# INTERACTIVE ORBIT CONTROL IN MATLAB [1]

J. Corbett and A. Terebilo, SLAC, Menlo Park, CA 94025, USA

*Abstract*

Recent advances in steering algorithms have made it possible to accurately control electron beam position in storage rings, implement fast and slow feedback systems, and in some cases detect hardware errors. In practice, however, the program operator would like to reduce the overhead of selecting variables and constraints and to easily view the data. To simplify the process, we constructed an interactive orbit control program in MATLAB [1]. The program modules are easily adapted to new algorithms or beam lines. This paper describes the program functionality and architecture.

## 1 INTRODUCTION

SSRL will soon replace the SPEAR 2 storage ring with a new version, SPEAR 3 [2]. The move invalidates most of the on-line accelerator application programs at SSRL, including a FORTRAN orbit control program using SVD [3]. Rather than adapt the old software to the new machine, we decided to rewrite the orbit control program in MATLAB to take advantage of easy-to-use mathematical and graphics routines [1]. Overall, the pure-MATLAB re-write reduced programming effort, streamlined program modules and increased operational functionality. The new program is EPICS compatible, interfaces to an Accelerator Toolbox simulation package [4,5] and can be adapted to different machines. The program structure is modular, provides easy access to data structures (but not object-oriented!) and allows pieces of the code to be used in future applications. The result is a flexible orbit control program with a graphical interface that is extensible to new control applications.

## 2 GRAPHICAL INTERFACE

The graphical interface is designed to provide easy manipulation of the electron beam orbit. The main operator actions (selection of BPM, corrector and eigenvectors) are performed graphically. To make orbit 'bumps' for instance, the operator simply 'drags' target BPMs to the desired position in the graphical interface, visually inspects the solution and applies correctors when ready. Command line control is optional.

### *2.1 General Layout*

Modeled in part after the SPEAR 2 program ORBIT [6], the orbit control interface for SPEAR 3 displays the electron beam trajectory and corrector currents in a single graphics window (Fig. 1). For convenience, the spectrum of singular values from the daigonalized response matrix is also displayed. Action commands (file i/o, corrector reset, display mode, etc) are carried out through simple push button sequences.

### *2.2 Orbit Display*

The orbit display (Fig. 1, top graph) shows a) reference (red), b) measured (blue), c) desired (red-dash) and d) predicted (green) electron beam trajectories. Color-coded circles indicate individual BPMs. BPMs can be toggled on/off graphically to control which BPMs serve as fitting constraints. Coefficients can also be entered for weighted-SVD control. Each time a BPM icon is selected, orbit information is displayed to the screen. A complete listing of all BPM related parameters is available via the BPM 'Show State' button. Columns of the response matrix and individual orbit-eigenvectors can be displayed on request. Control of plotting scales is achieved through the top-level menu bar. By graphically moving BPM icons in the 'drag' mode, the operator can manipulate the *desired* orbit to a new orbit position. Subsequent fitting calculations steer the beam toward the desired position.

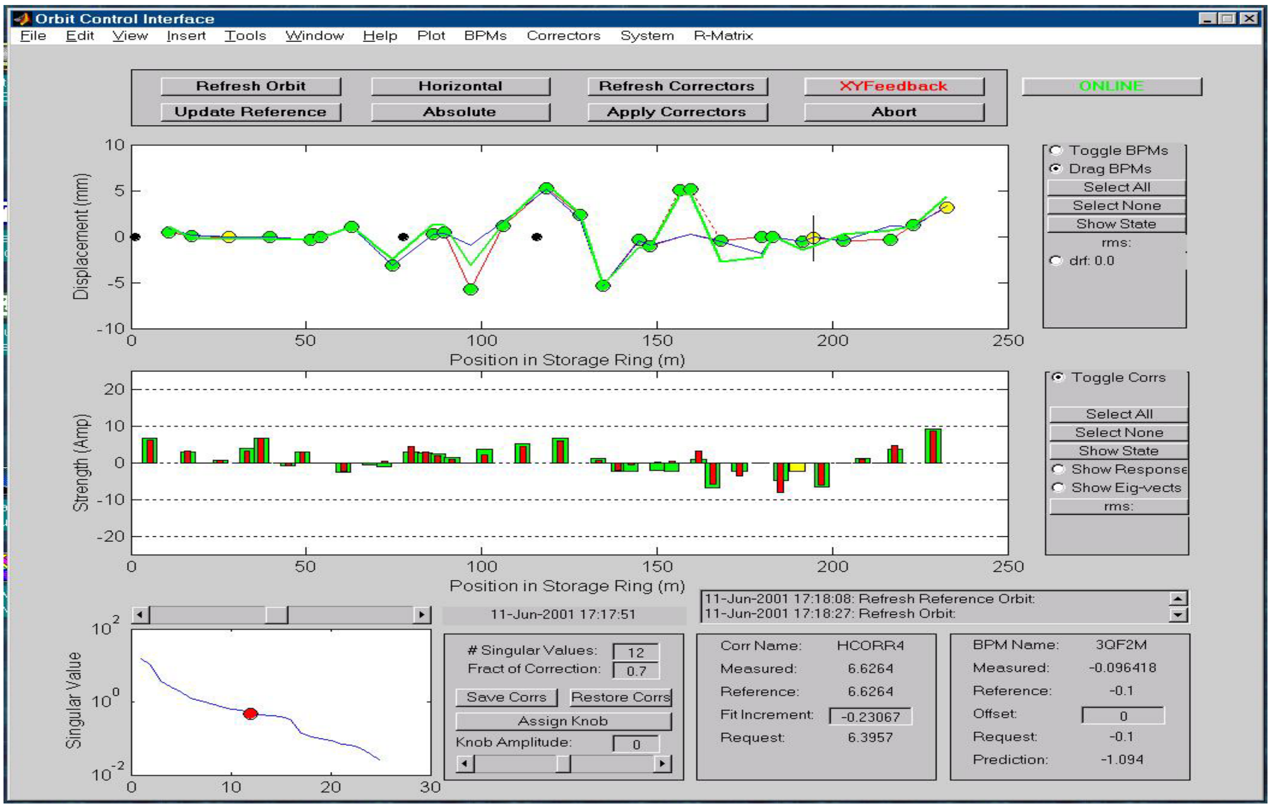
### *2.3 Corrector Display*

The corrector display (Fig. 1, middle graph) shows a) measured (green) and b) predicted (red) magnet values in bar plot form. Corrector icons can be graphically toggled on/off to control which magnets act as variables in the fitting algorithm. Individual magnet weighting also influences the fit. When a corrector icon is selected, magnet information is displayed to the screen. Activating the 'Show Response' button displays the BPM response to the individual corrector. A complete listing of all corrector related variables is available via the 'Show State' button. 'Save', 'Restore' and 'Assign Knob' pushbuttons control corrector magnet patterns. With 'Assign Knob', the operator activates a variable-amplitude slider which can be used for closed-bump manipulation. Control of plot scales is available through the top-level menu bar.

### *2.4 Singular Value Display*

The orbit control algorithm for SPEAR 3 will be based on weighted Singular Value Decomposition of the response matrix [7]. The spectrum of singular values is plotted in the program interface to simplify choice

[1]Work supported in part by DOE Contract DE-AC03-76SF00515 and Office of Basic Energy Sciences, Division of Chemical Sciences.



11-Jun-2001 17:17:51

11-Jun-2001 17:18:08: Refresh Reference Orbit:
11-Jun-2001 17:18:27: Refresh Orbit:

| | |
|---|---|
| # Singular Values: | 12 |
| Fract of Correction: | 0.7 |
| Knob Amplitude: | 0 |

| | |
|---|---|
| Corr Name: | HCORR4 |
| Measured: | 6.6264 |
| Reference: | 6.6264 |
| Fit Increment: | -0.23067 |
| Request: | 6.3957 |

| | |
|---|---|
| BPM Name: | 3QF2M |
| Measured: | -0.096418 |
| Reference: | -0.1 |
| Offset: | 0 |
| Request: | -0.1 |
| Prediction: | -1.094 |

Figure 1: Orbit control interface in MATLAB..

of cut-off value. By graphically selecting the number of eigenvectors, the operator immediately sees the predicted orbit correction and predicted corrector values in their respective plots. Orbit perturbations associated with individual eigenvectors can be viewed by activating the 'Show Eigenvector' pushbutton. When parameters in the fitting matrix are changed (BPM or corrector selection, weights, etc) the program automatically re-inverts the matrix, backsubstitutes, and displays the result. When the 'RF' button is selected, the dispersion component of the orbit is removed prior to backsubstitution.

### *2.5 Graphical Action Control Elements*

Graphical action control elements include push button with callbacks to refresh orbit, refresh correctors, switch planes, view orbit in absolute/relative mode, apply correctors, scale bumps, etc. A top-level menu bar also allows the operator to scale axes, read and write files and measure response matrices. The menu bar is easily expanded to perform auxiliary computations or launch new application programs.

## 3 PROGRAM STRUCTURE

The structure of the orbit program is arranged to accomplish several objectives:

*1. Separate mathematical, graphical, data i/o and file i/o routines*

*2. Separate BPM, Corrector and SVD graphic routines*

*3. Provide compact data structures for use outside of the graphics environment*

The first two objectives address basic programming discipline: the MATLAB code is arranged in folders with software modules that control graphics, respond to BPM and corrector manipulations and carry out SVD operations, respectively. To reduce interface complexity, graphical elements can be removed or callbacks disabled when the graphics window is initialized. Orbit correction can also be run in the background with no graphics, or program modules normally activated by GUI callbacks can be activated from the command prompt.

The third objective (compact data structures) is accomplished by defining several global data structures:

BPM, COR, MAG, RSP, SYS

The BPM, COR and MAG structures contain fields for element names, on/off for fitting, reference value, etc. The BPM structure has about 30 fields with BPM(1) and BPM(2) containing horizontal and vertical data, respectively. RSP contains response matrix data. The list of fields in each structure is easily extended. From the

MATLAB command prompt, for instance, the operator could create a '.beta' field to plot betafunctions at each BPM. The data structures can also be used to develop new orbit control algorithms that utilize the MATLAB Accelerator Toolbox [4,5], utilize the graphical display options and/or connect to the on-line machine. The SYS structure contains high level control parameters for the orbit control system.

File handling is controlled through top-level menu bars. Read/write options are available for the BPMs, correctors, response matrices and to save the system state for recall at a future time. At present, we utilize file formats from the SPEAR 2 orbit feedback program for system compatibility and testing.

## 4 DEVICE CONTROL

Access to the accelerator hardware is handled through machine-specific routines to acquire and control database parameters (BPMs, correctors, main magnets, rf frequency, etc). Depending on whether the program is in the *simulate* or *on-line* mode, the 'GET' and 'SET' commands communicate with the MATLAB Accelerator Toolbox (simulation) or with actual hardware via EPICS Channel Access (CA) (on-line). Details of the Accelerator Toolbox are described in [4,5]. For the online mode, MATLAB compatible CA routines have been compiled into a Channel Access toolbox (MCA) [8]. In this case, the MATLAB application acts as a CA client and communicates with a CA server on the central control computer. The server maps the SPEAR parameter database to EPICS process variables [9].

One requirement for robust control applications like the orbit feedback program is to perform sequences of operations in the background. A sequencer waits for a specified period of time or until new data is available before acting on the data. Two examples of sequenced operations are response matrix measurement and orbit correction feedback. At the same time, the program must respond to the asynchronous actions from the user such as aborting feedback. In a non-real-time, single-threaded environment such as MATLAB, it is not straightforward to satisfy this requirement. We resolved this problem by using (Windows) system timer calls within MATLAB.[2]

## 5 SIMULATION MODE

The simulation mode utilizes the Accelerator Toolbox (AT) for the machine model. AT provides a wide range of simulation options ranging from 6-D symplectic tracking to closed orbit calculations and coupled optics. The computationally intensive components of the AT software are written in native c-code compiled into a dynamic link library to execute quickly [4,5]. By issuing GET/SET commands in the simulation mode, the program reads BPM values (closed orbits), adjusts corrector values and re-computes the closed orbit in the AT model. Main magnet control is also possible (dipoles, quadrupoles, etc). By running background software to perturb element values in the accelerator model, realistic orbit feedback control can be simulated without utilizing beam time.

[2]This functionality is a part of the MCA Toolbox [9]. It extends the capabilities of MATLAB to create a pseudo-multitasking, pseudo-real-time environment in which multiple MATLAB applications can run and exchange data through the MATLAB workspace

## 6 SUMMARY & FUTURE WORK

A graphical orbit control program has been written in MATLAB for SPEAR 3 applications. The program structure is modular to permit rapid development of new algorithms or expansion of code functionality. Orbit control, for instance, can be accomplished either through the graphical interface or through external commands. The main data structures can be accessed as global variables to use at the command prompt or in other application programs. The basic orbit control functions have been tested on-line with SPEAR 2. The program will be adapted to SPEAR 3 in the simulation mode with the MATLAB Accelerator Toolbox prior to on-line deployment in 2003. Adaptation to new machines requires new element definitions in the model, redefining file i/o formats and linking to the new database protocol. Future work includes adding dispersion correction, extending the orbit correction to include x-ray BPMs, adding pre-defined closed bump features, corrector ironing and developing 'smart' orbit correction algorithms.

## ACKNOWLEDGEMENTS

The authors would like to thank Posu Yan, Damian Hamilton and Harvey Rarback for help with this work. A good example of an accelerator simulaor which uses Windows GUI programming and features a multi-task interface display is TRACY-V [10].

## 7 REFERENCES

[1] MatLab, The MathWorks, Natick, MA
[2] J. Corbett, et al, "SPEAR 3 Upgrade Project: A Status Report", these proceedings, or "SPEAR 3 Design Report", SLAC Pubs, 1999
[3] W.J. Corbett and D. Keeley, "Orbit control on SPEAR: A Progress Report", AIP Conference Proceedings 315, Upton, NY 1993
[4] A. Terebilo, "Accelerator Toolbox for MATLAB", SLAC-PUB-8732, February, 2001. www-ssrl.slac.stanford.edu/AT/
[5] A. Terebilo, "Accelerator Modeling with MATLAB Accelerator Toolbox", these proceedings
[6] M. Donald, et al, "Some Schemes for Online Correction of the Closed Orbit, Dispersion and Beta Functions in PEP", 1981 IEEE PAC, Wash. D.C., SLAC-PUB-2666
[7] A. Friedmann and E. Bozoki, "Eigenvector Method for Optimized Orbit Correction", AIP Conference Proceedings 315, Upton, NY 1993
[8] A. Terebilo, "Channel Access Toolbox for MATLAB" to be presented at ICALEPCS 2001, San Jose, CA.
[9] R. Rarback, "Old Wine in New Bottles – The SPEAR Control System Upgrade", 1999 ICALEPCS, Trieste, Italy
[10] H.Nishimura, "Accelerator Modeling and Control Using Delphi on Windows NT", IWCSMSA96, KEK Proceedings 97-19, 174