SSCL-N-819

Tunescan

B. Cole, J. Greene, and L. Singleton

Superconducting Super Collider Laboratory*
2550 Beckleymeade Ave.
Dallas, TX 75237

May 1993

*Operated by the Universities Research Association, Inc., for the U.S. Department of Energy under Contract No. DE-AC35-89ER40486.

1.0 INTRODUCTION

One of the most important factors affecting the performance of an accelerator is the choice of operating point. During the simulation of a proposed machine, the choice of operating point is usually somewhat *ad hoc,* frequently based upon an individual's experience with other machines. The defense of this practice is that this point is in some sense representative, usually based upon its not being "too near" any resonance lines, nearness being defined in terms of assumptions made about resonance widths. As many design decisions may be based on such simulations, it is highly desirable to obtain information about how the machines performance depends on the fractional tunes.

Tunescan is a joining of scripts and codes designed to address this issue, using a combination of Sun workstations and the Intel Hypercube. This facility enables the user to repetitively perform TEAPOT analyses and linear aperture determination (as defined in SSCL-SR-1038) for different fractional tunes. **Tunescan** uses **TEAPOT** to generate hfiles from a previously created thin-element file, then starts a script on the Hypercube, which employs **Vectrack** for the tracking portion of the run.

This note begins with the results of the first applications of this technique. Only the interested reader who might want to obtain and run this code for himself need read the entire document.

2.0 A TUNESCAN EXAMPLE: THE SSC COLLIDER

The Collider of the Superconducting Super Collider (SSC) has been extensively studied at the operating point (123.285, 122.265) by the Machine Simulation and Correction Group, the Collider group, and others. Another tune point closer to the half integer (123.435, 122.415) has also been proposed. Thus, the neighborhoods of these two points are natural first candidates for Tunescan. Figure 1 shows the tune plane in the neighborhood of the (123.285, 122.265) operating point, and Figure 2 shows the same information for the (123.435, 122.415) operating point. A certain amount of stochasticity or "noise" is visible, due primarily to the choice of grid spacing and the presence of narrow resonances; however, regions of distinctly better linear aperture are clearly visible.

Comparing the two, one sees that the (123.435, 122.415) operating point is considerably closer to a "good" region, but that the region near the (123.285, 122.265) point is considerably larger. The conclusion to be drawn is that shifting the tune from (123.285, 122.265) into the nearby region (perhaps (123.23, 122.19)) might increase the linear aperture by 30% over the (123.285, 122.265) result, and provide this aperture over a spread of tunes approximately twice as large as the region near (123.435, 122.415). Another interesting observation is the clear effect of the third integer resonances, as expected. Finally, the lack of significant impact of the coupling resonance might at first be puzzling, but it has a simple explanation: The machine is decoupled at every point in the tune plane, and the linear coupling is explicitly removed during the calculation of the smear, by the definition of the linear aperture.

The time required of a Sparc2 workstation for TEAPOT to produce a single tune point for Vectrack to track averaged 1 h. This included an initial tune correction, a closed orbit correction, a chromaticity correction, decoupling, and a final tune correction. The final correction was found to be required due to a shift caused by the previous operations.

3.0 WORK IN PROGRESS AND FURTHER APPLICATIONS

An interesting question at this point is, "What does the rest of the tune plane look like?" Figure 3 shows a "coarse scan" of the entire tune plane. Tunes for which the correction process did not succeed are black. (The machine became unstable.)

Possible future studies include characterizing the need/effectiveness of resonance correction schemes, studying other SSC lattices such as the High Energy Booster, and addressing issues of the effects of tune

"drift"—*i.e.* , changing the tune *without* recorrecting the lattice to simulate the drift of the operating point during a run.

4.0 ORGANIZATION AND FUNCTION OF THE SCRIPTS

4.1 Tunescan

A script that takes no arguments. Instead, the script is edited to specify all the files required for the run. Thus, the particular version of **tunescan** in a particular directory provides a record of the run done in that directory. Upon execution, **tunescan** first runs an error-checking program called **prescan,** which simply checks the existence of all the files in **tunescan** and lists them. When begun, **scanit** and **hyperbat** are called and backgrounded.

4.2 Scanit

Manages the **TEAPOT** runs. It reads a line from the file of tunes and uses the Unix facility 'sed' to insert the fractional portion of the x and y tune into the TEAPOT input file containing a "readfile" command, which uses the thin element file ('fort.7') for lattice information. It also contains an hwrite command to create the lattice file for the Hypercube. Once the hfile is created, it is moved to the hfiles directory and renamed according to the tunes. The hfile name is recorded in **.sunscanned_b**. This entire process is repeated for each tune pair. After the last tune has been used, the word "Done" is written to **.sunscanned_b**. To prevent disk overflow in the case of an error, **scanit** checks the number of files in the hfiles directory. If there are four or more, **scanit** sleeps for 10 min and checks again. Thus it is important that no irrelevant files exist in the hfiles directory.

4.3 Hyperbat

Checks the list of completed hfiles against a list of the hfiles that have been used by **Vectrack**, **.batscanned**. If they are identical but the last line of **.sunscanned_b** is not the word "Done", hyperbat goes to sleep. If the two are different, the first unprocessed file name is inserted into the **Vectrack** input file. The script then loads **Vectrack** onto a cube. Once the tracking is completed, the program **hfcn** is loaded, which combines all the files created by **Vectrack's** print_turns flag into a single file with an **_out** suffix. This file is now identical in format to a **TEAPOT** fort.8 file, and may be post–processed using standard TEAPOT tools. Next, the program **haptr** is loaded. **Haptr** calculates the linear aperture from the tracking data and from two command-line specifications, the tune shift and delta smear (0.005 and 0.05 by default). It writes the file name and apertures in both x and y to a file. This final output file has a prefix of **la** followed by the values of the tune shift and delta smear used by **haptr.** By default the file is called **la.005.05**. This entire process is repeated until the files **.sunscanned_b** and **.batscanned** are identical. See Figure 4 for a diagram of this process.

5.0 OPTIONAL SECOND CUBE

During typical use it was discovered that the Sun portion, **scanit**, required only half the time to do the lattice processing that **Vectrack** took to do the tracking. For these cases, the total processing speed could be doubled by having two sets of nodes associated with each invocation of **scanit**. This was done by creating a parallel cube script, now named **hypercat,** which gets another cube and tracks runs simultaneously. When this feature is employed, **scanit** sends the names of completed hfiles to **.sunscanned_b** and **.sunscanned_c** alternately. **Hypercat** is created from **hyperbat** using the sed file **.bat_to_cat**. **Hypercat** lists tracked files in **.catscanned**. It sends linear aperture output to the same **la** file as **hyperbat**. This has never presented a problem because **hypercat** always lags **hyperbat** by the length of one iteration of scanit. See Appendix C for use.

6.0 INPUT FILES

There are two input files, one for TEAPOT and one for **Vectrack**. These files are actually more like templates for creating tune-specific input files using sed. In the TEAPOT input file, the fractional portion of the tunes in a "tunethin" must instead be **TUNEX** and **TUNEY**. An example from ~greene/cube/scanner/run.collider:

```
tunethin, mux=123TUNEX,muy=122TUNEY,b1f=cqf[b1], b1d=cqd[b1],&
   numtries=4, stepsize=0.0005, tolerance=0.0001
```

IMPORTANT: this file must begin with "**readfile**" and include "**hwrite**" just before "**stop.**"

In the **Vectrack** input file, the lattice file must be hfiles/**FILENAME**. example from ~greene/cube/scanner/run.vectrack:

```
lattice = hfiles/FILENAME
```

An absence of a title is the only other difference from a typical **Vectrack** input file. **IMPORTANT**: The **print_turns** flag **MUST** be included.

7.0 EXECUTING TUNESCAN

To begin a **tunescan** run, first view **hyperbat** to see if the -d deletion flags are set as desired for **hfcn** and **haptr**. It is recommended that **hfcn** use -d at all times. The advantage of not using -d for **haptr** is the ability the use the **masshaptr** script. See Appendix B.

Next, **tunescan** is edited to specify the following:

7.1 TITLE

A basic description, the date, and file name will be added onto this string.

7.2 TUNEFILE

The file containing all the pairs of tunes to be scanned. The tunes are expected to be fractional, without a zero on the left side of the decimal, *e.g.*:

```
.285  .265
.285  .270
.290  .265
```

7.3 TPOT_INPUT

The input file template for **TEAPOT**.

7.4 VECTRACK_INPUT

The file containing the input and flags for **Vectrack.**

7.5 THIN_FILE

This is the thin-element file created by a previous **TEAPOT** run. To create the hfile for each tune, this file is read by the **TEAPOT** "readfile" command. This file is used as the <latticename> prefix for the **Vectrack** and **hfcn** output files.

7.6 NODES

The number of Hypercube nodes allocated (per cube) to **Vectrack**.

7.7 TWOCUBES

Set to either "ON" or "OFF," this decides whether **tunescan** uses one cube alone or two cubes simultaneously. The optimum setting will depend on the ratio of time required by the **TEAPOT** portion to that required by the Hypercube. If equivalent, or if only running one tune, "OFF" is the correct setting. The "ON" setting doubles the overall tracking rate. See Appendix C.

7.8 DIRECTORY

The directory off the cube directory in which the run is taking place. For ~greene/cube/scanner the correct setting would be "scanner."

When all of the above is satisfactory, execute **tunescan** from the UNIX prompt. All scripts are run in the background.

APPENDIX A
REQUIRED FILES

To perform a **tunescan** run, the following files are required:
~greene/cube/scanner/

tunescan	(the outermost script)
scanit	(the Sun run manager)
hyperbat	(the Hypercube script manger)
.bat_to_cat	(sed file to build the Hypercube script **hypercat**)
hfcn	(converts print_turns data to <latticename>_out)
haptr	(calculates linear aperture from <latticename>_out)
a **TEAPOT** input file	(for an example, see run.collider)
a **Vectrack** input file	(for an example, see run.vectrack)
a **Vectrack** particle file	(for examples, see load/ *.load)
a file of tunes	(for examples, see tunes/ tunes.*)

Hfcn and **haptr** each possesses a –d optional flag that will delete files after use.

APPENDIX B
SETUP AIDS

The following are setup aids for use with **tunescan**:

loadcon

A C program for the Sun that reads a file of **TEAPOT**–style particle declarations and writes a **Vectrack** particle file.

tunegrid (tunegrid2)

A C program for the Sun that creates a rectangular array of tune points. Tunegrid2 automatically eliminates leading zeros.

trigrid

A C program for the Sun that creates a triangular array of points. Leading zeros are eliminated.

The following are post-processing aids:

getdata

A C program for the Sun that reads an **la**.###.## file (linear aperture results) and writes a file containing the fractional x and y tunes and the lesser of the x and y apertures divided by the respective sigmas. Designed to produce data for Unigraph.

masshaptr

A Hypercube script that runs **haptr** on each **_out** file in a list contained in a file named **.masstodo**, appending an **la**.###.## file. This allows explorations of the **haptr** tuneshift and deltasmear settings without rerunning **tunescan**.

NOTE: the -d option for **haptr** in **hyperbat** must be removed prior to the run to take advantage of this script.

APPENDIX C
ADVICE AND POSSIBLE SOURCES OF ERRORS

You must have a user login for the Hypercube.

The directory in which **tunescan** runs are performed must be cross-mounted from the workstation to Sycamore, the System Resource Manager for the Hypercube.

If the machine from which you execute **tunescan** is not listed in the .rhosts file in your home directory on the Hypercube, the rsh calls will not work.

Two cubes or not two cubes

Experimentation is the only way to determine exactly how much time the Sun and Hypercube portions will require. By adjusting the number of cubes (1 or 2) and the nodes per cube (2, 4, 8, *etc.*) the configuration in which the Hypercube and Sun "sleep" the least can be discovered. The Hypercube script sleeps when there is no hfile to process. The Sun script sleeps when there are too many. The first line of **scanit** defines the number of hfiles that will cause it to sleep. In the most time-efficient configuation, the Sun will not sleep at all.

Once the limiting factor of the time per tune point on the Sun is found, splitting the range of tunes to be explored into pieces allows the project to be completed at convenient times.

Restarting

If a run only partially completes, simply edit the tunefile, removing the tunes for which a linear aperture is listed in the output file, and if required, add other unexplored tunes.

In case the run crashes and hfiles exist in the hfiles directory, those files can be listed in .sunscanned_b. The lines in **scanit** that destroy .sunscanned_b must be commented out. Upon execution of **tunescan**, hyperbat should begin processing the first file listed. If there are hfiles but no tunes left to explore, simply comment out **scanit** inside **tunescan**.

If the cube portion crashes and the Sun is still running or sleeping, **hyperbat** (and **hypercat**) can be started from a Hypercube prompt. First release the cube then execute hyperbat:

sycamore% hyperbat <nodes> <vectrack input> <cubename> &

To remain consistent, name the cube for the machine on which **scanit** is running. A _bat or _cat will automatically be added onto the end.

If a complete restart is required, remaining parts of the **tunescan** job must be killed on both the Sun and Sycamore. The previously used cube(s) must be released. **HINT**: Once the Sun portion is dead,

echo "Done" >> .sunscanned_b

will cause **hyperbat** to cease running and release its cube. Echoing "Done" into .sunscanned_c will cause **hypercat** to end. It will not happen immediatly if the cube script is asleep; this is simply easier than killing jobs one by one from a Hypercube prompt.

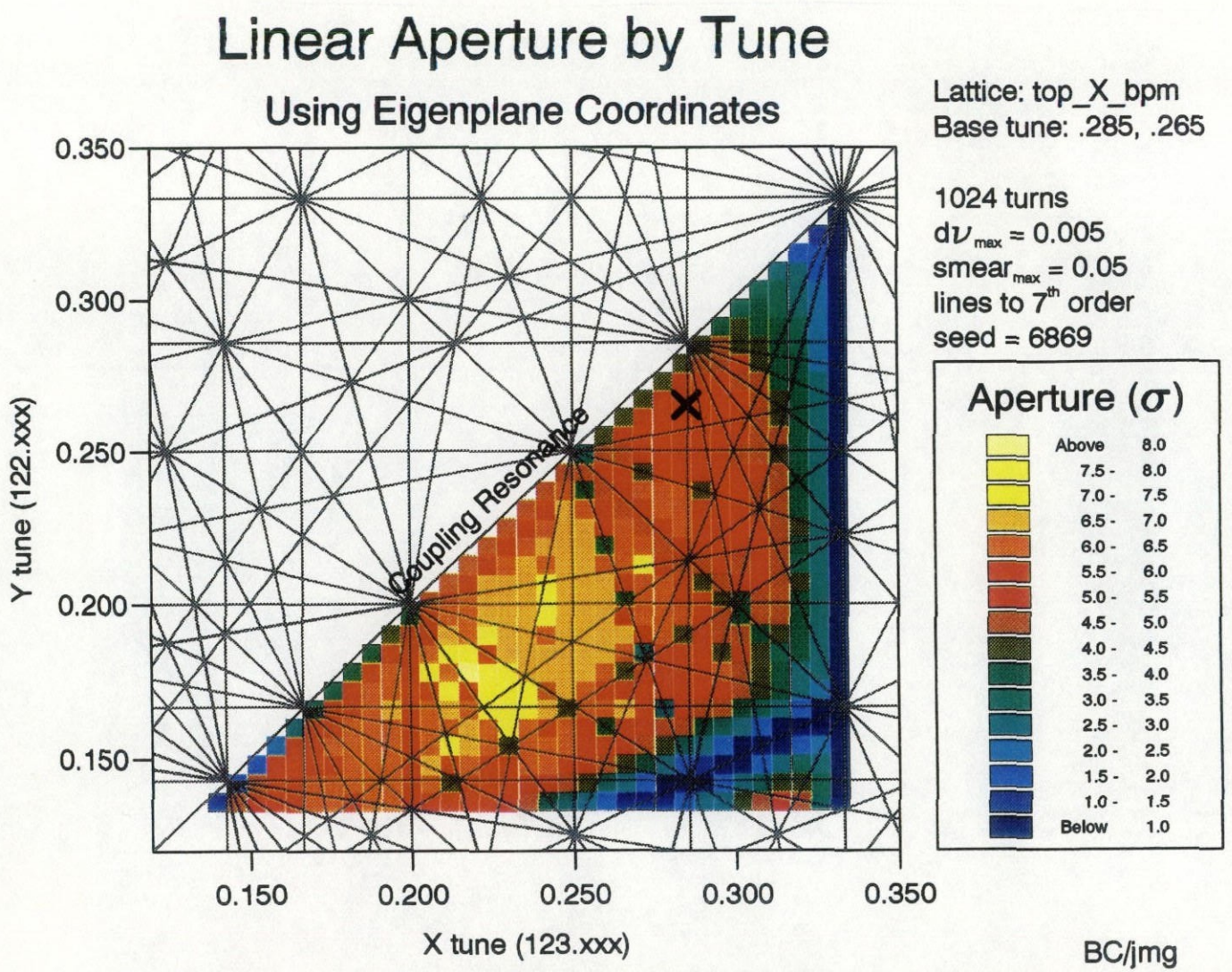

Figure 1. Tune Plane in the Neighborhood of the (123.285, 122.265) Operating Point.

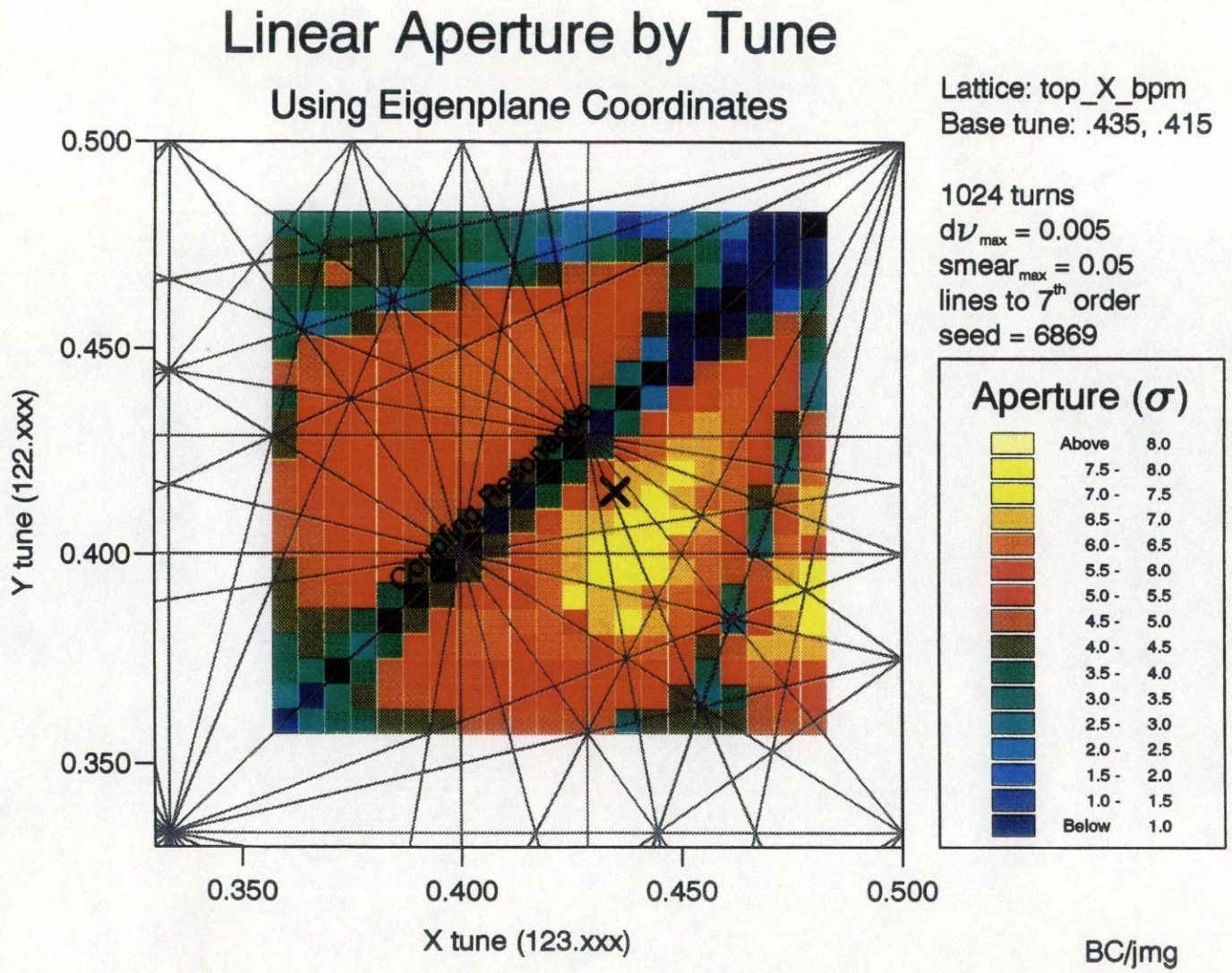

Figure 2. Tune Plane in the Neighborhood of the (123.435, 122.415) Operating Point.

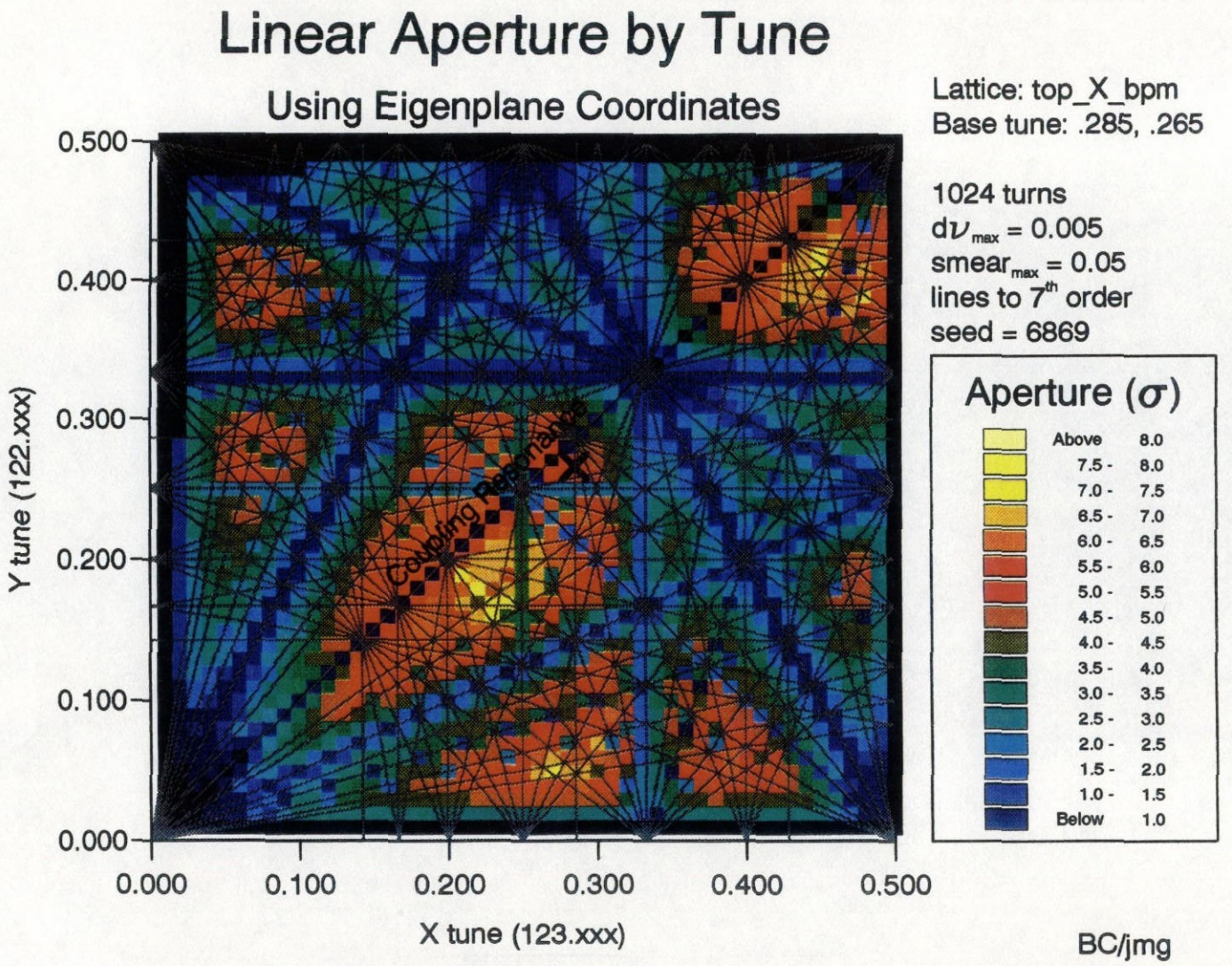

Figure 3. Coarse Scan of Entire Tune Plane.

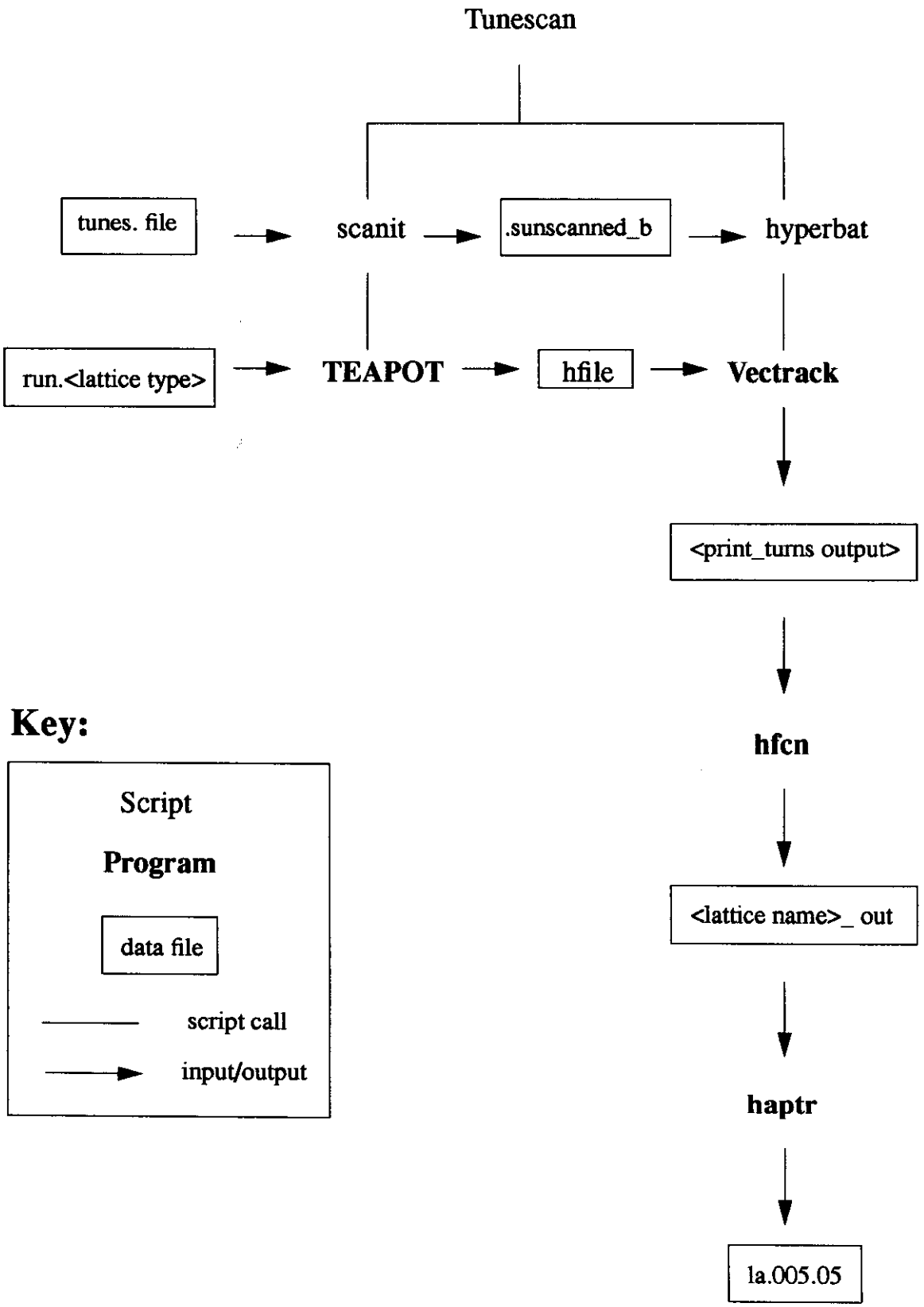

Figure 4. Tunescan Process.