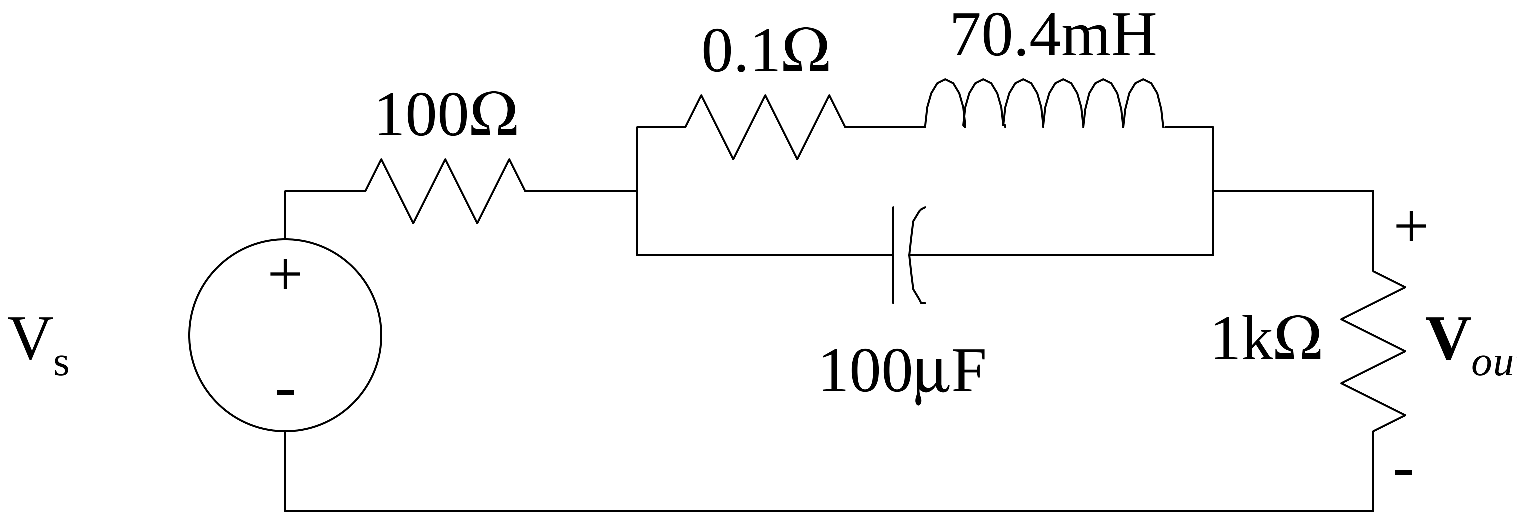
# Notch Filter Computations




$$V_s = ZI; \quad V_o = R_3 I = \frac{R_3}{Z} V_s$$

$$Z = R_1 + R_3 + Z_e; \quad Z_e = \left( \frac{1}{sL + R_2} + sC \right)^{-1} = \frac{sL + R_2}{s^2 LC + sCR_2 + 1}$$

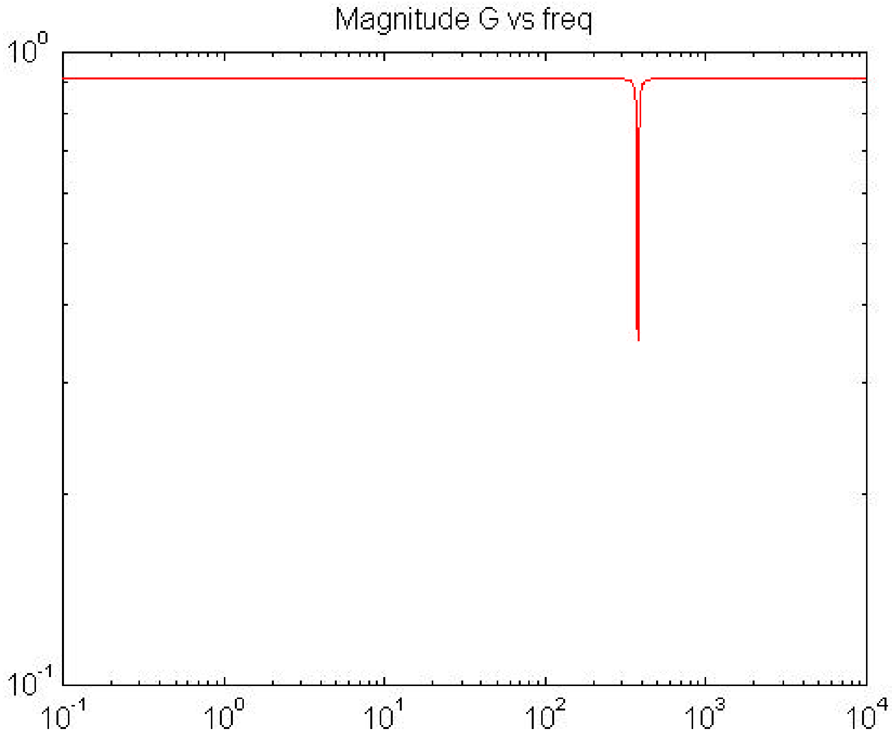
$$Z = \frac{s^2 LC(R_1 + R_3) + s[CR_2(R_1 + R_3) + L] + (R_1 + R_2 + R_3)}{s^2 LC + sCR_2 + 1}$$

$$V_o = \frac{s^2 LCR_3 + sCR_2 R_3 + R_3}{s^2 LC(R_1 + R_3) + s[CR_2(R_1 + R_3) + L] + (R_1 + R_2 + R_3)} V_s$$

# Matlab Computations

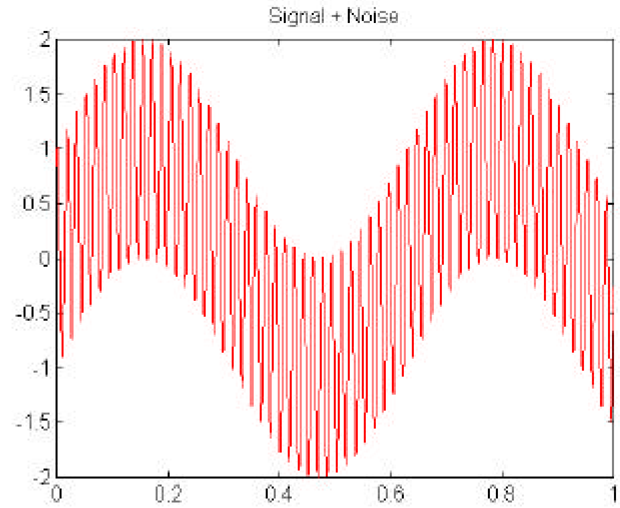
Let's find the frequencies rejected by this filter using Matlab:

» L=70.4e-3;R1=100;R2=0.1;R3=1e3;C=100e-6; % define the component values
» w=logspace(-1,4,1000); s=j*w ; % define a frequency vector
» % compute the frequency response
» G=(s.*s*L*C*R3+s*C*R2*R3+R3)./(s.*s*L*C*(R1+R3)+s*(C*R2*(R1+R3)+L)+R1+R2+R3);
» loglog(w,abs(G)) % and plot the magnitude



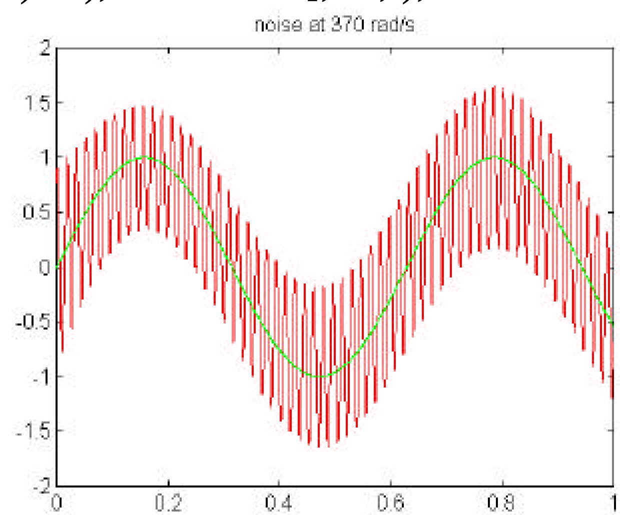

From the plot, this notch filter should attenuate frequencies ~370(rad/s)

# Matlab Simulation



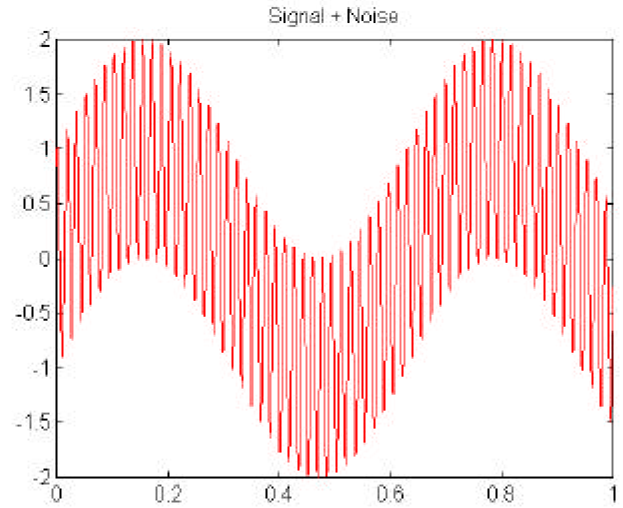

Now, let's visualize how this filter works:

```
» t=[0:1e-4:1]';              % Define a time vector
» V0=sin(10*t);               % and a "useful signal"
» noise=sin(370*t+1);         % which is corrupted by noise.
» Vs = V0+noise;              % This is what we measure
» plot(t,Vs)
»
»
»        % Simulate the filter response and compare the result
»        % with the original signal V0
» Vout=lsim([L*C*R3,C*R2*R3,R3],[L*C*(R1+R3),(C*R2*(R1+R3)+L),R1+R2+R3],Vs,t);
» plot(t,Vout,t,V0)
»
»        % The result is only a little less noisy than Vs
```



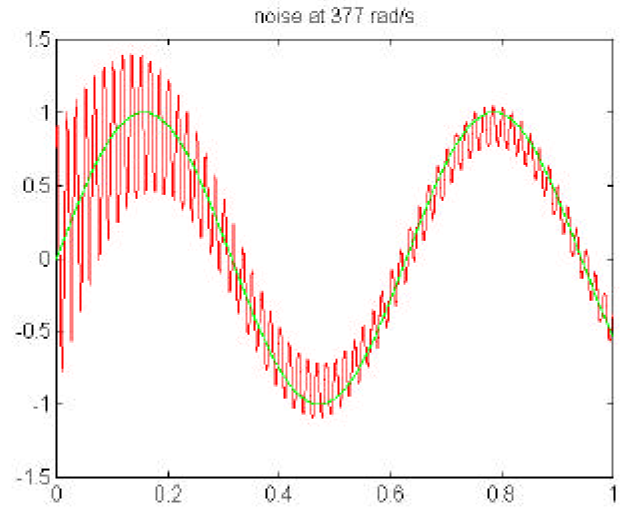

NOTE: For a brief explanation and calling format of Matlab commands use "help", e.g.,  >> help lsim

# Matlab Simulation




For a noise frequency 377 (rad/s):

```
» noise=sin(377*t+1);       % New noise.
» Vs = V0+noise;            % This is what we measure
» plot(t,Vs)
»
»
»
»          % Simulate the filter response and compare the result
»          % with the original signal V0
» Vout=lsim([L*C*R3,C*R2*R3,R3],[L*C*(R1+R3),(C*R2*(R1+R3)+L),R1+R2+R3],Vs,t);
» plot(t,Vout,t,V0)
»
»          % Impressive! The filter is very selective.
```



# Matlab Simulation

Finally, let's try to predict how much the noise is attenuated using AC analysis.
E.g, filtered noise = G(j370)*input noise
We need to use the concept of linearity, that is filtered (signal+noise) = filtered(signal)+filtered(noise)

```
» s=j*10
» G=(s.*s*L*C*R3+s*C*R2*R3+R3)./(s.*s*L*C*(R1+R3)+s*(C*R2*(R1+R3)+L)+R1+R2+R3)
G =
   0.9090 - 0.0006i
```

Use abs and angle to get the polar form: $G(j10) = 0.909\angle 0.0006$

And for s = j*377, we get G = 0.1232 + 0.0170i: $G(j377) = 0.124\angle 0.137$ (angles in rad)
So, in our phasor notation,

$$
\begin{aligned}
V_{out} &= G(j10)V_0 + G(j377)noise \\
&= [0.909\angle 0.0006][1\angle -1.57]|_{w=10} + [0.137\angle 0.124][1\angle -0.57]|_{w=377} \\
&= [0.909\angle -1.5694]|_{w=10} + [0.137\angle -0.446]|_{w=377}
\end{aligned}
$$

$$V_{out}(t) = 0.909\cos(10t - 1.57) + 0.137\cos(377t - 0.446)$$

The noise amplitude was reduced by a factor of 7, but in the process, 10% of the signal was lost too.
Notice that this expression is the steady-state voltage out, while in our previous simulation the plot includes the initial transient, lasting approximately 0.5 (s).