Simulation Class: *Wave equations, Standing Wave, Voltage and Current time evolution, Reflection Coefficient, Impedance and Admittance*

Please: Install the RF Toolbox; annotate **each figure adequately by using legends, axis labels and titles**.

Input data:

- Generator: V_g=2 V, Z_g=Z_0=50Ω
- Line length (air dielectric): *1. 25 m*
- Line characteristic impedance: Z_0=50 Ω
- Frequency: *300 MHz*
- Attenuation constant: α (*start simulations with 0 and then change to 0.2 Neper/m at point 6)*
- Load impedance Z_L: *user input (can be complex).*

Problem solving steps

1. Compute the wavelength λ, the period T, the load reflection coefficient ρ_L, the propagation constant, the angular frequency ω and generate a vector of distance x with 1000 points.
2. Compute the general phasor equations for:
   a. Incident Voltage and Current -$\mathbf{V}_i$(x) e $\mathbf{I}_i$(x)- (2 vectors)
   b. Reflected Voltage and Current -$\mathbf{V}_r$(x) e $\mathbf{I}_r$(x)- (2 vectors)
3. Compute the standing wave voltage amplitude vector V_{stand} in the transmission line (absolute value of sum of the incident and reflected voltage phasors), plot it as a function of **x** (activate the grid) and estimate, by hand or using graphical facilities, the VSWR;
4. Do as above **for the current** I_{stand}: use the previous figure but plot the current multiplied by 50 (for easier reading simultaneously with the voltage standing wave) and in red color;
5. Plot, in the same figure the symmetrical of the standing waves.
6. Develop an animation showing the propagation of the voltage along the line. Using a time step of T_{step}=T/32 (T is the period):
   a. Make as the active figure the figure of previous point 3 and 4 (standing wave);
   b. Freeze the ranges: use function *ylim* (or *axis*) for a symmetric ranging of y axis (or both axis);
   c. hold the figure (*hold on*);
   d. Set time to 0: t1=0;
   e. Do in a loop (suggested number of snapshots 5*32) *Note: do not use j or i for the loop control variable*
      i. Plot at time t_1 a snapshot of the line voltage vector $real\left(\boldsymbol{V}(x)e^{j\omega t_1}\right)$ as a function of **x** getting an handle to the plot hv *(**hv**=plot(…))*; *(the use of the time domain equations can also be used and are likely to be more efficient: try at home)*
      ii. Plot at time t_1 a snapshot of the scaled line current vector $real\left(\boldsymbol{I}(x)e^{j\omega t_1}\right)$ as a function of **x** getting an handle to the plot hi *(**hi**=plot(…))*;
      iii. Pause for a fraction of second (0.25s for example);
      iv. Increase t_1 by T_{step}
      v. Delete the two plots deleting the handles (*delete(hv); delete(hi)*) [i].
7. Comment the results and simulate for other load impedances and mainly for the **SC** and **CA.**
8. Compute and plot, in a new figure, a polar plot of the (complex) reflection coefficient (use the definition that is: $\rho(x) = \boldsymbol{V}_r(x)/\boldsymbol{V}_i(x)$).

Input impedance and susceptance

1. Compute the line input impedance $\mathbf{Z}_{in}(x)$ (use the definition ($\mathbf{V}(x)/\mathbf{I}(x)$) and plot in a new figure the resistance and reactance as a function of **x** (use different colors for real –resistance- and imaginary –reactance- parts).
2. Identify manually, or moving a point in the plot using graphical facilities, the closest distances to the load , x_1 and x_2, adequate to match the system by using a reactance in series with the line;
3. Compute the input admittance $\mathbf{Y}_{in}(x)$ ($1/\mathbf{Z}_{in}(x)$) and plot the conductance and susceptance (as above) in a new figure;
4. Identify the closest distances to the load, x_1' and x_2', adequate to match the system by inserting a parallel susceptance;
5. Plot as well, in a new figure, the complex impedance in a plane and make the *"axis equal"*. Plot also, in a new figure, the reflection coefficient in a Smith Chart (use the function *smithplot()*).
6. Run again the full script you have till now but using α=0.2 Np/m or other values.

Home work

1. Study carefully all the subjects addressed in this simulation class.

Some useful MatLab functions: atan2(y,x); abs(); angle(); plot(x,y,'b'); axis([x1 x2 y1 y2]), polar(θ, r); grid on; axis square; polarplot(); hold on; linspace; xlim(); ylim(); pause(); real(); imag(); pause(), figure(); legend(); xlabel(); ylabel(); title(); ylim(); xlim(); title(); smithplot()

Latex use in labels, legends, etc is possible: Z_{in} looks like Z_{in} ()

Note: Use "." "./" and ".^2" instead of "*" "/" "^" to make operation point by point.*

---

[i] To execute the animation you can also retrieve the handle to the plotted data and update it. The example plots a moving point over a sine plot

```
pause_time = .1;
nx = 150; x=linspace(0,2*pi,nx); y=sin(x);
figure(1); clf; plot(x,y); hold on; grid on;

h = plot(x(1),y(1),'ro'); %Gets an handle to the plotted point to be moved

for n=1:nx
   set(h,'XData',x(n),'YData',y(n)); %Replaces the moving point plotted
   pause(pause_time)
end
```

A possible solution is also to use the function *linkdata* that automatically updates the plot if plotted data is changed

Example

```
t=1:10;
x = randn(10,1);
plot(t,x);
linkdata on;        %Any future update to x is automatically updated in the plot;
pause(2);
x=randn(10,1);
```