Tune Scan Program

David Sagan
November 1, 2023

Contents

1 Introduction

A *tune scan* is a way of examining resonances which involves plotting a resonance related beam parameter such as the beam size or the beam lifetime as a function of the transverse Q_a and Q_b tunes. A scan can be done in an actual machine or in simulation which is what *tune_scan* program documented here does. Since lifetimes can be tricky and time intensive to calculate, the *tune_scan* program uses variations in the amplitude of a tracked particle as the indicator for resonance.

The *tune_scan* program is built atop the Bmad software toolkit [1]. The Bmad toolkit is a library, developed at Cornell, for the modeling relativistic charged particles in storage rings and Linacs, as well as modeling photons in x-ray beam lines.

The *tune_scan* program comes with the "Bmad Distribution" which is a package which contains Bmad along with a number of Bmad based programs. See the Bmad web site for more details.

The *tune_scan* program comes in two versions:

```
tune_scan       ! Single threaded version.
tune_scan_mpi   ! MPI threaded version.
```

The first version is a single threaded version and the second is a multi-threaded version using MPI.

[Note to Bmad Distribution maintainers: The single threaded version is built by default when compiling a Bmad Distribution. The MPI version can be built by setting in the *dist_prefs* file:

```
  export ACC_ENABLE_MPI=Y
```

and then using the "*mk*" command in the *bsim* directory to build the executable. See the documentation on the Bmad web site for more details.]

2 Running the Tune Scan Program

See the documentation for setting up the Bmad environment variables at

```
https://wiki.classe.cornell.edu/ACC/ACL/RunningPrograms
```

Once the Bmad environment variables have been set, the syntax for invoking the single threaded version of the *tune_scan* program is:

```
tune_scan {<master_input_file_name>}
```

Example:

```
tune_scan my_input_file.init
```

The *<master_input_file_name>* optional argument is used to set the master input file name. The default value is "*tune_scan.init*". The syntax of the master input file is explained in §5.

Example input files are in the directory (relative to the root of a Bmad Distribution):

```
bsim/tune_scan/example
```

When running the MPI version threaded over multiple computers, the details of how to start the process will vary depending upon the installation. See your local Guru for details. When running on a single machine, the typical command will look like

```
mpiexec -n <num_processes> tune_scan_mpi {<master_input_file_name>}
```

where *<num_processes>* is the number of processes. *miprun* is the typical alternative if *mpiexec* is not defined.

3 Nomenclature

A particle has three modes of oscillation which are labeled a, b, and z. The a-mode is the "horizontal-like" mode which, if there is no coupling, will correspond to purely horizontal motion.

The b mode is the "vertical-like" mode which, if there is no coupling, will correspond to purely vertical motion. The *Linear Optics* chapter of the Bmad manual has more details.

4 Fortran Namelist Input

Fortran namelist syntax is used for parameter input in the master input file. The general form of a namelist is

```
&<namelist_name>
  <var1> = ...
  <var2> = ...
  ...
/
```

The tag *"&<namelist_name>"* starts the namelist where *<namelist_name>* is the name of the namelist. The namelist ends with the slash *"/"* tag. Anything outside of this is ignored. Within the namelist, anything after an exclamation mark *"!"* is ignored including the exclamation mark. *<var1>*, *<var2>*, etc. are variable names. Example:

```
&place
  section = 0.0, "arc_std", "elliptical", 0.045, 0.025
/
```

here *place* is the namelist name and *section* is a variable name. Notice that here *section* is a "structure" which has five components – a real number, followed by two strings, followed by two real numbers.

Everything is case insensitive except for quoted strings.

Logical values are specified by *True* or *False* or can be abbreviated *T* or *F*. Avoid using the dots (periods) that one needs in Fortran code.

5 Master Input File

The *master input file* holds the parameters needed for running the tune scan program. The master input file must contain a single namelist (§4) named *params*. Example:

```
&params
  ts%lat_file = "$lat"     ! Bmad lattice file.
  ts%dat_out_file = ""     ! Data output file name. Default is "tune_scan.dat"

  ts%Q_a0 = 0.590          ! Q_a range is [Q_a0, Q_a1].
  ts%Q_a1 = 0.600          ! Units are rad/2pi.
  ts%dQ_a = 0.001          ! Q_a step size.

  ts%Q_b0 = 0.675          ! Q_b range is [Q_b0, Q_b1].
```

```
  ts%Q_b1 = 0.676          ! Units are rad/2pi.
  ts%dQ_b = 0.001          ! Q_b step size.

  ts%use_phase_trombone = F ! Use phase trombone ele to adjust transverse tunes?
  ts%quad_mask = ""         ! Quadrupole mask used with quadrupole variation.
  ts%group_knobs = "", ""   ! Tune variation can optionally be done using group elements.

  ts%Q_z0 = 0.066          ! Q_z range (when RF is on) is [Q_z0, Q_z1].
  ts%Q_z1 = 0.067          ! Units are rad/2pi.
  ts%dQ_z = 0.001          ! Q_z step size.

  ts%pz0  = -0.01          ! pz range (when RF is off) is [pz0, pz1].
  ts%pz1  =  0.01
  ts%dpz  =  0.005         ! pz setp size.

  ts%a_emit = 0            ! a-mode emit (overrides lattice calc)
  ts%b_emit = 0            ! b-mode emit (overrides lattice calc)
  ts%sig_pz = 0            ! z-mode emit (overrides lattice calc)
  ts%a0_amp = 1            ! Init a-mode amplitude relative to the beam sigma.
  ts%b0_amp = 1            ! Init b-mode amplitude relative to the beam sigma.
  ts%pz0_amp = 1           ! z-mode amplitude.

  ts%rf_on = F             ! RF acceleration field on or off?
  ts%n_turn = 2000         ! Number of turns to track a particle.
  ts%timer_print_dtime = 120  ! Time (sec) between progress printouts.

  bmad_com%radiation_damping_on = F        ! Set radiation damping on/off.
  bmad_com%radiation_fluctuations_on = F   ! Set radiation excitation on/off.
/
```

ts%lat_file
Name of the Bmad lattice file to use. This name is required.

ts%dat_out_file
Data output file name. Default if blank or not present is *tune_scan.dat*.

ts%group_knobs
If *ts%group_knobs* is set, tune variation can be done by variation of two *group* elements whose names are given by *ts%group_knobs*. Example:

```
  ts%group_knobs = "qtune1", "qtune2"
```

ts%Q_a0, ts%Q_a1, ts%dQ_a
[ts%Q_a0, ts%Q_a1] is the range of the tune plane grid along the Q_a axis and *ts%dQ_a* is the spacing between grid points. Tunes are in units of rad/2pi and only the fractional part should be specified.

ts%Q_b0, ts%Q_b1, ts%dQ_b
[ts%Q_b0, ts%Q_b1] is the range of the tune plane grid along the Q_b axis and *ts%dQ_b* is

the spacing between grid points. Tunes are in units of rad/2pi and only the fractional part should be specified.

ts%use_phase_trombone
Use a phase trombone element to vary the lattice (Q_x, Q_y) tunes from point to point (§6)? Default is False. Also see *ts%quad_mask*.

ts%quad_mask
When *not* using a phase trombone element (see *ts%use_phase_trombone*), it is sometimes desired not have certain quadrupole strengths varied. This can be achieved by setting *ts%quad_mask* which is a list of quadrupoles to be excluded from variation. See §6 for more details.

ts%Q_z0, ts%Q_z1, ts%dQ_z
Only used when *ts%rf_on* is set to True. In the RF on mode, the longitudinal tune Q_z is set to a certain value and a tune plane scan is made. Q_z is then changed and a new tune plane scan is made, etc. The range of values for Q_z is set by *[ts%Q_z0, ts%Q_z1]* with a spacing of *ts%dQ_z* between points.

ts%pz0, ts%pz1, ts%dpz
Only used when *ts%rf_on* is set to False. In the RF off mode, the phase space p_z of the tracked particle is set to a certain value and a tune plane scan is made. p_z is then changed and a new tune plane scan is made, etc. The range of values for p_z is set by *[ts%p_z0, ts%p_z1]* with a spacing of *ts%dp_z* between points.

ts%a_emit, ts%b_emit
Emittances of the a and b modes. These are used to calculate the particle starting position when tracking (§6). If not present or set to zero, the emittances as calculated from the lattice are used.

ts%sig_pz
Sigma of phase space p_z coordinate. This is used to calculate the particle starting position when tracking (§6). If not present or set to zero, the value as calculated from the lattice is used. Also see *ts%pz0_amp*

ts%a0_amp, ts%b0_amp
particle starting position a and b-mode amplitudes relative to the equilibrium beam size. The default is zero.

ts%pz0_amp
Particle initial phase space z-mode amplitude relative to the equilibrium longitudinal beam size. The default is zero.

ts%rf_on
Determines if the RF is turned on or off. If *False* (the default), the RF is off and if *True* the RF is on.

ts%n_turn
Number of turns to track a particle.

ts%timer_print_dtime
Time (seconds) between progress printouts.

bmad_com
Bmad common structure used for setting Bmad global parameters. See the Bmad manual for more details. For the *tune_scan* program, the two most important parameters are the ones that control whether the radiation damping or excitation are on or off.

6 Simulation

Summary: A grid of points is defined in the (Q_a, Q_b) tune plane. For each point, lattice parameters are adjusted so that the lattice tunes are equal to the tunes defined by the tune plane point. After this, a particle is tracked and the RMS and maximum amplitudes for each mode of oscillation are recorded in the output file. This is called a tune plane "*data set*". Multiple data sets may be calculated at a set of longitudinal tunes (if the RF is on) or at a set of particle p_z values (if the RF is off).

The (Q_a, Q_b) tune plane grid is defined by setting in the master input file§5 the Q_a and Q_b ranges which are *[ts%Q_a0, ts%Q_a1]* and *[ts%Q_b0, ts%Q_b1]* respectively along with the spacing between points given by *ts%dQ_a* and *ts%dQ_b*.[1] The units are in radians/2π. Only the fractional part of the tune should be given.

The transverse (Q_x, Q_y) tunes can be veried in one of three ways:

If *ts%use_phase_trombone* is set to True, a *match* element in phase trombone mode is added to the lattice and the phase advance in the element is adjusted as needed. This gives a "smooth" tune variation in the sense that the Twiss parameters do not change around the ring. It is somewhat unrealistic though when trying to simulate an actual machine.

If *ts%group_knobs* (which is a vector of two strings) is set to non-blank values, the group elements in the lattice matching the names will be used to vary the tune.

```
ts%group_knobs = "qtune1", "qtune2"
```

If none of the above methods are used, the lattice tune variation is achieved by varying quadrupole strengths. The general algorithm is to exclude any quadrupoles that are tilted and then divide the upright quadrupoles into two groups. One group were $\beta_a > \beta_b$ at the quadrupole and the other group with $\beta_a < \beta_b$. The k_1 quadrupole strength of the quadrupoles of each group are varied in unison. The two groups represent two variables which can be used to match the two tune conditions. It may be desirable to not vary selected quadrupoles. For example, the quadrupoles in an interaction region. To exclude quadrupoles from being varied, the *ts%quad_mask* string can be set appropriately. See the Bmad manual for details about element selection. For example:

[1] If the interval size *ts%Q_a1 — ts%Q_a0* is not commensurate with the spacing *ts%dQ_a*, the actual grid upper bound is taken to be *ts%Q_a0 + na * ts%dQ_a* where *na* is a integer such that the actual upper bound is as close to *ts%Q_a1* as possible. A similar calculation is done with the b axis.

```
ts%quad_mask = "QT* IP:M34"
```

In this example, all quadrupole elements whose name begins with "*QT*" along with all quadrupoles in between elements *IP* and *M34* are excluded from variation. Example:

```
ts%quad_mask = "* ~qt%"
```

In this example all quadrupole elements are vetoed except for quadrupoles whose name begins with "*QT*" and has three letters. That is, only quadrupoles whose name begins with "*QT*" and has three letters will be varied.

There are two modes of operation. In the *RF off* mode, *ts%rf_on* is set to False (the default). In the *RF on* mode, *ts%rf_on* is set to True. In the *RF off* mode, tune plane data sets are calculated at a set of particle p_z values. These values are in the range *[ts%pz0, ts%pz1]* with a spacing between points given by *ts%dpz*. Generally this mode is only used with radiation damping and excitation turned off. In the *RF on* mode, tune plane data sets are calculated at a set of longitudinal tune values in the range *[ts%Q_z0, ts%Q_z1]* with the spacing between points given by *ts%dQ_z*. The longitudinal tune is adjusted by varying the RF voltage at the RF cavities.

At each tune plane point, a particle is tracked for *ts%n_turn* turns and the variation of the amplitude is recorded. Without coupling or dispersion, the initial phase space position of the particle is set to:

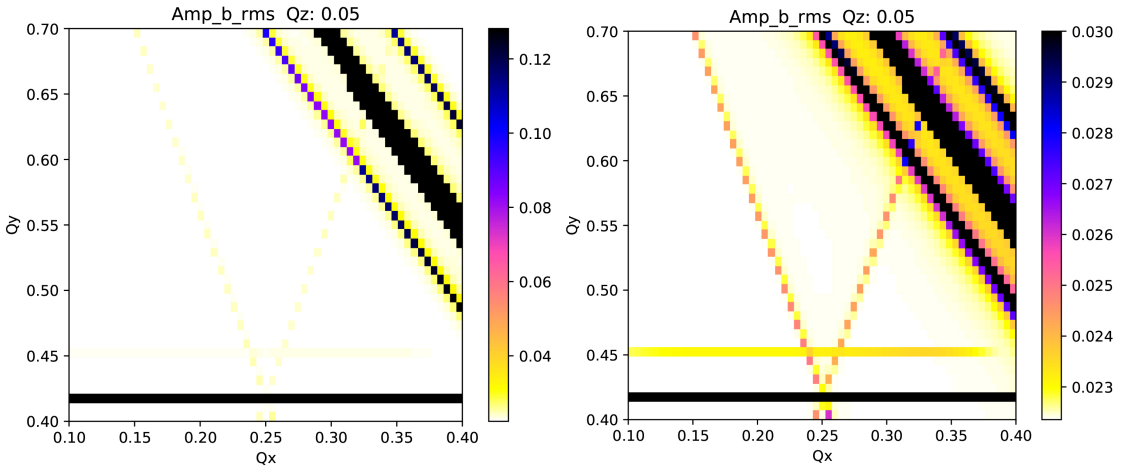
$$\begin{aligned}(x, p_x, y, p_y, z, p_z) &= (A_a\,\sigma_a, 0, A_b\,\sigma_b, 0, p_{z0} + n\,dp_z) \qquad \text{! RF off} \\ &= (A_a\,\sigma_a, 0, A_b\,\sigma_b, 0, A_z\,\sigma_{pz}) \qquad \text{! RF on}\end{aligned} \tag{1}$$

Amplitudes A_a, A_b, and A_z are set by *ts%a0_amp*, *ts%b0_amp* and *ts%z0_amp* respectively. The σ_a and σ_b beam sizes are calculated using the equilibrium emittances ϵ_a and ϵ_b and the local Twiss parameters. ϵ_a and ϵ_b may be set by setting *ts%a_emit* and *ts%b_emit* respectively. If not set, the calculated emittances are used. With RF off, the initial p_z coordinate is set to $p_{z0} + n\,dp_z$ where n is an integer, p_{z0} is set by *ts%pz0*, and dp_z is set by *ts%dpz*. With RF on, A_z is set by *ts%pz0_amp* and σ_{pz} is set by *ts%sig_pz*. If *ts%sig_pz* is not set, the equalibrium value as calculated from the lattice is used. When there is coupling or dispersion, the initial phase space position is shifted to keep the initial amplitudes of oscillation of the three modes equal to what one would have without coupling and dispersion.

7 Output

The output data file is a table of values with each row being data from one tune plane grid point. The thirteen columns are:

```
ja, jb, jz      ! Index of the tune plane grid point
Q_a, Q_b, Q_z   ! Tunes
```



(a) 2D density plot of the amp_b_rms data.

(b) 2D density plot like 1a with the range maximum reduced to 0.03.

Figure 1

```
data_turns      ! Number of turns over which data is taken
amp_a_rms, amp_b_rms, amp_z_rms
                ! RMS of the amplitude variation for the three modes.
amp_a_max, amp_b_max, amp_z_max
                ! Max amplitude for the three modes.
```

Amp_a, *amp_b* and *amp_z* are dimensionless normal mode amplitudes normalized by the normal mode emittances.

ja, jb, jz
Indexes *ja*, *jb*, and *jz* start from zero which corresponds to tunes *ts%Q_a0*, *ts%Q_b0*, and *ts%Q_z0* respectively.

data_turns
Data_turns is the number of turns over which data is taken. This is generally equal to *ts%n_turn* unless the particle hits an aperture or it is not possible to calculate Twiss parameters due to the lattice being unstable. In the latter case, *Data_turns* will be set to zero and the RMS and max amplitudes will be set to -1.

amp_a_rms, amp_b_rms, amp_z_rms RMS of the amplitude variation turn-to-turn relative to the equilibrium amplitude.

amp_a_max, amp_b_max, amp_z_max Maximum of the amplitude variation turn-to-turn relative to the equilibrium amplitude.

8 Plotting

There is a script named *tune_scan_density_plot.py* for 2D density plotting of a tune plane data set using Python. This script is in the directory (relative to the root of a Bmad Distribution)

```
bsim/tune_scan/plotting
```

Example plots are in Fig. 1.

The syntax for running the script is

```
tune_plane_density_plot.py {-cmap <color_map>} {-column <col_to_plot>}
            {-min <min_val>} {-max <max_val>} {-z <z_index>} {<data_file_name>}
```

Example:

```
det_pix_plot.py -column 9 -max 0.05 -z 1 this_run.dat
```

<color_map>
A *color map* maps data values to the colors used in plotting. The underlying plotting package is called *matplotlib* and this package has a number of color maps which may be used. The default is *gnuplot2_r*. For a list of possible color maps, google "matplotlib color maps".

<col_to_plot>
Setting the column index *<col_to_plot>* sets the column of data (§7) which is plotted. Columns are numbered starting from zero so the column index of the *amp_a_rms* data is column 7 and the *amp_z_max* data is column 12.

<min_val>, <max_val>
minimum and maximum values for the data range to map onto the color map. Default values are the minimum and maximum data values. Any datum values lower than the range minimum are set to the range minimum and any datum values greater than the range maximum are set to the range maximum. This is useful to emphasize features. For example, Fig. 1b is similar to Fig. 1a except the range maximum has been reduced to 0.03. This better shows the weaker resonance lines.

<data_file_name>
The name of the data file. Default is *tune_scan.dat*.

References

[1] D. Sagan, "Bmad: A Relativistic Charged Particle Simulation Library" Nuc. Instrum. & Methods Phys. Res. A, **558**, pp 356-59 (2006).