Filter Realizations Using Lumped Elements

Our **first** filter circuit will be "**realized**" with lumped elements.

Lumped elements—we mean inductors L and capacitors C!

Since each of these elements are (ideally) perfectly **reactive**, the resulting filter will be **lossless** (ideally).

We will first consider two configurations of a **ladder circuit**:

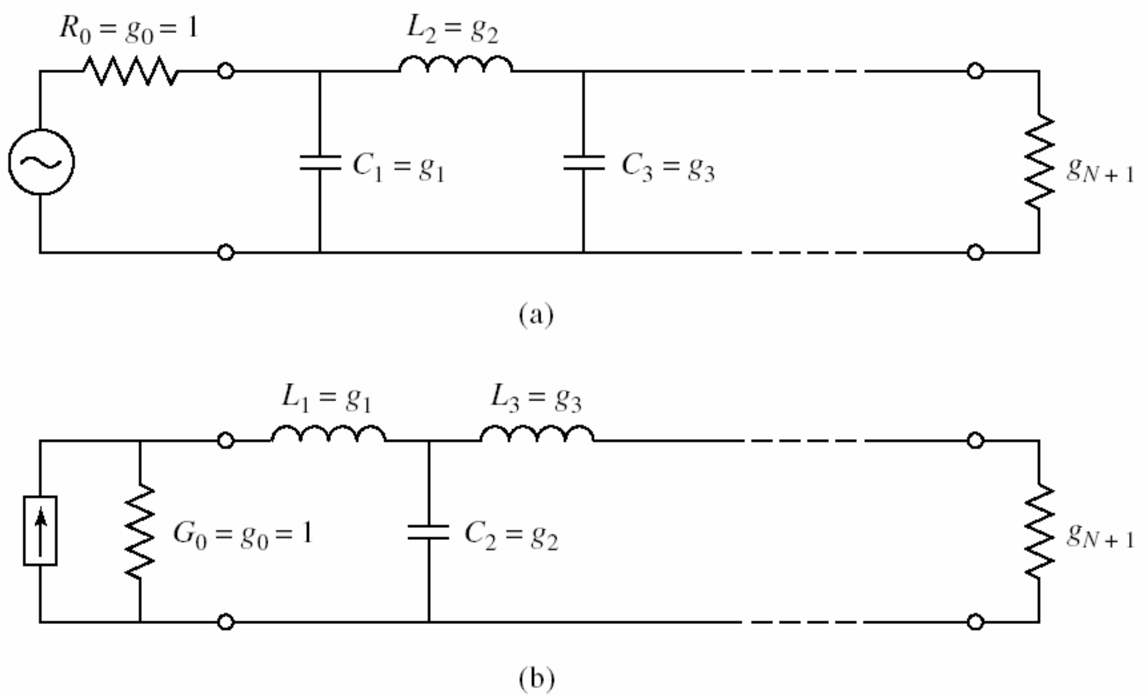

Figure 8.25 (p. 393)
Ladder circuits for low-pass filter prototypes and their element definitions. (*a*) Prototype beginning with a shunt element. (*b*) Prototype beginning with a series element.

Note that these two structures provide a **low-pass** filter response (evaluate the circuits at $\omega = 0$ and $\omega = \infty$!).

Moreover, these structures have N different **reactive elements** (i.e., N degrees of design freedom) and thus can be used to realize an ***N*-order** power loss ratio.

For example, consider the **Butterworth** power loss ratio function:

$$P_{LR}(\omega) = 1 + \left(\frac{\omega}{\omega_c}\right)^{2N}$$

Recall this is a **low-pass** function, as $P_{LR} = 1$ at $\omega = 0$, and $P_{LR} = \infty$ at $\omega = \infty$. Note also that at $\omega = \omega_c$:

$$P_{LR}(\omega = \omega_c) = 1 + \left(\frac{\omega_c}{\omega_c}\right)^{2N} = 1 + 1^{2N} = 2$$

Meaning that:

$$\Gamma(\omega = \omega_c) = \mathrm{T}(\omega = \omega_c) = 1/2$$

In other words, ω_c defines the 3dB bandwidth of the low-pass filter.

Likewise, we find that this Butterworth function is **maximally flat** at $\omega = 0$:

$$P_{LR}(\omega = 0) = 1 + \left(\frac{0}{\omega_c}\right)^{2N} = 1$$

and:

$$\left.\frac{d^n\ P_{LR}(\omega)}{d\omega^n}\right|_{\omega=0} = 0 \qquad \text{for all } n$$

Now, we can determine the function $P_{LR}(\omega)$ for a lumped element ladder circuit of N elements using our knowledge of **complex circuit theory**.

Then, we can **equate** the resulting polynomial to the **maximally flat** function above. In this manner, we can determine the appropriate **values** of all inductors L and capacitors C!

An **example** of this method is given on pages 392 and 393 of your book. In this case, the filter is very **simple**—just **one** inductor and **one** capacitor. However, as the book shows, finding the solution requires quite a bit **complex algebra**!

Fortunately, your book likewise provides **tables** of complete Butterworth and Chebychev Low-Pass **solutions** (up to order 10) for the ladder circuits of figure 8.25—**no** complex algebra required!

TABLE 8.3 Element Values for Maximally Flat Low-Pass Filter Prototypes ($g_0 = 1, \omega_c = 1, N = 1$ to 10)

N	g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8	g_9	g_{10}	g_{11}
1	2.0000	1.0000									
2	1.4142	1.4142	1.0000								
3	1.0000	2.0000	1.0000	1.0000							
4	0.7654	1.8478	1.8478	0.7654	1.0000						
5	0.6180	1.6180	2.0000	1.6180	0.6180	1.0000					
6	0.5176	1.4142	1.9318	1.9318	1.4142	0.5176	1.0000				
7	0.4450	1.2470	1.8019	2.0000	1.8019	1.2470	0.4450	1.0000			
8	0.3902	1.1111	1.6629	1.9615	1.9615	1.6629	1.1111	0.3902	1.0000		
9	0.3473	1.0000	1.5321	1.8794	2.0000	1.8794	1.5321	1.0000	0.3473	1.0000	
10	0.3129	0.9080	1.4142	1.7820	1.9754	1.9754	1.7820	1.4142	0.9080	0.3129	1.0000

Source: Reprinted from G. L. Matthaei, L. Young, and E. M. T. Jones, *Microwave Filters, Impedance-Matching Networks, and Coupling Structures* (Dedham, Mass.: Artech House, 1980) with permission.

TABLE 8.4 Element Values for Equal-Ripple Low-Pass Filter Prototypes ($g_0 = 1$, $\omega_c = 1$, $N = 1$ to 10, 0.5 dB and 3.0 dB ripple)

0.5 dB Ripple

N	g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8	g_9	g_{10}	g_{11}
1	0.6986	1.0000									
2	1.4029	0.7071	1.9841								
3	1.5963	1.0967	1.5963	1.0000							
4	1.6703	1.1926	2.3661	0.8419	1.9841						
5	1.7058	1.2296	2.5408	1.2296	1.7058	1.0000					
6	1.7254	1.2479	2.6064	1.3137	2.4758	0.8696	1.9841				
7	1.7372	1.2583	2.6381	1.3444	2.6381	1.2583	1.7372	1.0000			
8	1.7451	1.2647	2.6564	1.3590	2.6964	1.3389	2.5093	0.8796	1.9841		
9	1.7504	1.2690	2.6678	1.3673	2.7239	1.3673	2.6678	1.2690	1.7504	1.0000	
10	1.7543	1.2721	2.6754	1.3725	2.7392	1.3806	2.7231	1.3485	2.5239	0.8842	1.9841

3.0 dB Ripple

N	g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8	g_9	g_{10}	g_{11}
1	1.9953	1.0000									
2	3.1013	0.5339	5.8095								
3	3.3487	0.7117	3.3487	1.0000							
4	3.4389	0.7483	4.3471	0.5920	5.8095						
5	3.4817	0.7618	4.5381	0.7618	3.4817	1.0000					
6	3.5045	0.7685	4.6061	0.7929	4.4641	0.6033	5.8095				
7	3.5182	0.7723	4.6386	0.8039	4.6386	0.7723	3.5182	1.0000			
8	3.5277	0.7745	4.6575	0.8089	4.6990	0.8018	4.4990	0.6073	5.8095		
9	3.5340	0.7760	4.6692	0.8118	4.7272	0.8118	4.6692	0.7760	3.5340	1.0000	
10	3.5384	0.7771	4.6768	0.8136	4.7425	0.8164	4.7260	0.8051	4.5142	0.6091	5.8095

Source: Reprinted from G. L. Matthaei, L. Young, and E. M. T. Jones, *Microwave Filters, Impedance-Matching Networks, and Coupling Structures* (Dedham, Mass.: Artech House, 1980) with permission.

Q: *What?! What the heck do these values g_n mean?*

A: We can use the values g_n to find the values of inductors and capacitors required for a given **cutoff frequency** ω_c and source resistance R_s (Z_0).

Specifically, we use the values of g_n to find ladder circuit **inductor** and **capacitor** values as:

$$L_n = g_n \left(\frac{R_s}{\omega_c} \right) \qquad C_n = g_n \left(\frac{1}{R_s\, \omega_c} \right)$$

where $n = 1, 2, \cdots, N$

Likewise, the value g_{N+1} describes the **load impedance**. Specifically, we find that **if** the **last** reactive element (i.e., g_N) of the ladder circuit is a **shunt capacitor**, then:

$$R_L = g_{N+1}\, R_s$$

Whereas, **if** the **last** reactive element (i.e., g_N) of the ladder circuit is a **series inductor**, then:

$$R_L = \frac{R_s}{g_{N+1}}$$

Note however for the **Butterworth** solutions (in Table 8.3) we find that $g_{N+1} = 1$ **always**, and therefore:

$$R_L = R_s$$

regardless of the last element.

Moreover, we note (in Table 8.4) that this (i.e., $g_{N+1} = 1$) is likewise true for the Chebychev solutions—provided that ***N* is odd**!

Thus, since we typically desire a filter where:

$$R_L = R_s = Z_0$$

We can use **any** order of **Butterworth** filter, or an **odd** order of **Chebychev**.

→ In other words, avoid **even order Chebychev** filters!