USPAS 2015 June – Beam Based Diagnostics

Computer Class – Session 3

Xiaobiao Huang (6/12/15)

Goals: Practice optics and coupling correction with LOCO.

Part 1: LOCO with simulated SPEAR3 data

Open the script "wednesday_loco.m" and follow the instructions.

Cell 1: Initialization

Set MML path for SPEAR3. Put in optics and coupling errors to the lattice model. Also put in BPM gain errors to a few BPMs. Save the perturbed lattice as the reference.

Cell 2: Take LOCO data (in simulator mode)

With MML, one can measure LOCO data with

```
>> measlocodata
```

Data to be taken include the orbit response matrix, dispersion, and BPM noise level. Several popup dialogs will show up. Click "Yes" to save data to default directories. You can skip the BPM noise data since they cannot be measured in the simulator mode. We load a BPM data file and replace the data with a random sample from the Gaussian distribution.

Cell 3: Build up the LOCO input data

To set up the input file for the Matlab LOCO GUI, run

```
>> buildlocoinput
```

There will be several popup dialogs.

For "New LOCO Input File Name", type File name "locoin" and press Enter.

For "AT Model", accept the default model.

For "Response Matrix", click "Get From File" and choose from most recent LOCO data directories (for data that we just saved).

For "Dispersion", do the same as above.

For "BPM Sigma", click "Get From File" and choose the file "BPMData_sim.mat" in the working directory.

A Matlab daa file "locoin.mat" will be created.

Cell 4: Run Matlab LOCO GUI to fit LOCO data

To open the Matlab LOCO GUI, run

```
>> locogui
```

Click menu "File -> Open" to select the LOCO input data we just created, "locoin.mat".

Click menu "Inputs -> Weight for Horizontal Dispersion" and change it to 20.

Change the "Plot Type" to "Beta Beat From Iteration 0: Horizontal" and "Beta Beat From Iteration 0: Vertical" for the left and right plots, respectively.

Change “# of Iterations” to 2, and click the “Start” button to run fitting.

Observe the change of χ^2 from iteration 0 to 2, and the change of beta beat in the plots. Change the “Plot Type” to see the BPM gains, horizontal and vertical dispersions, and other data.

Cell 5: Inspect fitting results

In addition to examining fitting data and results with the GUI, one can also export fitting results to the Matlab working environment as variables. Or alternatively, one can load the data directly from “locoin.mat”.

Load the fitting parameters from “locoin.mat”

```
>> load locoin.mat FitParameters
```

Then, plot the changes of fitting parameters and compare to the target values. The results are as shown in Figure 1. Note that the fitted parameters do not exactly recover the target values. Instead, errors are also found for nearby quadrupole magnets.

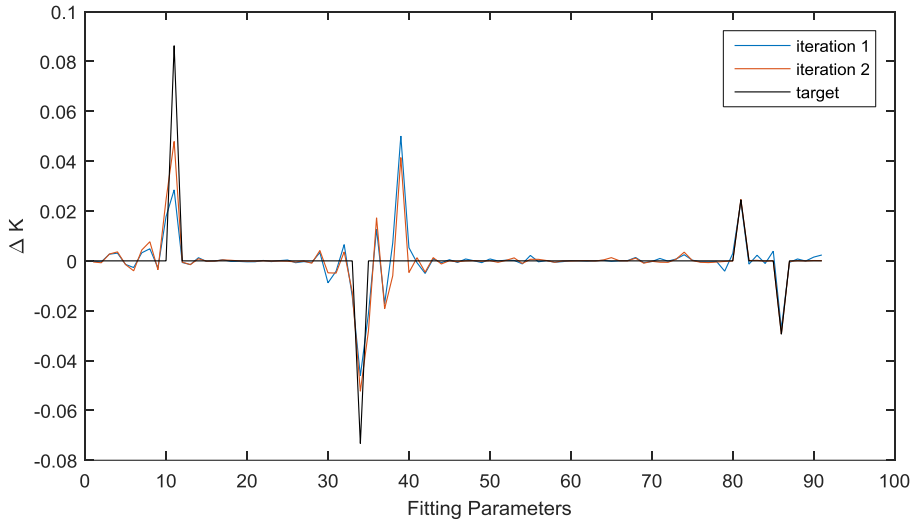

Figure 1: changes of quadrupole gradient variables (ΔK) after fitting are compared to the target values. The first 78 parameters are quadrupoles and the last 13 are skew quadrupoles.

Cell 6: Comparison of beta beat and coupling

We now calculate the beta beat for the fitted lattice and the target lattice with respect to the nominal lattice and compare the results. Even though the fitted quadrupoles are not exactly the same as the target, the optics of the two lattices are nearly the same.

We also calculate the coupling of the fitted lattice and the target lattice with the function “calccoupling”.

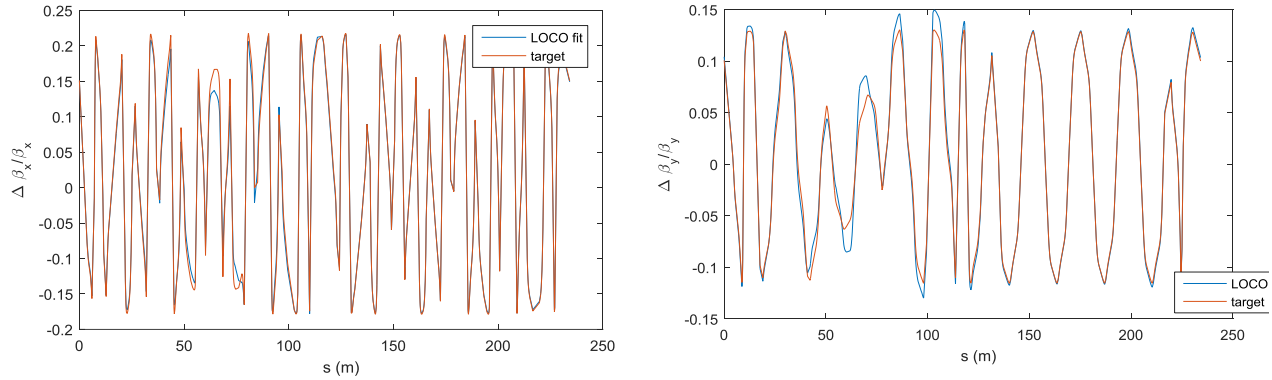

Figure 2: beta beat of the fitted lattice and the target lattice.

Part 2: LOCO without Matlab Middle Layer

For machines without a working MML setup, Matlab LOCO can still be used for optics measurement and correction. We use this example to show how to build up the LOCO input with raw measurement (not structured by MML). This example is for SNS (Spallation Neutron Source).

Enter the subdirectory 'loco_no_MML', open the script 'locobuildinput.m' and follow the instructions.

Cell 1: Build RINGData from MAD8 Twiss output

An AT lattice deck can be built from MAD8 output of the Twiss command using the function 'readmad.m'. The lattice is supplied to the structure 'RINGData'.

Cell 2: Build LOCO measurement data

LOCO measurement data structures include 'LOCOMeasData', 'BPMData' and 'CMData'. For this example, there are 24 horizontal correctors, 26 vertical correctors, and 38 BPMs. We build up these data structures by filling out the required fields. Many of the fields are only for the purpose of keeping records and are thus non-essential.

Cell 3: Set up fitting parameters for LOCO

Without MML, we have to locate the quadrupole magnets in the lattice model, and group them together as fitting parameters if needed. Note that the data structure 'FitParameters' has two required fields: 'Values' is a vector that holds the numerical values of the fitting lattice parameters, 'Params' holds the location of the magnets and the subfields to be fitted (usually 'K' for quadrupoles if 'QuadLinearPass' is used), and 'Deltas' is a vector of step changes in Jacobian matrix calculation.

In this example 6 families of quads are fitted, grouped by power supplies.

Cell 4: Set up LOCO fitting flags

The LOCO fitting flags are stored in 'LocoFlags'. These flags can be changed with the GUI. Finally the data structures are validated with the function 'locofilecheck.m' and saved to a LOCO input file.

Part 3: Correction of optics distortion by an insertion device in the lattice model with LOCO

Insertion devices cause optics distortion to the ring. On the real machine such optics errors are corrected with LOCO as usual. In this exercise we demonstrate how to incorporate the ID effects into the AT model and the correction of the optics errors in the model. BL11 of SPEAR3 is used here as an example.

Enter the subdirectory 'ID' and open the script 'wednesday_BL11.m' and follow the instructions.

Cell 1: Preparation of lattice models

Each of the scripts 'latticeprepmad.m' and 'latticeprepwig.m' generates a lattice file. The two lattice models have the same number of elements. But the ID effect in the one generated by 'latticeprepmad.m' is turned off by setting the corresponding passmethods to 'DriftPass'.

The dynamic effect of the ID is represented with a kick table and the corresponding passmethod 'WigTablePass.m'. The static effects (first field integrals) are represented by multipole fields with passmethod 'ThinMPolePass'.

Cell 2: Checking the kick table

We plot the horizontal kick curve at $y = 0$ and the vertical kick curve at $x = 0$. The integrated vertical focusing gradient is found by fitting the kick vs. y curve. The horizontal dynamical kick (Figure 3 top) is due to transverse field roll-off at about $x \approx \pm 25$ mm.

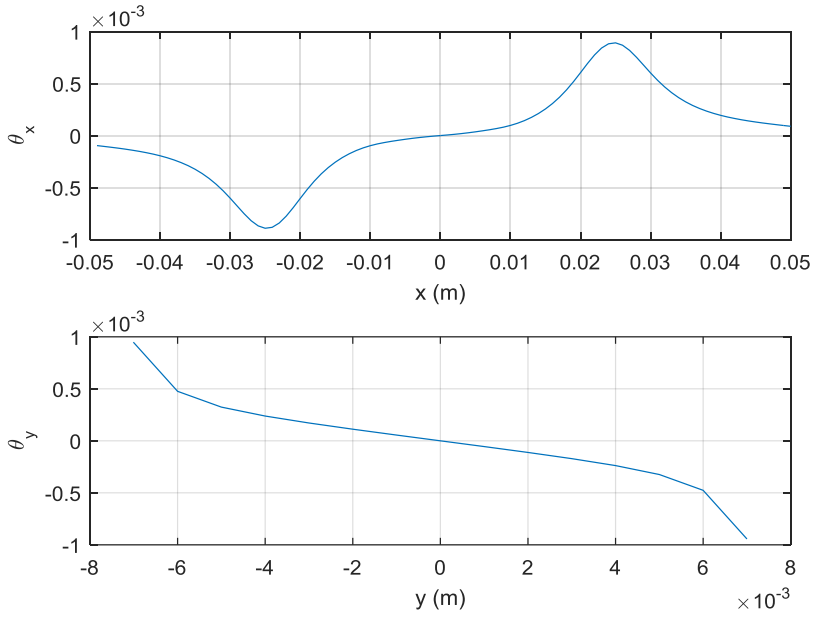

Figure 3: kick curves on the middle planes.

Cell 3: Cancellation of horizontal kick with multipole field

The nonlinear horizontal kick of BL11 caused much trouble in operation. The problem was solved by introducing "magic fingers" to the device to modify the field map and cancel the nonlinear kick. The multipole kick due to the "magic fingers" are plotted and compared to the dynamic kick (Figure 4).

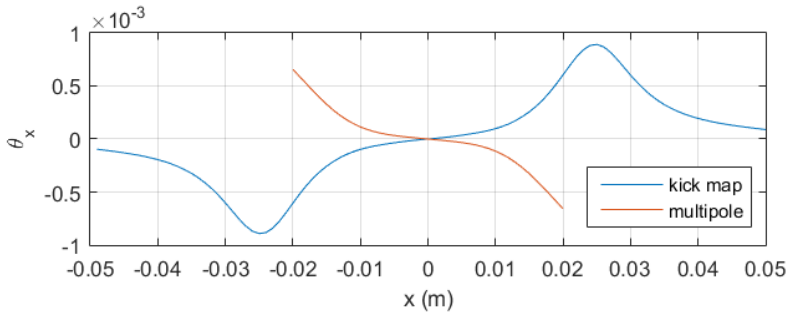

Figure 4: the multipole kick due to SPEAR3 BL11 “magic fingers”.

Cell 4: Build LOCO input for optics correction

The vertical dynamic kick of the ID causes significant beta beating on the vertical plane. When we study the nonlinear effects of the ID, it is desirable to correct the optics errors first. This can be done with LOCO.

We build a LOCO input file for this purpose. The initial lattice is the lattice with optics errors. The orbit response matrix data are generated with the ideal lattice.

Cell 5&6: LOCO fitting and checking result.

Use LOCO GUI to fit the data. Check the vertical optics (beta function) before and after two iterations of fitting. Export the fitted lattice to the Matlab workspace and save to a file. This lattice with corrected optics can be used for nonlinear dynamics study.

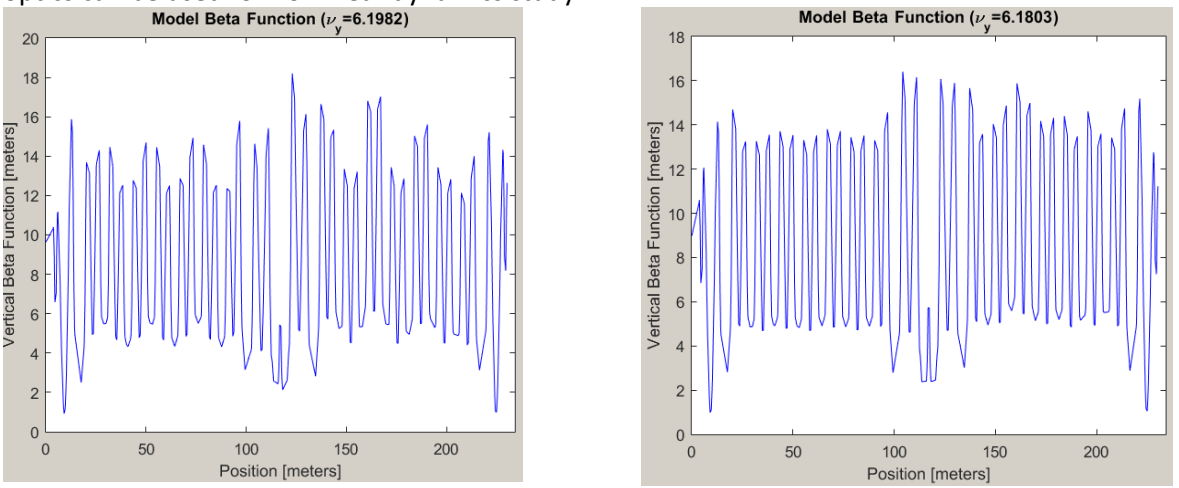

Figure 5: Vertical beta function before (left) and after (right) LOCO fitting.