language=Matlab, frame=single

ALOHA Manual

J.Hillairet

Last update : 03/10/10

Abstract

This document is the documentation of the Advanced LOwer Hybrid Antenna code (ALOHA). It describes the main characteristic of the code and how to use it, for both basic or advanced uses.

Contents

1 Introduction

ALOHA (*Advanced LOwer Hybrid Antenna*) is a Lower Hybrid (LH) coupling code, i.e. a code which purpose is to calculate the coupling between a LH antenna (often called "grill") and a magnetised plasma, for a specified antenna and plasma model.

The physics underlying the code is based on the linear coupling theory mainly due to M.Brambilla[2]. However, in contrary to previous coupling code such as SWAN[6], ALOHA provide a 2D description of the LH antennas[1], which means that both y and z description of an antenna must be provided. An example of a typical LH antenna geometry as described in ALOHA is shown on figure 1.

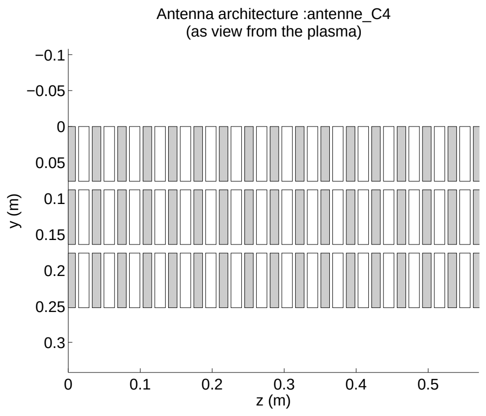

Figure 1: Example of a LH antenna geometry as described in ALOHA. This example corresponds to a Passive-Active Antenna (PAM) ; grey rectangles are passive waveguides.

The code core had been developed by between 1994 and 1996 by S.Berio and Ph.Bibet[3]. The project has been resumed in 2006 by D.Voyer and Ph.Bibet, then by D.Voyer and M.Goniche. Since 2008, the code is maintained by J.Hillairet[5, 4].

Edge plasma is described in ALOHA as an inhomogeneous cold plasma in the radial direction (for each toroidal line of waveguides). The plasma is described as a linear evolution of the density in the direction of the core plasma (x). One or two electron density gradients ∇n_e are taken into account after a step density n_{e0} which is supposed to be the density at the mouth of the waveguides. An example of a density profile with 2 density gradients ∇n_{e0}, ∇n_{e1} is shown on figure 2. Since the gradient value depends on the initial density, the slope of the linear density gradient is actually set using the decay length λ_n (in meters):

$$\lambda_n = \frac{n_e}{\nabla n_e} \tag{1}$$

For the antenna part, the electromagnetic waves are modelled by a discrete sum of conventional TE/TM modes. Since the plasma is supposed to have an infinite extend in the radial direction – which imply that all the waves are supposed to be

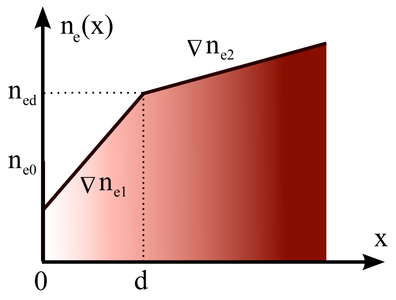

Figure 2: Example of density profile as it is defined in ALOHA.

absorbed into core plasma and satisfy to the WKB(J) boundaries conditions – the electromagnetic fields in the plasma are modelled by a continuous sum of planes waves.

In harmonic regime, boundaries conditions at the interface allow one to match antenna fields with plasma fields by expressing the plasma as a equivalent load for the antenna, ie as an equivalent surface admittance, and then to express the coupling between antenna and plasma.

Following the derivation of Bers and Theilhaber[1], ALOHA can handle both slow and fast waves (ALOHA-2D) or only slow waves (ALOHA-1D) and provide different results related on the LH coupling of an antenna with a plasma. Basic results returned are the reflection coefficients as if it were measured behind the antenna or the $n_\parallel$ antenna spectrum.

2 Preamble

ALOHA is a set of Matlab scripts and FORTRAN binaries. In order to run ALOHA, the code need to access to is own libraries. In order to do so, the directory containing all the ALOHA Matlab scripts must be load into the Matlab `PATH` system variable, in order that Matlab may be able to find them. Then, the first thing to do is to add the `libaloha` directory to the Matlab path[1] :

```
>> addpath(genpath(['absolute_path_to_aloha_main_dir/libaloha']));
```

This command can be automatically set up when starting Matlab. See the Matlab documentation.

There is two way to use ALOHA : a basic and an advanced. In the basic way, user runs a Matlab script which calculates the coupling with one antenna and for one plasma configuration.In the advanced way, users runs a Matlab function which take into input argument a “scenario”, i.e. a combinaison of antennas and plasma configurations. The advanced usage allows user to easily do multiple calculations and parameter studies, such as edge electron density variation, etc.

[1]where >> design the prompt line of Matlab.

3 Basic use

The simplest and the fastest way to use ALOHA is to modify and then to run the `aloha.m` Matlab script file, which is in the main directory. All the parameters and their meaning should be explained in this file. When running the `aloha.m` script file, all results are saved into the Matlab workspace into a Matlab structure named `scenario`. This structure contains all the input parameters such as the antenna description (antenna name, power feeding, etc.), plasma description (density, scrape-off length, etc.) and ALOHA options (1 or 2 gradients, 1D or 2D, etc.). The structure also contains the results calculated by ALOHA, such reflection coefficients, spectrum (if asked), directivity (if asked), etc.

4 Advanced use

4.1 scenario(s)

In order to run many simulations during the same run (as it could be with a Unix *batch*) or in order to save some reference simulations for future comparisons, one can use the ALOHA *scenarios*. An ALOHA “scenario” is a Matlab structure which contains all the parameters required by ALOHA to make a simulation. Using the `aloha_scenario` function and a ALOHA scenario as input, one retrieves (as a returned parameter) a new ALOHA scenario which contains the input scenario plus all the results computed by ALOHA. The general workflow of ALOHA is illustrated on Fig.3. More details on Antenna Scattering Matrix, Antenna feeding and Plasma description are given in the appendix sections.

Since Matlab structures can also been vectorized, one can construct vectors ($1 \times N$) of ALOHA scenarios and gives it as an input parameter to the `aloha_scenario` function function. In this case, ALOHA will compute all these scenarios and return a new vector of scenarios, each one containing its own results. See help for `aloha_scenario` for more details.

In order to make it easier post-treatments of the ALOHA scenario variables, a bundle of Matlab scripts exists to retrieve specific variables such as parameters or results from scenarios. Most of these function accept vectorized scenarios. An example is explained below.

4.2 Example

In order to generate an ALOHA scenario, an example named `scenario_example.m` is furnished in the `scenario` directory. This Matlab script contains all the parameters needed by ALOHA for a computation, and returns a scenario structure[2]. In the following example, this file is modified and saved to an other name : `scenario_testC2.m`.

[2]This means that the Matlab call should be as : `my_scenario = scenario_example();`.

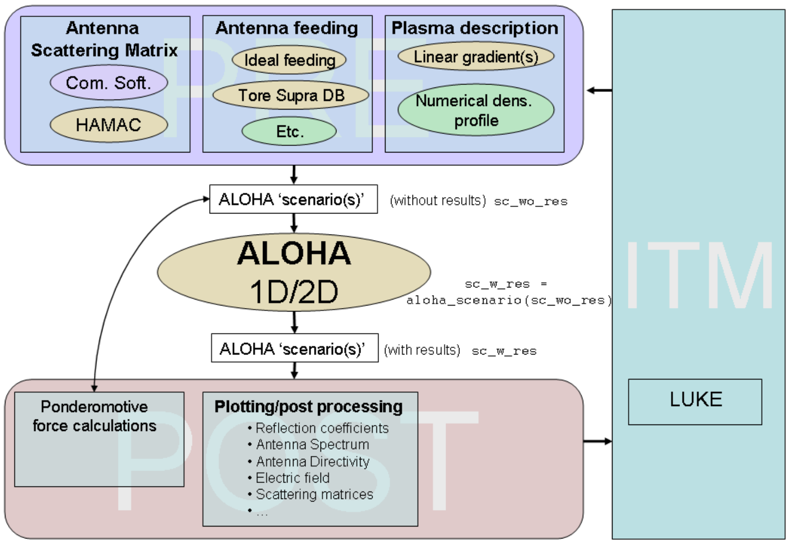

Figure 3: General description of the ALOHA workflow. All input parameters such as antenna and plasma descriptions are set into a Matlab structure which is called an ALOHA scenario. ALOHA process this input structure and return a new Matlab structure similar to the input one, but including all the calculated results. Many ALOHA function have been written in order to facilitate pre and post-processing.

4.2.1 Setting the main parameters

In this example, we want to compute the reflection coefficient and the spectrum for different electron densities for the Tore-Supra C2 antenna. First of all, we need to modify the `scenario_testC2.m` in order to fit with our requirements. Commonly used parameters are set to the following values[3]:

- `antenna.architecture = 'antenne_C2';`
- `options.version_code = '1D';`
- `version_plasma_1D = 3;`
- `plasma.ne0 = 5e17;`
- `plasma.lambda_n(1) = 2e-3;`
- `options.bool_compute_spectrum = true;`

[3]Other parameters are not modified in this example since they satisfy to our requirements. Of course, users must pay attention to other parameters and change them if necessary.

- `options.bool_display_spectrum = false;`

These parameters define a slow-wave only simulation (ALOHA-1D, `options.version_code = '1D'`), 1 gradient plasma (`version_plasma_1D=3`) with a step density of $n_{e0} = 5 \times 10^{17}\, m^{-3}$ and a scrape-off length of $\lambda_n = 2\, mm$. The antenna spectrum will be calculated, but not displayed : this choice is motivated by the fact that we will compute many scenarios, so it is unnecessary to show the spectrum for each of these scenarios[4].

4.2.2 Testing for one scenario

First, we generate the ALOHA scenario from the script we had modified in the previous section. Calling this script will give back a Matlab structure corresponding to the desired ALOHA scenario :

```
>> my_scenario = scenario_testC2.m

my_scenario =

    antenna: [1x1 struct]
    options: [1x1 struct]
     plasma: [1x1 struct]
    results: [1x1 struct]
```

At this step, you can check that the field `results` of the structure `my_scenario` is empty : this is normal, since we haven't run ALOHA yet. In order to calculate all the desired results, we ask ALOHA to process this scenario :

```
>> my_scenario_with_results = aloha_scenario(my_scenario);
```

Then, ALOHA will display many messages lines, which indicate in which step the code is. When the computation is finished, the new variable `my_scenario_with_results` will be :

```
>> my_scenario_with_results

my_scenario_with_results =

    antenna: [1x1 struct]
    options: [1x1 struct]
     plasma: [1x1 struct]
    results: [1x1 struct]
```

Now, the structure `my_scenario_with_results` contains the same parameters that the structure `my_scenario`, but also the results computed by ALOHA. These results are saved is the field `results` of the structure :

[4]Once the scenarios have been processed by the `aloha_scenario` function, the spectrum can be plot by either `aloha_plot_spectrum` or `aloha_plot_spectra` functions (see the help for these functions).

```
>> my_scenario_with_results.results

ans =

            CoeffRefPuiss: [8x1 double]
                      Pin: 2.0672e+06
          Ptr_plasma_RC: 2.0228e+06
    Ptr_plasma_spectrum: 1.9010e+06
                S_acces: [8x8 double]
               S_plasma: [204x204 double]
                a_acces: [8x1 double]
               a_plasma: [204x1 double]
                b_acces: [8x1 double]
               b_plasma: [204x1 double]
                     dP: [41x1001 double]
                  dP_nz: [1x1001 double]
            directivite: 0.7599
        directivite_tab: [1x500 double]
                    dny: 0.1000
                    dnz: 0.0200
                     ny: [1x41 double]
                     nz: [1x1001 double]
                rac_Zhe: [204x204 double]
...
```

In order to get or to watch for one of these results, one can uses either the Matlab workspace environment[5] (or Matlab command line) or more conveniently some of the ALOHA functions. By example, knowing that the reflection coefficients Γ^2 (in percents) for 8 modules of the C2 antenna are contained in the field named `CoeffRefPuiss`[6], one can get these values using the `aloha_scenario_get` function:

```
>> RC=aloha_scenario_get(my_scenario_with_results, 'CoeffRefPuiss')

RC =

    2.2070
    1.6138
    2.4489
    2.1221
    2.6350
    2.2712
    1.9757
    1.8985
```

[5] See help for `workspace` or `openvar` Matlab functions.

[6] In the Matlab structure tree, this is in fact `my_scenario_with_results.results.CoeffRefPuiss`

which are the reflection coefficients. The function `aloha_scenario_get` can take multiple arguments, which can be convenient for getting multiple parameters. By example, if one wants to get the plasma parameters for which the simulation had been made:

```
> [n_e,lamb_n,vers]= ...
      aloha_scenario_get(my_scenario_with_results, ...
                'ne0','lambda_n','version');
```

4.2.3 Running for multiple scenarios

Working with multiple scenarios is similar to working with only one. The only difference is that the input scenario parameter would be now a 1D-array of scenario structures. Moreover, the resulting variable from `aloha_scenario` will be an array of scenario structures containing all the results for each scenarios. Let by example do 2 computation for 2 different electronic densities, still using our preceding example :

```
>> sc(1)=scenario_testC2;
>> sc(2)=scenario_testC2; % duplicate scenario_test2
>> sc(2).plasma.ne0=10e17; % change only ne into scenario #2
>> aloha_scenario_get(sc, 'ne0') % check

ans =

   1.0e+18 *

    0.5000
    1.0000
```

Then, running ALOHA for this array of scenarios:

```
>> sc=aloha_scenario(sc);
```

Results can be easily extracted from the resulting array of scenarios `sc` still using the `aloha_scenario_get` function. For multiple scenarios, results coming from same fields are stacked together : scalar values becomes array (which length is the length of the scenario array), vectors becomes matrices and matrices becomes array of matrices.

In order to facilitate the computation of many scenarios, the file `batch_example.m` provides an example of a way of doing a density scan.

4.2.4 Save and load scenario(s)

In order to easily keep the results obtained, one can save a scenario (or an array of scenarios) using the `aloha_scenario_save` function[7] :

[7]Scenario files are saved in the Matlab-V6 `'.mat'` file format, in order to assure backward compatibility with previous versions.

```
>> aloha_scenario_save(sc, 'my_scenario.mat');
```

Retrieving this scenario can be done using the `aloha_scenario_load` function[8] ;

```
>> sc = aloha_scenario_load('my_scenario.mat');
```

4.3 Plotting results

The ALOHA library contains some useful functions for plotting results which are contained into a scenario. All of these functions start with `aloha_plot_`. Most usefull functions are:

- `aloha_plot_export` : export a Matlab figure to an image file (.png, .tif, .pdf, etc.) with a correct aspect for publication or slide purposes ;
- `aloha_plot_figure` : generate a new figure with correct options for publication or slide purposes ;
- `aloha_plot_spectrum` : plot spectrum for each scenario
- `aloha_plot_spectra` : plot superposed spectrum of scenarios for comparison.
- `aloha_plot_reflectionCoeff` : plot reflection coefficient(s).
- `aloha_plot_antenna` : plot the antenna used in the scenario, as view from the plasma ;
- `aloha_plot_champEmbouchure` : plot the toroidal electric field E_z into the grill mouth (i.e. for $x = 0.$) ;
- `aloha_plot_directivity` : plot the directivity of the antenna for the scenario. The directivity definition can be set with an optionnal parameter.

See the help for each of these function.

4.4 Other functions, documentation

All ALOHA functions begin with `aloha_`. All the functions of the ALOHA library are described in the `documentation` directory which should be in the main directory (see `index.html` in this directory). If this directory does not exist, one could generate this documentation using the `aloha_install_doc` function.

5 FAQ

TO-DO

[8] We assume that we are in the same directory that the `.mat` file.

6 References

References

[1] A. Bers and K.S. Theilhaber. Three-Dimensional Theory of Waveguide-Plasma Coupling. *Nuclear Fusion*, 23(1):41–48, 1983.

[2] Marco Brambilla. Slow-Wave Launching at the Lower Hybrid Frequency Using a Phased Waveguide Array. *Nuclear Fusion*, 16(1):47–54, 1976.

[3] St?phane B?rio and Philippe Bibet. Numerical simulation of the coupling properties of advanced lower hybrid waves launchers. In *23rd EPS Conference on Controlled Fusion and Plasma Physics, Kiev, Ukrain*, June 1996.

[4] J. Hillairet, O. Meneghini, D. Milanesio, D. Voyer, and M. Goniche. Comparisons between the TOPLHA and the ALOHA codes on Lower Hybrid antenna coupling. In Volodymyr Bobkov and Jean-Marie Noterdaeme, editors, *Radio Frequency Power In Plasmas: Proceedings Of The 18th Topical Conference, Aip Conference Proceedings*, volume 1187, pages 379–382. AIP, 2009.

[5] Julien Hillairet, Damien Voyer, Biance Frincu, Orso Meneghini, Annika Ekedahl, and Marc Goniche. Modeling of lower hybrid antennas using the ALOHA code and comparisons with Tore Supra experiments. *Fusion Engineering and Design*, 84:953–955, 2009.

[6] Didier Moreau and T.K. Nguyen. Couplage de l'onde lente au voisinage de la fr?quence hybride basse dans les grands Tokamaks. Technical report, EUR-CEA-FC1246 Euratom-CEA, 1984.

A Antenna geometrical and RF description

A.1 General description

The global frame, as defined in classical papers [2, 1], is illustrated in Fig.4:

- x is the radial direction towards plasma. $x = 0$ is the plane of the antenna mouth;
- y is the poloidal direction;
- z is the toroidal direction.

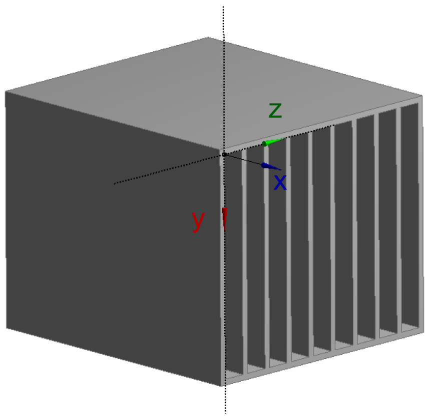

Figure 4: Axes definition in ALOHA. z is the toroidal direction, y the poloidal direction and x the radial direction towards plasma. $x = 0$ in the plane of the antenna mouth.

In ALOHA the coupling between waveguides and plasma is calculated in the plane $x = 0$. An account of the curvature of an antenna can be made with its scattering matrix.

A.2 Front-face geometrical description

A.3 RF description

Antenna's scattering matrix can be calculated from any commercial RF software, or with HAMAC, an ALOHA Matlab plug-in.

TODO HAMAC

B Antenna feeding

In ALOHA, a LH antenna is described as an assembling of one or many modules. Transmission lines which feed the antenna are supposed to be connected to each module. Thus, assuming that power input of an antenna is defined with a vector $\mathbf{P}_{in}$, size vector will be of the size $(1 \times N_{mod})$ where N_{mod} is the total number of modules. Amplitude and phase informations are required for each module, which means that the vector $\mathbf{P}_{in}$ may be complex valued.

B.1 Experimental feeding from Tore Supra database

In Tore Supra, all builted LH antennas consisted in 2 rows of 8 modules. Modules are indexed as illustrated in Fig.5.

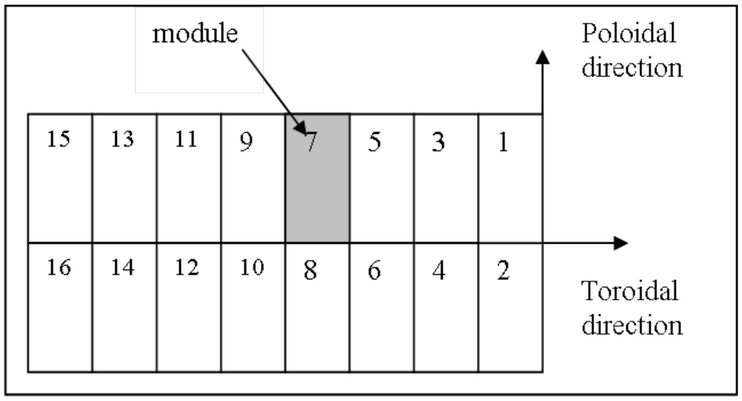

Figure 5: Tore Supra module indexing description as viewed from the plasma.

Tore Supra database signals for injected power are :

- `GPINJC1`$(t, 16)$: incident power in module $n = (1 \ldots 16)$ for the first coupler in kW;
- `GPINJC2`$(t, 16)$: incident power in module $n = (1 \ldots 16)$ for the second coupler in kW;
- `GPHIC1` $(t, 16)$: incident phase in module $n = (1 \ldots 16)$ for the first coupler in degree;
- `GPHIC2`$(t, 16)$: incident phase in module $n = (1 \ldots 16)$ for the second coupler in degree.

Since in ALOHA, module indexing starts from the left to the right as view from the plasma, the following reordering is made automatically when using database signals:

```
p_inc_mes = p_inc_mes([15 13 11 9 7 5 3 1 16 14 12 10 8 6 4 2]);
```

which means that the 8 first values of `p_inc_mes` correspond to upper module injected power and the 8 last values correspond to lower modules.

Remark : Each module consists in a multi-junction, which RF properties are different due to the toroidal and poloidal curvatures of the antenna. Thus, each module have its own scattering matrix. Modules are referenced as in the following Table.

Antenna	Module reference, Left to Right (view from plasma)
C2 (before flip) Top	2H 1H 2H 1H 2H 1H 2H 1H
Bottom	2B 1B 2B 1B 2B 1B 2B 1B
C2 (after flip) Top	1B 2B 1B 2B 1B 2B 1B 2B
Bottom	1H 2H 1H 2H 1H 2H 1H 2H
C3 Top	24H 23H 22H 21H 14H 13H 12H 11H
Bottom	24B 23B 22B 21B 14B 13B 12B 11B
C4 Top	24H 23H 22H 21H 14H 13H 12H 11H
Bottom	24B 23B 22B 21B 14B 13B 12B 11B

Table 1: Tore Supra LH antenna modules reference.

C Spectrum of the whole antenna

Suppose that the whole antenna is composed by the two same half-antennas, and that the electric fields in both antenna waveguides are the same. Then the total electric and magnetic fields are :

$$\begin{aligned}
\mathbf{E}_{\text{total}}(y,z) &= \mathbf{E}(y-\frac{\Delta}{2},z)+\mathbf{E}(y+\frac{\Delta}{2},z) && (2)\\
\mathbf{H}_{\text{total}}(y,z) &= \mathbf{H}(y-\frac{\Delta}{2},z)+\mathbf{H}(y+\frac{\Delta}{2},z) && (3)
\end{aligned}$$

If we define fourier transform as :

$$\tilde{A}=\frac{1}{2\pi}\iint A(y,z)e^{jk_y y+jk_z z}dy\,dz \qquad (4)$$

Then the fourier transform of the fields are :

$$\begin{aligned}
\tilde{\mathbf{E}}_{\text{total}}(k_y,k_z) &= e^{-jk_y\frac{\Delta}{2}}\tilde{\mathbf{E}}(k_y,k_z)+e^{+jk_y\frac{\Delta}{2}}\tilde{\mathbf{E}}(k_y,k_z)\\
&= 2\cos\left(k_y\frac{\Delta}{2}\right)\tilde{\mathbf{E}}(k_y,k_z) && (5)
\end{aligned}$$

The complex power spectrum, which is related to the product

$$d\mathbf{P}_{\text{total}}\equiv\tilde{\mathbf{E}}_{\text{total}}\times\tilde{\mathbf{H}}^*_{\text{total}} \qquad (6)$$

can then be found in function of the spectrum of one half-antenna:

$$d\mathbf{P}_{\text{total}}=4\cos^2\left(k_y\frac{\Delta}{2}\right)d\mathbf{P}_{1/2} \qquad (7)$$