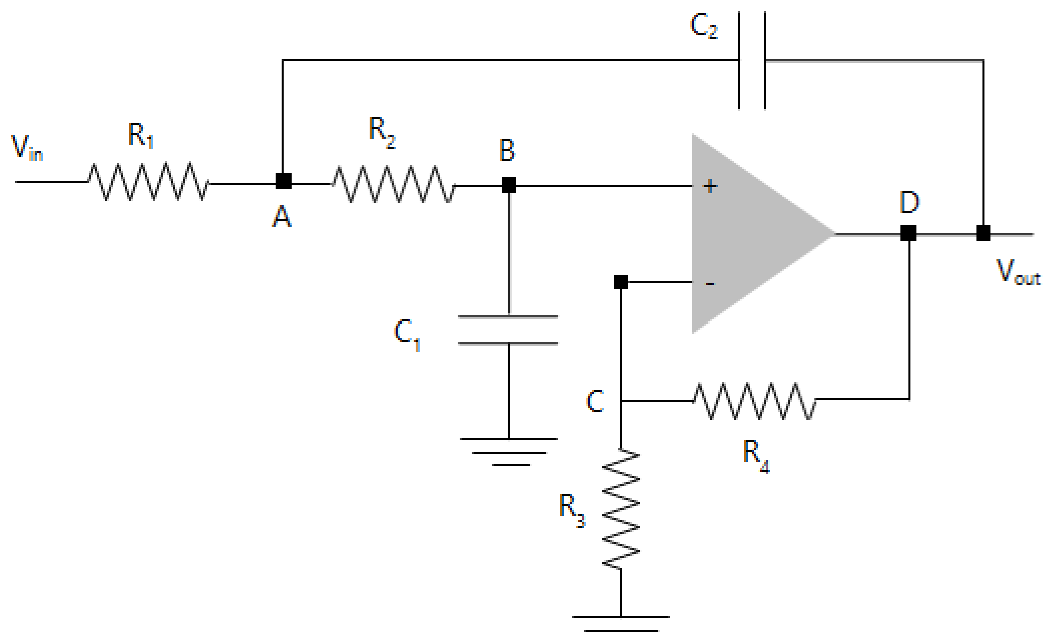
Sensitivity Analysis of a Sallen-Key Low-Pass Filter

This application performs a worst case circuit analysis of this filter circuit.



Specifically, the application

- derives a transfer function describing the ratio of the output voltage to the input voltage
- for each component, calculates the partial derivatives of the transfer function
- generates two parameter sets, taking into account the sign of the partial derivatives
- for both parameter sets, plots the DC response, response at 1.5 kHZ, and a magnitude plot

Op amp parameters are taken from a data sheet for a ISL70444SEH op amp. The DC gain is 90 dB, so the open loop voltage gain A_{o1} is $10^{90/20} = 31623$

Parameters

The elements in this matrix contain the

- name of each component
- nominal parameter value
- lower tolerance value
- higher tolerance value

Resistors have a symmetric tolerance of 2%. Capacitors have an asymmetric tolerance of -19% to +15%.

$$\mathtt{data} := \begin{bmatrix} R_1 & 13.3\times10^{3} & -0.02 & 0.02 \\ R_2 & 41.2\times10^{3} & -0.02 & 0.02 \\ R_3 & 10\times10^{3} & -0.02 & 0.02 \\ R_4 & 56.9\times10^{3} & -0.02 & 0.02 \\ C_1 & 0.0068\times10^{-6} & -0.19 & 0.15 \\ C_2 & 0.0068\times10^{-6} & -0.19 & 0.15 \\ R_{in} & 10\times10^{3} & 0 & 0 \\ R_o & 60 & 0 & 0 \\ A_{ol} & 31623 & 0 & 0 \\ i_b & 650\times10^{-9} & -1 & 1 \\ i_{os} & 50\times10^{-9} & -1 & 1 \\ V_{os} & 0.5\times10^{-3} & -1 & 1 \end{bmatrix}$$

$$\mathtt{data} := \mathtt{convert}(\mathtt{data}, \mathtt{listlist})$$

Circuit Analysis

Apply Kirchoff's Current Law

$$eqIA := \frac{V_A - V_{in}}{R_1} + (V_A - V_D)\cdot C_2\cdot s + \frac{V_A - V_B}{R_2} = 0$$

$$eqIB := \frac{V_B - V_A}{R_2} + V_B\cdot C_1\cdot s + i_{os} - i_b = 0$$

$$eqIC := \frac{V_C}{R_3} + \frac{V_C - V_D}{R_4} - i_b - i_{os} + \frac{V_C - V_{os} - V_B}{R_{in}} = 0$$

$$eqID := \frac{V_D - V_C}{R_4} + \frac{V_D - A_{ol}\cdot(V_C - V_{os} - V_B)}{R_o} + (V_D - V_A)\cdot C_2\cdot s = 0$$

Symbolically solve for V_{out} in terms of V_{in}

$$sol := solve([eqIA, eqIB, eqIC, eqID], [V_A, V_B, V_C, V_D])$$

$$V_{out} := rhs(sol[1, 4])$$

Partial Derivatives of Output Voltage wrt Components

Generate a list of component names

```
varNames := [data[ .., 1][ ], s, V_in] = [R_1, R_2, R_3, R_4, C_1, C_2, R_in, R_o, A_ol, i_b, i_os, V_os, s, V_in]
```

Generate a list of component nominal values

```
varValues := [data[ .., 2][ ], 1, 0]
```

Generate a list of equations for the component nominal values

```
values_eq := [seq(varNames[i] = varValues[i], i = 1 ..14)]
```

Hence the partial derivatives with respect to each component, evaluated at the nominal value.

```
derivs := [seq(eval(diff(V_out, varNames[j]), values_eq), j = 1 ..14)]
```

If the partial derivative is positive, V_{out} increases if the component value increases. However, if the partial derivative is negative, V_{out} decreases if the component value increases.

Hence a combination of the lower and upper tolerances determines the minimum and maxiumum values of V_{out}.

Given the upper and lower tolerances, we will now generate two parameters sets that describe the worst case behaviour.

For each component, the sign of the partial derivative determines if a parameter set contains the lower

```
lower_tol := data[ .., 3]          upper_tol := data[ .., 4]
```

If a component's partial derivative is positive, multiply the nominal value by the lower tolerance (or the upper tolerance otherwise)

```
pars_1 := [seq(varNames[i] = varValues[i]·(1 + ifelse(derivs[i] > 0, lower_tol[i],
upper_tol[i])), i = 1 ..12)]
```

If a component's partial derivative is positive, multiply the nominal value by the upper tolerance (or the lower tolerance otherwise)

```
pars_2 := [seq(varNames[i] = varValues[i]·(1 + ifelse(derivs[i] > 0, upper_tol[i],
lower_tol[i])), i = 1 ..12)]
```

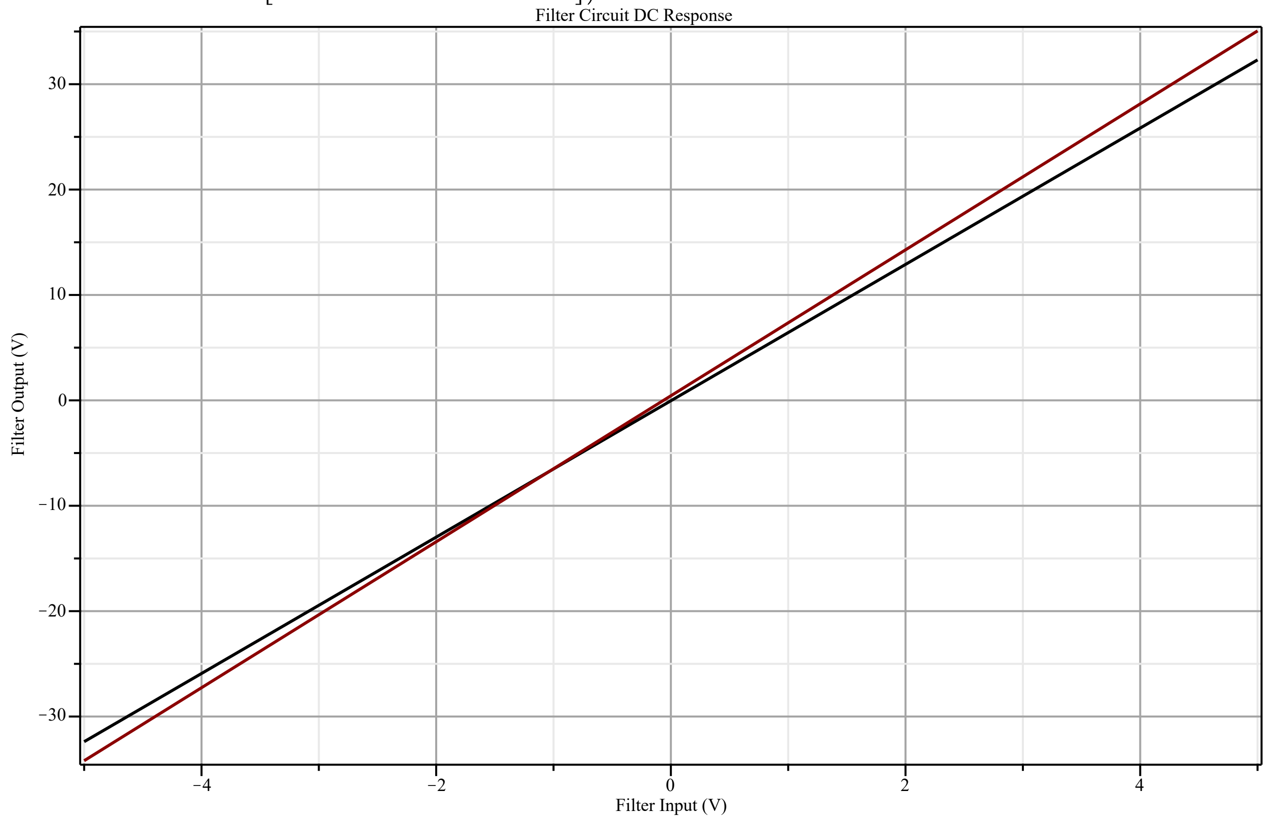
DC Response

Set s = 0 to calculate the DC response over a range of V_{in}

```
p1 := plot(eval(V_out, [pars_1[ ], s = 0]), V_in = -5 .. 5, color = black)
```

```
p2 := plot(eval(V_out, [pars_2[ ], s = 0]), V_in = -5 .. 5, color = "DarkRed")
```

```
plots:-display(p1, p2, axes = box, gridlines, title = "Filter Circuit DC Response",
labels = ["Filter Input (V)", "Filter Output (V)"],
labeldirections = [horizontal, vertical]) =
```

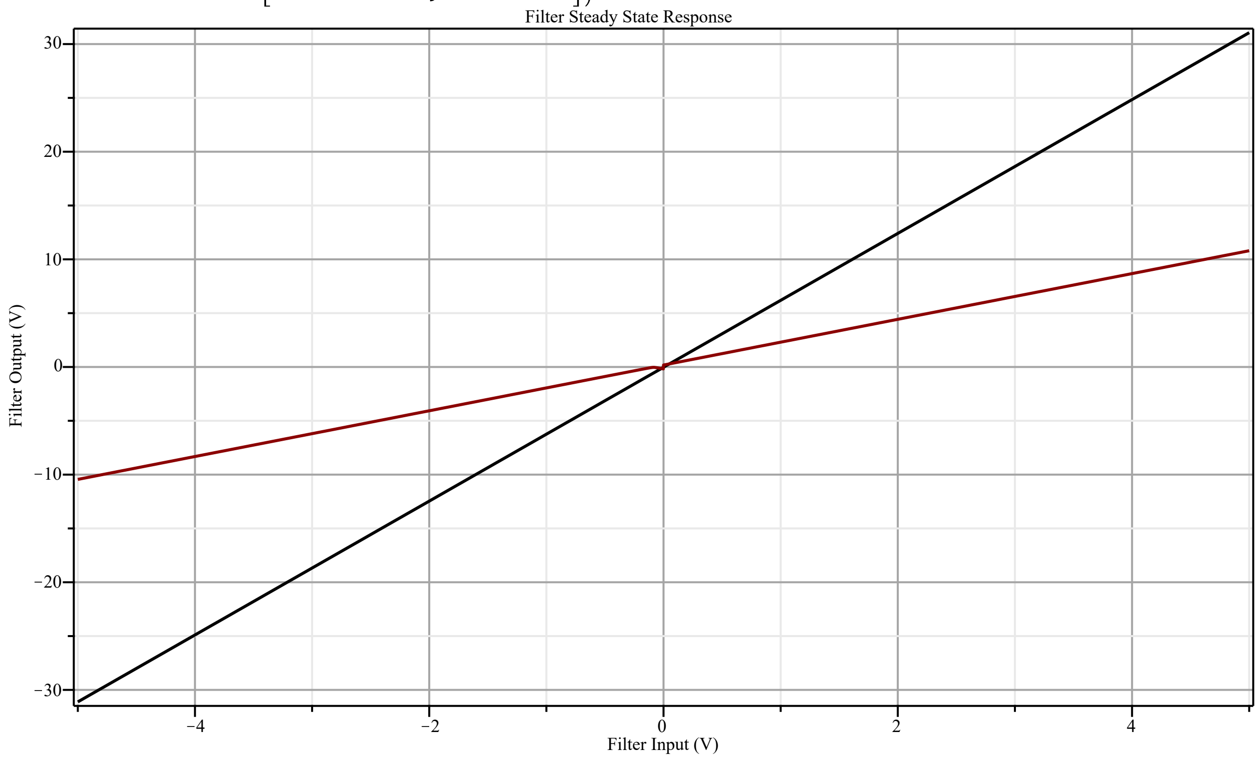


Steady-state Response at 1.5 kHz

```
p1 := plot(ifelse(V_in < 0, -1, 1)·abs(eval(V_out, [pars_1[ ], s = 2·π·1 i·1500])),
V_in = -5 .. 5, color = black)
```

```
p2 := plot(ifelse(V_in < 0, -1, 1)·abs(eval(V_out, [pars_2[ ], s = 2·π·1 i·1500])), V_in
= -5 .. 5, color = "DarkRed")
```

```
plots:-display(p1, p2, color = red, axes = box, gridlines,
 title = "Filter Steady State Response",
labels = ["Filter Input (V)", "Filter Output (V)"],
labeldirections = [horizontal, vertical]) =
```



Cutoff Frequency

A magnitude plot demonstrates how the cutoff frequency varies with the two worst case parameter sets

tf := DynamicSystems:-TransferFunction$\left(\frac{V_{out}}{V_{in}}\right)$

p1 := DynamicSystems:-MagnitudePlot(tf, parameters = [pars_1[ ], V_{in} = 5], range = 10 ..10^4, hertz, color = black)

p2 := DynamicSystems:-MagnitudePlot(tf, parameters = [pars_2[ ], V_{in} = 5], range = 10 ..10^4, hertz, color = "DarkRed")

```
plots:-display(p1, p2, title = "Filter Frequency Response") =
```