lmsuite User's Manual
for version 0.92

August 6, 2003

Contents

1 Introduction

LATTE/MUSE Numerical Suite (*lmsuite*) is a Traveling Wave Tube (TWT) code developed as a part of my Ph.D. thesis research [7]. The code can solve three different 1-d nonlinear multifrequency TWT models: LATTE, MUSE, and S-MUSE. LATTE (Lagrangian TWT Equations) [10] is a "large-signal" code of the variety originally developed for single frequency by Nordsieck [5] and later extended to multifrequency by El-Shandwily [3] and Giarola [4]. The MUSE (multifrequency spectral Eulerian) and S-MUSE (simplified-MUSE) models [10] are nonlinear models derived for their unique spectral characteristics. Generally MUSE and S-MUSE are not accurate in saturation due to their inability to predict electron overtaking effects, however due to their spectral structure they are very useful for gaining insights into nonlinear TWT physics [10, 11, 8, 9, 7]. Together the three models comprise a "suite" hopefully useful for TWT education, research, and design.

2 Acknowledgements

Thanks to Mark Converse for comments on structuring the input deck and being the first beta tester. Thanks to Michel Olagnon for helping restructure the LATTE portion of the code to run faster.

Thanks to my research advisors Professor John Booske and Professor Ian Dobson for their advice and encouragement. Thanks to the Innovative Microwave Vacuum Electronics MURI and AFOSR for funding support.

3 Physics

THIS SECTION IS IN PROGRESS. For general model derivations see [10]. Details of code implementations including losses, normalizations, and slowly varying envelope approximations are in [7] and will appear here soon.

- geometry
- models

3.1 Sheath helix model

The sheath helix model currently implemented is the simplest one available [6, pg. ??39–41??]. It does not account for a conducting barrel and should only

be used for qualitative purposes. The equations computed are the cold circuit phase velocity

$$v_{\rm ph} = \frac{c}{\cot\psi}\left[\frac{I_0(\gamma r_{\rm h})K_0(\gamma r_{\rm h})}{I_1(\gamma r_{\rm h})K_1(\gamma r_{\rm h})}\right]^{1/2}, \quad (1)$$

and the cold circuit interaction impedance

$$K = \sqrt{\frac{\mu_0}{\epsilon_0}}\frac{\cot\psi}{2\pi}\left[I_0(\gamma r_{\rm h})K_0(\gamma r_{\rm h})I_1(\gamma r_{\rm h})K_1(\gamma r_{\rm h})\right]^{1/2}. \quad (2)$$

In (1) and (2) we have

$$\gamma^2 = k^2\cot^2\psi\frac{I_1(\gamma r_{\rm h})K_1(\gamma r_{\rm h})}{I_0(\gamma r_{\rm h})K_0(\gamma r_{\rm h})} \quad (3)$$

$$k^2 = \omega^2\mu_0\epsilon_0 \quad (4)$$

$$\cot\psi = \tfrac{2\pi r_{\rm h}}{z_{\rm h}}, \quad (5)$$

where $r_{\rm h}$ is the helix radius, $z_{\rm h}$ is the turn to turn spacing (entered into the code as "helix pitch"), and μ_0, ϵ_0 and c are the permeability, permittivity, and the speed of light of free space respectively.

3.2 Space charge reduction factor

The formula for computing space charge reduction factor is taken from [2]. Using the Bessel function identity

$$\frac{d}{dz}K_0(z) = -K_1(z) \quad (6)$$

we have for the factor $R_{\rm sc}$

$$R_{\rm sc} = 1 + \frac{2}{r_{\rm bo}^2 - r_{\rm bi}^2}\left\{\left[r_{\rm bo}I_1(\kappa r_{\rm bo}) - r_{\rm bi}I_1(\kappa r_{\rm bi})\right]\left[\frac{K_0(\kappa r_{\rm h})}{I_0(\kappa r_{\rm h})}\left(r_{\rm bi}I_1(\kappa r_{\rm bi}) - r_{\rm bo}I_1(\kappa r_{\rm bo})\right)\right.\right.$$
$$\left.\left. - r_{\rm bo}K_1(\kappa r_{\rm bo})\right] - r_{\rm bi}I_1(\kappa r_{\rm bi})\left[r_{\rm bi}K_1(\kappa r_{\rm bi}) - r_{\rm bo}K_1(\kappa r_{\rm bo})\right]\right\} \quad (7)$$

where $r_{\rm bo}$ is the outer beam radius, $r_{\rm bi}$ is the inner beam radius, $r_{\rm h}$ is the helix radius, and

$$\kappa^2 = \beta_{e,c}^2 - \frac{\omega^2}{c^2}. \quad (8)$$

For the term $\beta_{e,c}$ one can use either the electron stream wavenumber β_e

$$\beta_e = \frac{\omega}{u_0} \quad (9)$$

where u_0 is the dc beam velocity, or the circuit wavenumber β_c

$$\beta_c = \frac{\omega}{\tilde{v}_{\mathrm{ph}\ell}} \tag{10}$$

where $\tilde{v}_{\mathrm{ph}\ell}$ is the frequency dependent cold circuit phase velocity. To use β_c one sets `use_beta_c = true` in the `&dispersion` namelist. However, when `use_beta_c = true`, β_e is still used for frequencies that do not have a phase velocity specified.

4 Running *lmsuite*

To download executables and source code for *lmsuite* go to `http://www.lmsuite.org` and follow the instructions provided.

To run *lmsuite*:

1. open a command shell (Windows, Linux, or Unix)
2. change to the `LMSuitev0.92` directory
3. type `lmsuite` at the shell prompt.

The program reads files in the directory `./LMSuitev0.92/inputs/` to determine what is to be computed. Possible files in `./inputs/` are `lmsuite.nml` (main input deck), `scan_data.nml` (optional input deck containing scan information), `movies.nml` (optional input deck containing information to generate "frame" files used for animation), `circuit.in` (optional file defining TWT circuit), `frequencies.in` (optional file specifying input frequencies, input powers, and input phases), `dispersion.in` (optional file specifying TWT dispersion properties), and `losses.in` (optional file specifying circuit losses). For details on using these files see Section 4.3. The output data is written to the directory `./LMSuitev0.92/outputs/`. See Section 4.3.4 for a description of the sample namelists included with the distribution.

Generally one can run the code in either "single pass mode" or "scan mode" as described below.

4.1 Single pass mode

The single pass mode solves the TWT equations for a set of fixed input parameters. Single pass mode is chosen by setting `num_scan_namelists` = 0 in the `&run` namelist.

Available plotting routines in single pass mode are:

- `power_out_vs_freq`: Circuit power at TWT output versus frequency for all circuit frequencies. Output data file: `pow_vs_freq.X.sp.dat`.
- `phase_vs_freq`: Voltage phase at TWT output versus frequency for all circuit frequencies. Output data file: `phase_vs_freq.X.sp.dat`.

- `circuit_power_vs_z`: Circuit power versus axial distance z for all circuit frequencies. Output data file: `pow_vs_z.X.sp.dat`.

- `magnitude_vs_z`: Magnitude of circuit voltage, circuit current, space charge field, beam velocity, or beam density versus axial distance z for all circuit or space charge frequencies (depending on whether the variable is a circuit variable or a beam variable). Variable chosen by integer $1-6$ respectively. See `&output_data` namelist for details. Output data file: `mag_vs_z.X.sp.dat`.

- `phase_vs_z`: Phase of circuit voltage, circuit current, space charge field, beam velocity, or beam density versus axial distance z for all circuit or space charge frequencies (depending on whether the variable is a circuit variable or a beam variable). Variable chosen by integer $1-6$ respectively. See `&output_data` namelist for details. Output data file: `phase_vs_z.X.sp.dat`.

- `phase_differences_vs_z`: Phase difference between two of: circuit voltage, circuit current, space charge field, beam velocity, or beam density versus axial distance z for all circuit or space charge frequencies (depending on whether the variables are circuit variables or a beam variables). Variables chosen by integers $1-6$. See `&output_data` namelist for details. Output data file: `phase_diff_vs_z.X.sp.dat`.

- `disk_orbits_vs_z`: not implemented

- `conserved_quantity_vs_z`: not implemented

- `beam_energy_vs_z`: not implemented

- `dc_beam_vel_vs_z`: The average electron beam velocity as as function of axial distance z. Output data file: `dcbeam_vs_z.X.sp.dat`.

- `phase_space` (LATTE only): Create particle phase space plots at a specified number of times. Output data file: `phase_space.M.dat` where M corresponds to the location of the `time_array`.

In the above file names `X` is `L, M,` or `S` depending on whether LATTE, MUSE or S-MUSE generates the data. The tag `.sp.` in the file name represents "single pass."

4.2 Scan mode

In contrast to the single pass mode one can have a simulation scan various input parameters, solving the TWT equations for each set of input parameters.

4.2.1 Single parameter scans

For a single parameter scan a single input parameter is scanned over a specified range. Each single parameter scan adds 1 to the value of `num_scan_namelists` in the `&run` namelist. The scan has its own `&scan` namelist in the file `scan_data.nml` and the type is specified by the `scanID`. Full details for specifying scans are given in Section 4.3.2. The types of single parameter scans and the output each can provide are:

1. Input power scan (`scanID` = 1): For a chosen frequency the input power is varied over a range. This scan is used for making the widely used AM/AM and AM/PM curves.
   Available plotting routines in the input power scan are:

   - `power_out_vs_power_in`: Output power versus input power at either the frequency specified in the scan or all frequencies. Output data file: `pout_vs_pin.X.N.dat`.
   - `gain_vs_power_in`: Gain versus input power at either the frequency specified in the scan or all frequencies. Output data file: `gain_vs_pin.X.N.dat`.
   - `phase_out_vs_power_in`: Output voltage phase versus input power at either the frequency specified in the scan or all frequencies. Output data file: `phase_vs_pin.X.N.dat`.
   - All of the `*_vs_z` plots (see Section 4.1 for a listing). For each of the input power values, a full set of `*_vs_z` curves are given.

2. Input phase scan (`scanID` = 2): For a chosen frequency the input phase is varied over a range. This scan is useful for studying harmonic/intermodulation injection where getting the critical input phase is neccessary to obtain reduction of the harmonic/intermodulation outputs.
   Available plotting routines in the input power scan are:

   - `power_out_vs_phase_in`: Output power versus input phase at the frequency specified in the scan. Output data file: `pout_vs_phasein.X.N.dat`.
   - All of the `*_vs_z` plots (see Section 4.1 for a listing). For each of the input phase values, a full set of `*_vs_z` curves are given.

3. Frequency scan (`scanID` = 3): For a chosen input `frequency_integer` *array location* (namelist `&frequency_list`), scan the frequency over a range of values. That is, scan the value of frequency for a particular array element in the frequency array `frequency_integer`.
   Available plotting routines in the frequency scan are:

   - `power_out_vs_freq`: Output power at the *scanned* frequency versus the scanned frequency OR output power at a *different frequency* versus the scanned frequency. Output data file: `pout_vs_freq.X.N.dat`.
   - `gain_vs_freq`: Gain at the *scanned* frequency versus the scanned frequency. Output data file: `gain_vs_freq.X.N.dat`.

- `phase_vs_freq`: Output voltage phase at the *scanned* frequency versus the scanned frequency OR output voltage phase at a *different frequency* versus the scanned frequency. Output data file: `pout_vs_freq.X.N.dat`.

4. Dispersion parameter scan (`scanID` = 4): At a particular frequency scan phase velocity, interaction impedance, loss, or space charge reduction factor.
   Available plotting routines in the dispersion parameter scan are:

   - All of the `*_vs_z` plots (see Section 4.1 for a listing). For each of the dispersion parameter values, a full set of `*_vs_z` curves are given.

5. Beam parameter scan: not implemented

In the above file names `X` is `L, M,` or `S` depending on whether LATTE, MUSE or S-MUSE generates the data, and `N` is an integer corresponding to the scan number, i.e., the order that the scan appears in the file `scan_data.nml`.

In principle one can run as many scans as desired. Presently the maximum number is 10 (set by `max_scans` in the source file `parameters.f90`), but this can easily be changed for future versions.

4.2.2 Two parameter scans

A two parameter scan connects two `&scan` namelists so that two different parameters are scanned simultaneously. Two `&scan` namelists must be provided for each two parameter scan desired. The first of these two namelists must have `two_parameter = true`, where the second namelist must have `two_parameter = false`. Correspondingly a count of 2 must be added to `num_scan_namelists` in the `&run` namelist in `lmsuite.nml`.

As a general rule two parameter scans provide data that will generate parameterized curves. The first scan (i.e. the scan with `two_parameter = true`) should be over the variable that you wish to have as your independent coordinate axis in graphs, and the second scan should be over the variable that will parameterize the curves.

Not all scan types (i.e. `scanID`s) may be used together. The allowable combinations of `scanID`s are given below. Note that the ordering of the scans in the list below must be the same ordering entered into the namelist.

- Input power scan (`scanID = 1`) and dispersion parameter scan (`scanID = 4`). Available plotting routines for this two parameter scan are `power_out_vs_power_in`, `gain_vs_power_in`, and `phase_out_vs_power_in`.

4.3 Namelists

4.3.1 `lmsuite.nml`

NAMELIST DESCRIPTIONS ARE CURRENTLY GIVEN ONLY IN THE PROVIDED NAMELIST FILES. AS SOON AS THE NAMELISTS SEEM TO

HAVE MATURED, THE INFORMATION WILL BE TRANSFERRED TO THIS SECTION.

4.3.2 `scan_data.nml`

NAMELIST DESCRIPTIONS ARE CURRENTLY GIVEN ONLY IN THE PROVIDED NAMELIST FILES. AS SOON AS THE NAMELISTS SEEM TO HAVE MATURED, THE INFORMATION WILL BE TRANSFERRED TO THIS SECTION.

4.3.3 `movies.nml`

NAMELIST DESCRIPTIONS ARE CURRENTLY GIVEN ONLY IN THE PROVIDED NAMELIST FILES. AS SOON AS THE NAMELISTS SEEM TO HAVE MATURED, THE INFORMATION WILL BE TRANSFERRED TO THIS SECTION.

4.3.4 The sample namelists

The namelists (`*.nml`) and `*.in` files provided with the code are based on the TWT circuit found in [1]. The default run is a single pass of the slowly varying envelope version of LATTE for 1.538 GHz. A scan namelist is also provided to create a figure similar to Fig. 3 of [1]. The circuit and beam parameters are only approximate, so we do not expect to be able to reproduce exactly the results of [1]. Comments are provided below on certain aspects of the input files to help the new user get started.

- `&user_interface`
  - This namelist is presently set to maximum verboseness. Setting all of these flags to false forces the code to run in "silent mode."
  - Once one is comfortable that they are entering the proper data, they can set `echo_namelists = false`. However, when preparing a new namelist this flag can be *very* useful in determining whether you are entering the data that you intend to. Setting `echo_namelists = true` should always be your first recourse if the code is not doing what you think it should do.
- `&run`
  - Since the TWT circuit has a substantial sever (see Fig. 1 of [1]) one has to use the slowly varying envelope version of the models (`svea = true`) due to reflection issues. Since only LATTE presently has the slowly varying envelope formulation available, we choose only to run LATTE (`select_code = 'L'`).
  - For single pass mode one has `num_scan_namelists = 0`. To run the input power scan change `num_scan_namelists = 1`. The input power

scan produces data that when graphed resemble Fig. 3(a) of [1]. Changing `read_from_file = true` in `&frequency_list` and changing `int_param_1 = 1750` in the `&scan` namelist should approximate Fig. 3(b) of [1].

- `&output_data`
  - Since `plot_dispersion = true` files will be generated in `./outputs/` containing the circuit information. Graph the contents of these files to see how we have approximated the circuit of Fig. 1 in [1]. `vph.dat` contains phase velocity data, `impedance.dat` has interaction impedance data, `loss.dat` contains loss information, and `scrf.dat` contains space charge reduction factor data.
- `&frequency_list`
  - To be able to resolve frequencies down to 1.0 MHz we set `base_frequency`= 1.0e6 Hz. Therefore for 1.538 GHz the frequency integer is 1538. Changing `read_from_file = true` changes the frequency to 1.75 GHz since this is the frequency specified in `frequencies.in`.
  - We set `highest_order_IMP = 2` so that the second harmonic is computed. One could include the 3^{rd} harmonic *in the beam* by changing `highest_order_IMP = 3` and increasing the `max_space_charge_freq` to 3900. However, unless additional dispersion data is supplied the `max_ckt_freq` must not be increased.
- `&circuit`
  - Since `read_from_file = true` circuit data is read from `circuit.in`.
- `&dispersion`
  - There are 10 sections in this circuit. There are three "pull turns," two uniform sections, a tapered section, another small uniform section, and finally three pull turns. To get the tapered section set `inrplt_sections = true` and list section number 6 in `interpol_sects_list`. Sections 6 and 7 have identical parameters so that paramters are continuous across their interface. This is an example of a moderately complicated circuit.
  - Since `use_antonsen_formula = true` one can put 0.0's in for all `space_charge_redux` values.
- `&losses`
  - Frequency independent loss is entered by setting two loss frequencies, one below `min_ckt_freq` and one above `max_ckt_freq`.
  - To compare how loss data is entered when `read_from_file = true`, see `losses.in`.

- `&numerical_data`
  - The numbers in this namelist are good starting points for most simulations.

4.4 Problems running?

THIS SECTION IS IN PROGRESS. PLEASE REPORT SUGGESTIONS.

1. Q: The program crashes while reading data. A:
2. Q: The program crashes while reading data. Error: "Too many values are specified for the item name X in the NAMELIST input data."
   A: Code built by the Lahey compiler does not like blank lines in namelists. Remove any blank lines or lines WITH ONLY COMMENTS from your namelists.
3. Q: Something is fishy. A: Try echoing input to make sure you are putting in what you think you are putting in.

5 Compiling

See `http://www.lmsuite.org`. Instructions will eventually be transferred to here.

5.1 Features accessible only by recompiling

This section describes some features in the code that are so specialized that they do not warrant control by namelist variables. To access the features one must edit the code and recompile.

5.1.1 Create only harmonic frequencies

The algorithm for computing intermodulation products is *very* inefficient if one only wants to compute harmonics of a fundamental. For modest numbers of harmonics this is not a problem, however for large numbers of harmonics, e.g. > 40, the time it takes to compute the harmonics becomes on the order of several minutes. When the number of harmonics is even larger, e.g. > 100 then it takes unacceptably long to compute the harmonics. The code contains a function that computes the harmonics using the simple harmonic formula, rather than the more general intermodulation frequency computation, which makes simulations using hundreds of space charge harmonics possible.

To use this function open the file `initialize.f90`, comment out the line `call create_frequency_array(.true.)`, and comment in the line `call create_harmonic_array(.true.)`. Recompile as described earlier in this section.

6 Contributing

See `http://www.lmsuite.org`. Instructions will eventually be transferred to here.

7 To Do List

7.1 Testing

Below are input scenarios for which the code has not been extensively tested. The code was written to handle the cases, but without testing one cannot be sure that the cases are handled correctly. If you are getting unexpected results from the code, have a look at this list to see if you are in "uncharted waters." Alternatively, if you would like to help test the code, consider putting it through its paces with any of the following input scenarios.

1. Space charge frequency range wider than circuit frequency range. Some aspects to look at include plotting magnitudes and phases of the various quantities. For example if choose to plot magnitude or phase of a circuit quantity are only circuit frequencies included in the output, if choose to plot a beam quantity do all space charge frequencies appear in the output.
2. The space charge reduction formula for an annular beam has not been tested.
3. The dispersion scan (`scanID = 4`) has not been tested extensively, particularly for loss and space charge reduction factor.

7.2 Models

1. LATTE
   (a) scan routines: beam parameter scan
   (b) plot routines: phase_vs_z (velocity only), phase_differences_vs_z (velocity only), disk_orbits_vs_z, conserved_quantity_vs_z, beam_energy_vs_z
2. MUSE
   (a) check that $\mathbf{V}$ matrix can be larger than number of space charge frequencies
   (b) slowly varying envelope implementation
   (c) scan routines: beam parameter scan
   (d) plot routines: conserved_quantity_vs_z, beam_energy_vs_z
3. S-MUSE
   (a) slowly varying envelope implementation
   (b) scan routines: beam parameter scan
   (c) plot routines: conserved_quantity_vs_z, beam_energy_vs_z

7.3 Physics

1. modulated electron beam (need initial values of space charge electric field E?)
2. 2-d beam stuff, thermal spread

7.4 Helix

1. Sheath helix solver
2. Tape helix solver
3. Loss model?

7.5 Beam

1. Antonsen correction to space charge reduction factor
2. DC space charge depression
3. Relativistic effects

7.6 Namelist

1. Test code defaults, i.e. remove `lmsuite.nml`
2. Put in dispersion and loss frequencies in real numbers (?) (Hz?)

7.7 Plotting

1. Roll all single pass `*_vs_freq` plots into one routine.

7.8 Numerical

1. reduce arrays for circuit and space charge field to their minimum number of elements
2. implement adaptive stepsize (dormand-prince)
3. implement linear multistep method (adams-bashford)

7.9 Compiler

1. study optimization flags for all supported platforms
2. g95 build

7.10 Documentation

1. Register copyright, see http://www.gnu.org/licenses/gpl-howto.html for instructions.

2. compilation and installation instructions

3. Manual

References

[1] D.K. Abe, M.T. Ngô, B. Levush, T.M. Antonsen, Jr., and D.P. Chernin. A comparison of L-band helix TWT experiments with CHRISTINE, a 1-D multifrequency helix TWT code. *IEEE Trans. Plasma Sci.*, 28(3):576–587, June 2000.

[2] T.M. Antonsen, Jr. and B. Levush. Traveling-wave tube devices with nonlinear dielectric elements. *IEEE Trans. Plasma Sci.*, 26(3):774–786, 1998.

[3] M.E. El-Shandwily. Analysis of multi-signal traveling-wave amplifier operation. Technical Report 85, Electron Physics Lab., University of Michigan, Ann Arbor, June 1965.

[4] A.J. Giarola. A theoretical description for the multiple-signal operation of a TWT. *IEEE Trans. Electron Devices*, ED-15(6):381–395, June 1968.

[5] A. Nordsieck. Theory of the large signal behavior of traveling-wave amplifiers. *Proc. IRE*, 41:630–637, May 1953.

[6] J.E. Rowe. *Nonlinear Electron-Wave Interaction Phenomena*. Academic Press, New York, 1965.

[7] J.G. Wöhlbier. *Nonlinear Distortion and Suppression in Traveling Wave Tubes: Insights and Methods*. PhD thesis, University of Wisconsin–Madison, 2003.

[8] J.G. Wöhlbier, J.H. Booske, and I. Dobson. Mechanisms for phase distortion in a traveling wave tube. Submitted to *Phys. Rev. E.*

[9] J.G. Wöhlbier, J.H. Booske, and I. Dobson. On the physics of harmonic injection in a traveling wave tube. Submitted to *IEEE Trans. Plasma Sci.*

[10] J.G. Wöhlbier, J.H. Booske, and I. Dobson. The Multifrequency Spectral Eulerian (MUSE) model of a traveling wave tube. *IEEE Trans. Plasma Sci.*, 30(3):1063–1075, 2002.

[11] J.G. Wöhlbier, I. Dobson, and J.H. Booske. Generation and growth rates of nonlinear distortions in a traveling wave tube. *Phys. Rev. E*, 66, 2002. Article 56504.