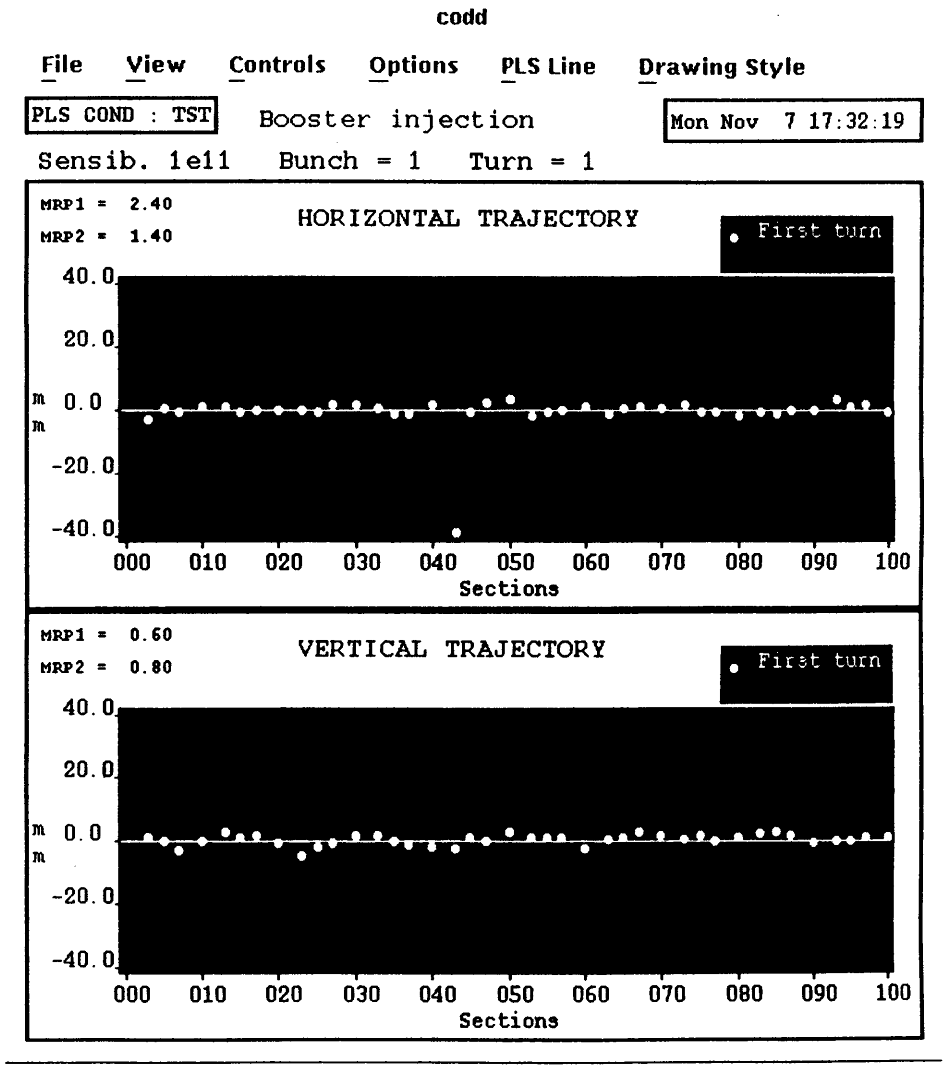
PLS COND : TST   Booster injection   Mon Nov 7 17:01:49

Sensib. 1e11   Bunch = 1   Turn = 1

MRP1 = 2.80
MRP2 = 1.50

MRP1 = 1.00
MRP2 = 1.20

BEFORE CORRECTION



PLS COND : TST   Booster injection   Mon Nov 7 17:32:19

Sensib. 1e11   Bunch = 1   Turn = 1

MRP1 = 2.40
MRP2 = 1.40

MRP1 = 0.60
MRP2 = 0.80

AFTER CORRECTION