

Arbitrary circuit analysis

This example goes through creating a **circuit** object from a **netlist**, which is contained in a text file. We start by calling the Circuit class and giving it the path to the netlist. The netlist automatically gets stored as a property called **'list'** on the circuit object.

The circuit structure in this case, has been arbitrarily chosen.

```
circuit = Circuit('circuits/passive/c7_rlc_arbitrary.txt');  
circuit.list
```

```
ans =  
'Vcc 3 0 DC Vcc  
Vs 4 1 AC 0  
R2 3 2 1000  
R1 1 0 1000  
C1 4 0 0.000001  
C2 2 4 0.00001  
,
```

To symbolically **analyze** the circuit object, we give it to the **ELAB** class. This performs **MNV** to define the equations, which describe the circuit.

```
ELAB.analyze(circuit)
```

```
Symbolic analysis successful (0.505408 sec).
```

An **electrical circuit** is just a subclass of a generic system, which can be represented in any number of ways. For example, as a transfer function in the **symbolic domain**. To see this representation, we can use the **'ec2sd'**-function.

```
sym_TF = ELAB.ec2sd(circuit, 3, 2)
```

```
Symbolic transfer function calculated successfully (6.959100e-03 sec).
```

```
sym_TF =
```

$$\frac{v_2}{v_3} = \frac{V_{cc} + C_1 R_1 V_{cc} s + C_2 R_1 V_{cc} s + C_2 R_2 V_s s}{V_{cc} (C_1 R_1 s + C_2 R_1 s + C_2 R_2 s + C_1 C_2 R_1 R_2 s^2 + 1)}$$

Say the netlist contained numerical values for some, or all of its components. The evaluate function will use whatever equations were found by ELAB.analyze() to numerically **evaluate** the circuit. This numerical evaluation can now be used to create a Matlab-transfer-function object.

```
ELAB.evaluate(circuit)
```

```
Numerical evaluation successful (0.183118 sec).
```

```
num_TF = ELAB.ec2tf(circuit, 3, 2)
```

```
Transfer function object created successfully (3.580330e-02 sec).
```

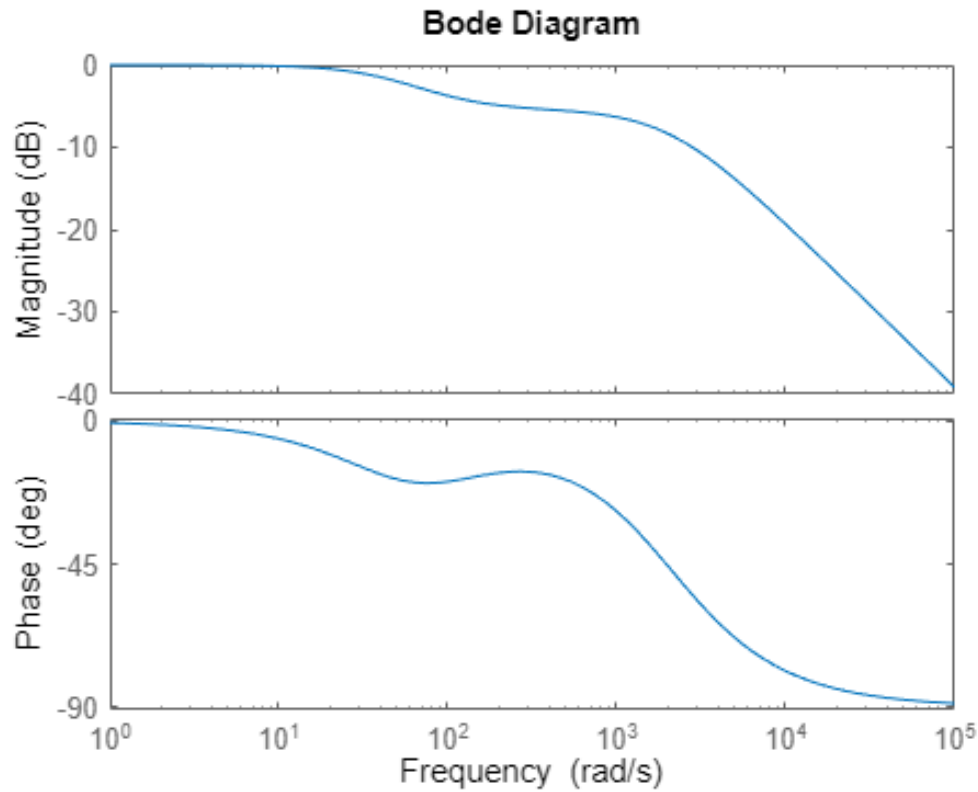
```
num_TF =
```

$$\frac{1100 s + 100000}{s^2 + 2100 s + 100000}$$

Continuous-time transfer function.

Matlab already has extensive functionality to handle transfer function objects.

```
bode(num_TF);
```



During analysis, all node voltages and element currents were symbolically defined and stored in the circuit object.

```
circuit.symbolic_node_voltages
```

ans =

$$\left(\begin{array}{l} v_1 = -\frac{R_1 s (C_1 V_s - C_2 V_{cc} + C_2 V_s + C_1 C_2 R_2 V_s s)}{\sigma_1} \\ v_2 = \frac{V_{cc} + C_1 R_1 V_{cc} s + C_2 R_1 V_{cc} s + C_2 R_2 V_s s}{\sigma_1} \\ v_3 = V_{cc} \\ v_4 = \frac{V_s + C_2 R_1 V_{cc} s + C_2 R_2 V_s s}{\sigma_1} \end{array} \right)$$

where

$$\sigma_1 = C_1 R_1 s + C_2 R_1 s + C_2 R_2 s + C_1 C_2 R_1 R_2 s^2 + 1$$

circuit.symbolic_element_currents

ans =

$$\begin{pmatrix} i_{R2} = \frac{C_2 s (V_{cc} - V_s + C_1 R_1 V_{cc} s)}{\sigma_1} \\ i_{R1} = -\frac{s (C_1 V_s - C_2 V_{cc} + C_2 V_s + C_1 C_2 R_2 V_s s)}{\sigma_1} \\ i_{C1} = \frac{V_s + C_2 R_1 V_{cc} s + C_2 R_2 V_s s}{C_1 \sigma_1} \\ i_{C2} = \frac{V_{cc} - V_s + C_1 R_1 V_{cc} s}{C_2 \sigma_1} \end{pmatrix}$$

where

$$\sigma_1 = C_1 R_1 s + C_2 R_1 s + C_2 R_2 s + C_1 C_2 R_1 R_2 s^2 + 1$$

And during evaluation, they were all numerically evaluated and stored. Since no numerical value was given for V_{cc} , the evaluation is partial.

circuit.numerical_node_voltages

ans =

$$\begin{pmatrix} v_1 = \frac{V_{cc} s}{100 \left(\frac{s^2}{100000} + \frac{21 s}{1000} + 1 \right)} \\ v_2 = \frac{V_{cc} + \frac{11 V_{cc} s}{1000}}{\frac{s^2}{100000} + \frac{21 s}{1000} + 1} \\ v_3 = V_{cc} \\ v_4 = \frac{V_{cc} s}{100 \left(\frac{s^2}{100000} + \frac{21 s}{1000} + 1 \right)} \end{pmatrix}$$