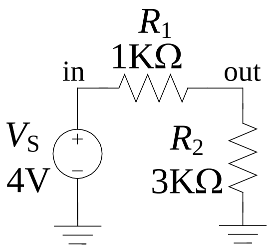
MONTE-CARLO ANALYSIS – SIMPLE EXAMPLE

Dr. J. E. Rayas-Sánchez

1. Generate conventional SPICE file




```
Simple Voltage Divider
*-----------------------
Vs   in  0   DC 4V
R1   in  out 1K
R2   out 0   3K

.control
op
print v(in) v(out) i(Vs)
.endc
.end
```

2. Re-direct output to a csv file

```
Simple Voltage Divider
*-----------------------
Vs   in  0   DC 4V
R1   in  out 1K
R2   out 0   3K

.control
op
write  VoltDiv_op_out.csv v(in) v(out) i(Vs)
.endc
.end
```

3. Generate a parameterized Matlab file to drive SPICE simulation

```
% ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
%                     Driving VoltDiv_m.cir from Matlab
%
% This function drives the WinSpice file VoltDiv_m.cir from Matlab, and
% returns the simulation results in cell arrays psi and R.
%
% Usage: [psi,R] = VoltDiv_SPICE1(x)
%         psi: cell array containing the independent variables (if any).
%         R: cell array containing the circuit responses
%         x: vector of selected parameterized elements
% Function required: None.

function [psi,R] = VoltDiv_SPICE1(x)

% Define SPICE Script, ss
ss{1} = 'Simple Voltage Divider';
ss{2} = '* -----------------------------------------------------';
ss{3} = '* Dr. J.E. Rayas-Sanchez                  April 4, 2016';
ss{4} = '* -----------------------------------------------------';
ss{5} = '*                 Simple Voltage Divider';
ss{6} = ['Vs   in  0   DC ' num2str(x(1))];
```

```
ss{7} = ['R1 in out ' num2str(x(2))];
ss{8} = ['R2 out 0 ' num2str(x(3))];
ss{9} = '.control';
ss{10} = 'op';
ss{11} = 'write VoltDiv_OP.csv v(in) v(out) i(Vs)';
ss{12} = 'quit';
ss{13} = '.endc';
ss{14} = '.end';

% Save SPICE Script as a Circuit File in Matlab Working Directory
CircuitFileName = 'VoltDiv_m.cir';
ckt_file = char(ss);
[rows,~] = size(ckt_file);
fid = fopen(CircuitFileName,'w+');  % File identifier opened.
for i = 1:rows
   fprintf(fid, '%s', ckt_file(i,:)); % Save each row of ckt_file.
   fprintf(fid, '%s\r\n', '');
end
fclose(fid);  % File identifier closed.

% Run WinSpice Circuit File
ExecFile = 'C:\command_line_WinSpice\wspice3 ';
system([ExecFile CircuitFileName]);

% Read WinSpice Output Files
RespOP = csvread('VoltDiv_OP.csv',1,0); % Read OP responses.
psi{1} = [];
R{1} = RespOP;

% Erase WinSpice Output Files
delete VoltDiv_OP.csv;
```

4. Generate a Matlab file to test the driver

```
% Nominal Design
Vs = 4;
R1 = 1e3;
R2 = 3e3;
x = [Vs R1 R2];

% Calculating Responses
[psi,R] = VoltDiv_SPICE1(x);

% Reading DC Operating Point Response
OPresp = R{1};
Vin = OPresp(:,2);
Vout = OPresp(:,3);
Ivs = OPresp(:,4);

% Displaying Results
disp('DC Operating Point:')
disp(['Vin = ' num2str(Vin) ' V']);
disp(['Vout = ' num2str(Vout) ' V']);
disp(['I(Vs) = ' num2str(Ivs*1e3) ' mA']);
```

Runing with:

```
Vs = 4;
R1 = 1e3;
R2 = 3e3;

>>
DC Operating Point:
Vin = 4 V
Vout = 3 V
I(Vs) = -1 mA
```

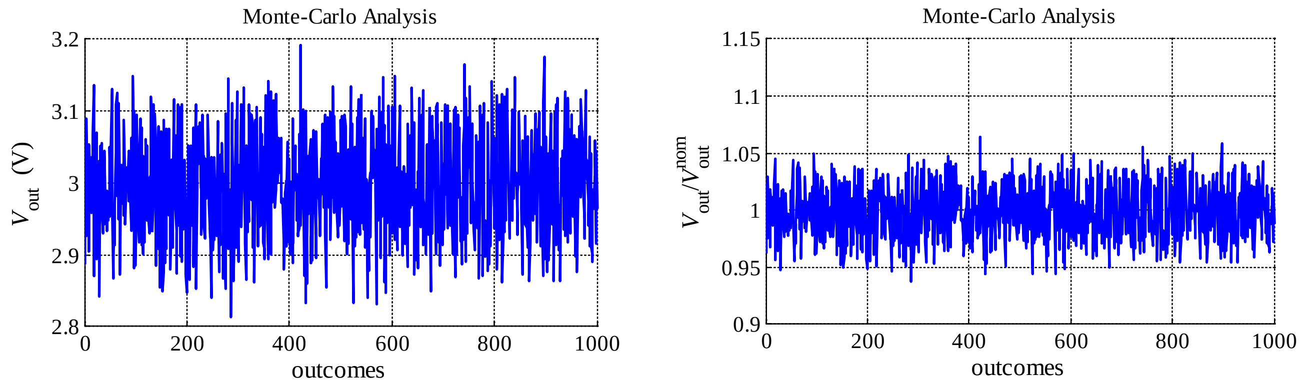
Runing with:

```
Vs = 5;
R1 = 2e3;
R2 = 1e3;

>>
DC Operating Point:
Vin = 5 V
Vout = 1.6667 V
I(Vs) = -1.6667 mA
```

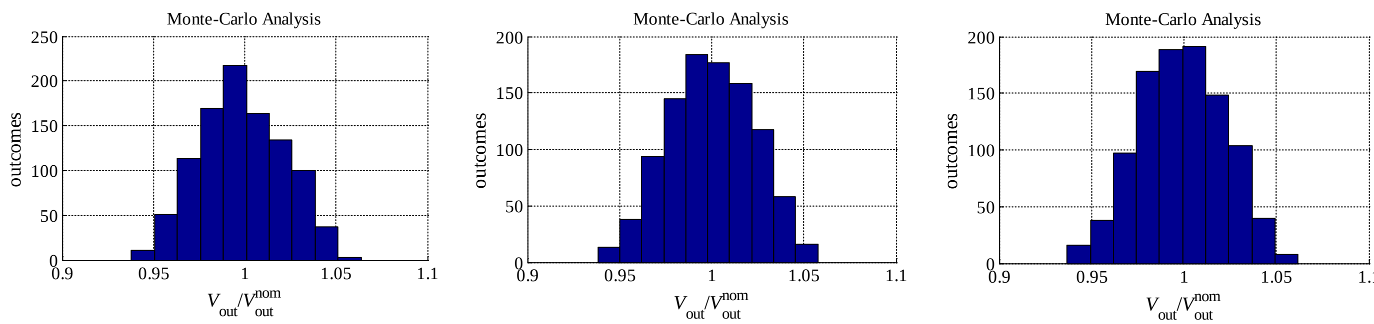
5. Generate a Matlab script to perform Monte-Carlo analysis

```
% Nominal Design
Vs = 4;
R1 = 1e3;
R2 = 3e3;
Ynom = [Vs R1 R2];

% Calculate Responses of Interest at Nominal Design
[psi,R] = VoltDiv_SPICE1(Ynom);
OPresp = R{1};
Vout_nom = OPresp(:,3);

% Define Outcomes and Tolerances
N = 300;      % Number of outcomes.
tauVs = 0.03; % Tolerance for Vs.
tauR1 = 0.05; % Tolerance for R1.
tauR2 = 0.1;  % Tolerance for R2.
tau = [tauVs tauR1 tauR2]; % Vector of tolerances.

% Generate Random Outcomes with Uniform PDF
Y = zeros(N,length(Ynom)); % Matrix to store outcomes.
for j = 1:N
    Y(j,1) = Ynom(1)*(1+tau(1)*(2*rand-1));
    Y(j,2) = Ynom(2)*(1+tau(2)*(2*rand-1));
    Y(j,3) = Ynom(3)*(1+tau(3)*(2*rand-1));
end

% Calculate Responses of Interest at Random Outcomes
Vout_rand = zeros(1,N);
for j = 1:N
    [psi,R] = VoltDiv_SPICE1(Y(j,:));
    OPresp = R{1};
    Vout_rand(j) = OPresp(:,3);
end

% Ploting Results ...
```

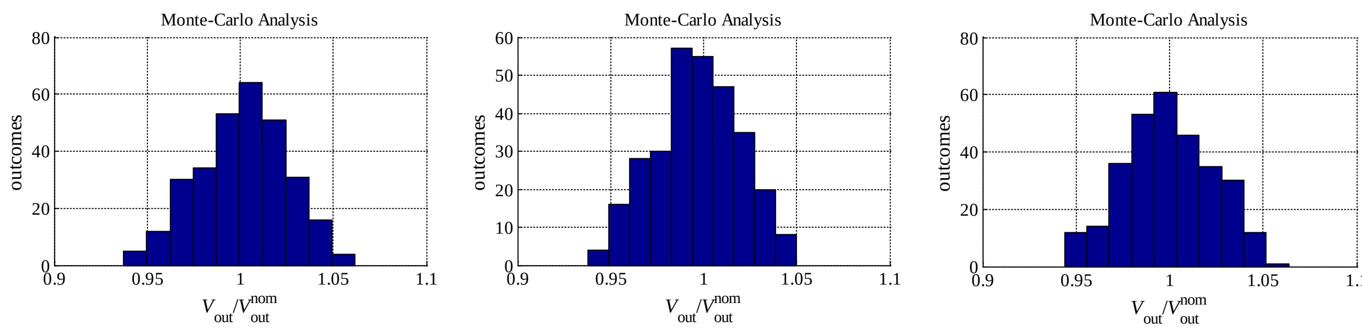
Using N = 1000:



Three runs using N = 1000:



Three runs using N = 300:



Three runs using N = 100: