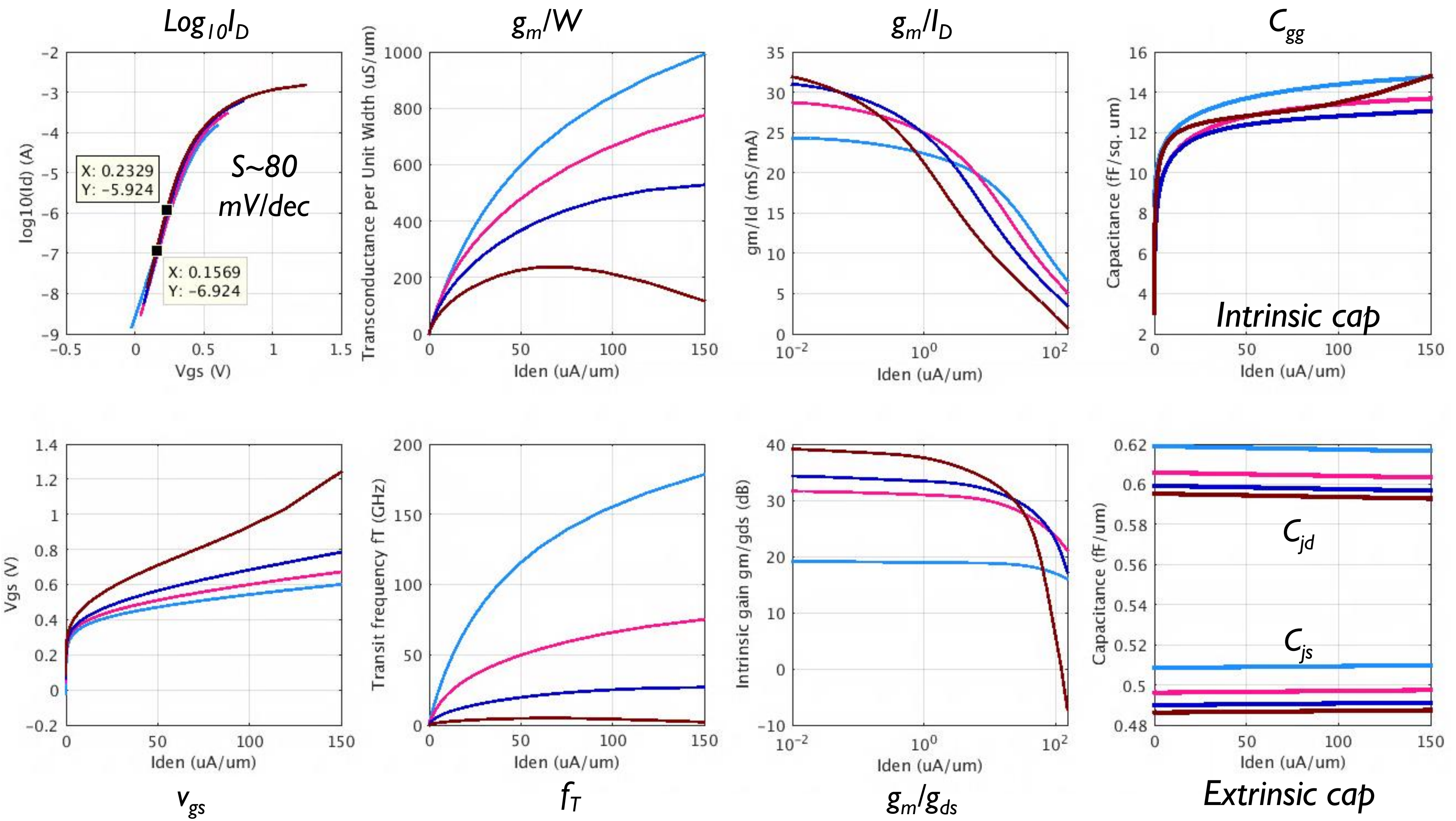
Transistor Characterization and Cadence/Matlab Scripting

Jeyanandh Paramesh

In class, we showed how to derive device characteristics (nch) using simple simulations. In this exercise, you will do the same for the transistors that you will be using for your designs in this class. You will do so using by using a combination of the ADE GUI, OCEAN scripts and Matlab scripts.



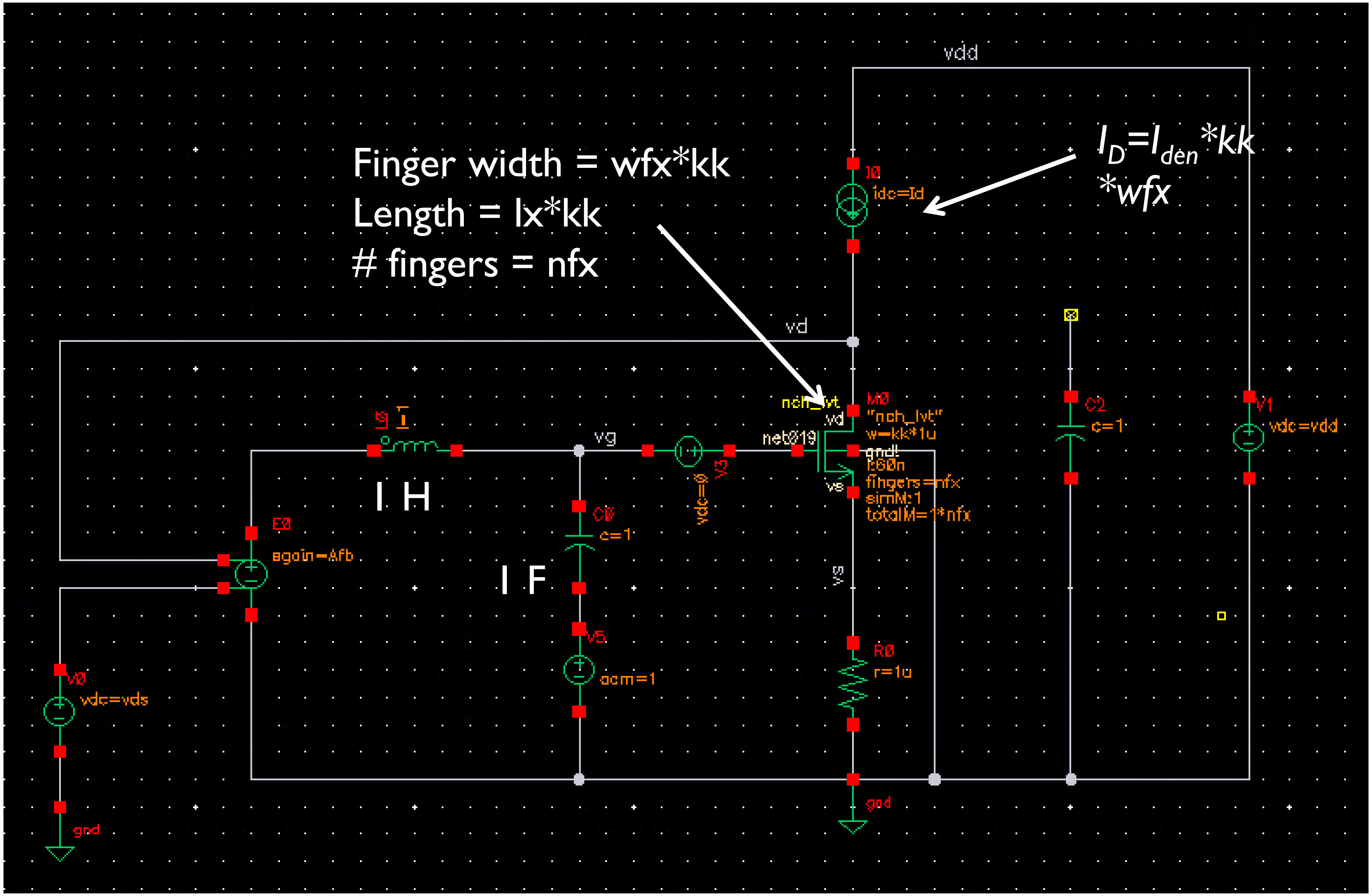
L = [1 2 4 10]*60 nm; v_{DS} = 0.4 V

Objectives

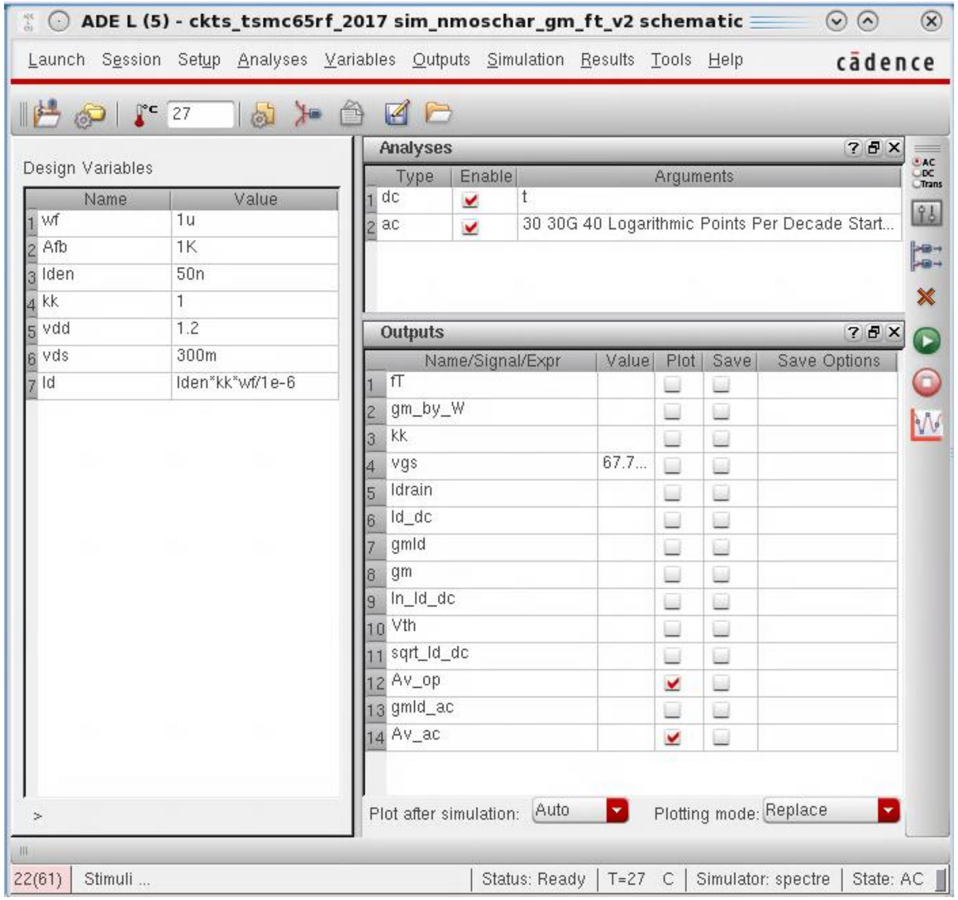
This exercise is particularly useful for three reasons:

- 1. In previous classes, transistor parameters were given to you in some form. In practice, you will have to do extract such parameters on your own by running simple simulations on stand-alone transistors under controlled bias conditions. This exercise shows you how to do this.
- 2. A good understanding of transistor operation and simulation models is extremely valuable when designing high-performance circuits. This exercise will give you an opportunity to explore transistor models.
- 3. Scripting in Cadence and using Matlab (or an equivalent) is a valuable skill. This exercise will give you exposure to such scripting.

Schematic Setup

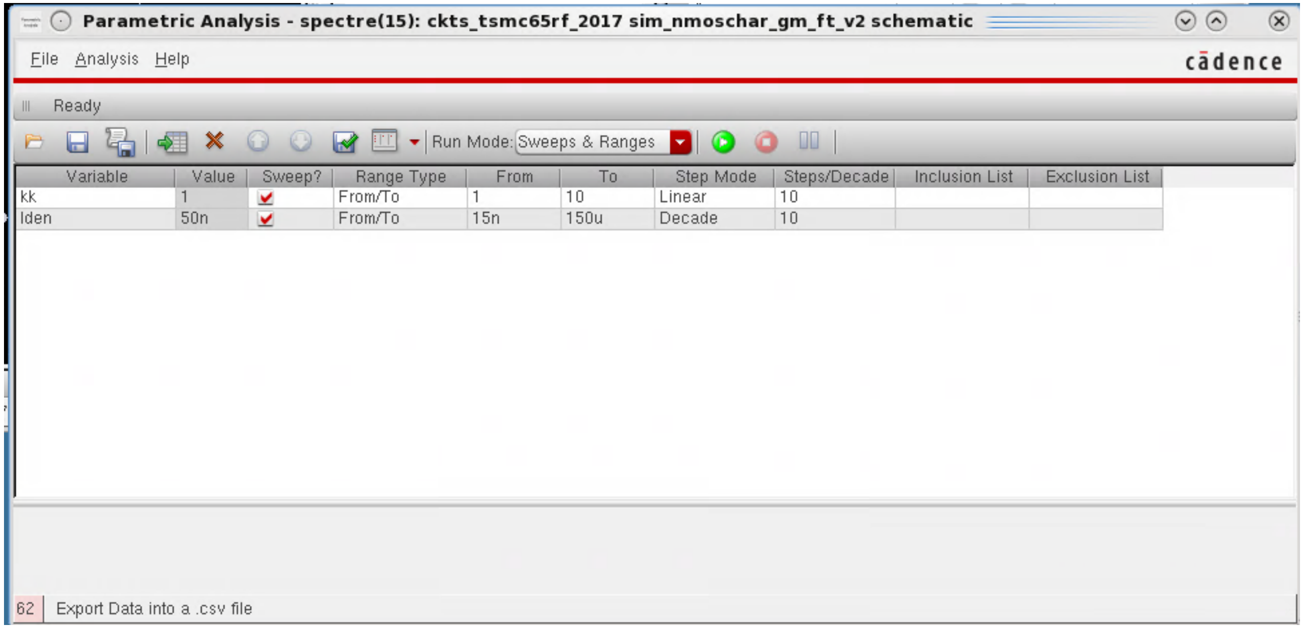


ADE Setup



- Add parameters in ADE
- Optionally, add outputs
  - You can play around with this. I use the calculator to compose outputs and then pull them into ADE
- Here, I have set up both DC and AC simulations
- Here, we will predominantly use DC results and so you can ignore the AC.

Parametric Analysis Setup



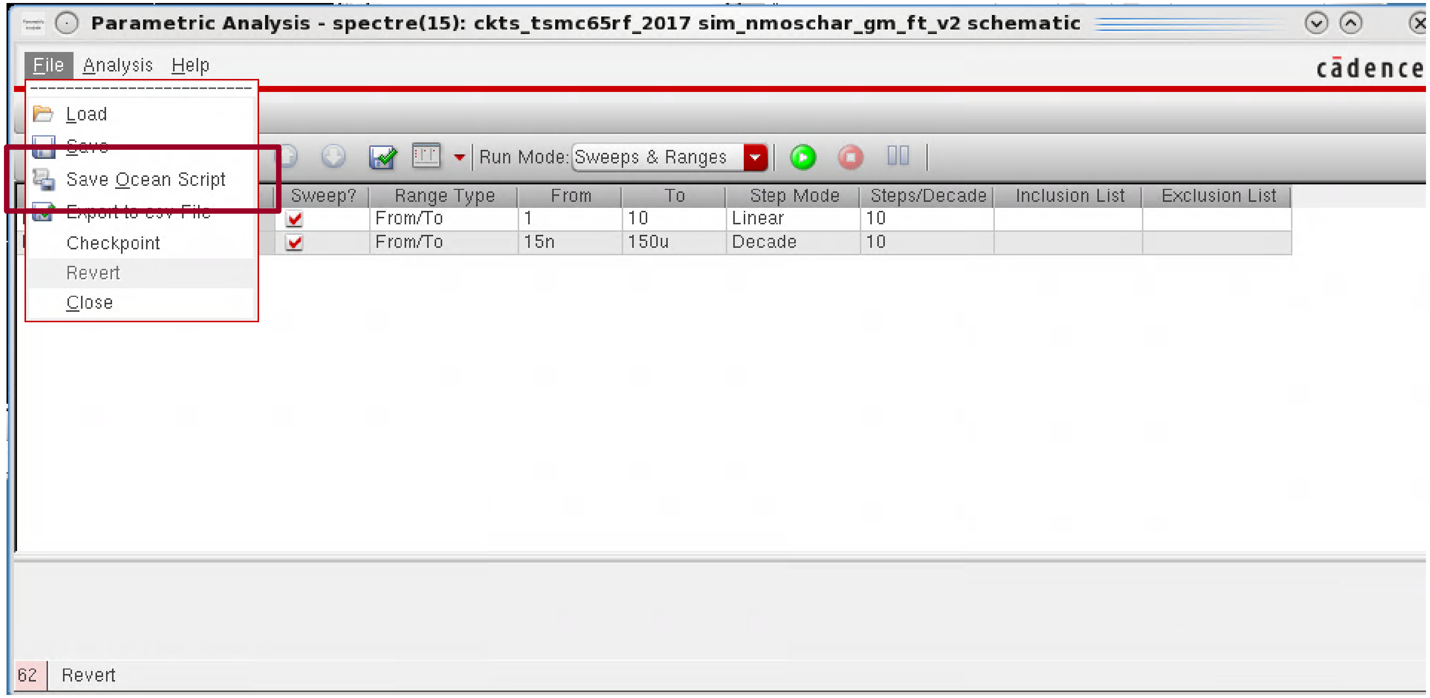
For every transistor length

For every value of current density

Run DC Op point analysis

- Run the parametric analysis once directly from ADE
  - Inspect a few outputs (Vds, I_D, gm using the calculator)
- Make sure they are ok

Ocean Script



- Save your simulation setup to an "Ocean" script (Note: Ocean is a scripting language in Cadence)

- **You can run any Ocean script from the Cadence Interface Window (CIW) by typing load "my_ocn.ocn"**

RAW Ocean Script (1)

These are model files, but for a different process from what you will be using in this class

emacs@rfic2.andrew.cmu

File Edit Options Buffers Tools He

```
simulator( 'spectre )
design(  "/home/scratch/paramesh/sim_nmosc[illegible]m_ft_v2/spectre/schematic/netlist/netlist")
resultsDir( "/home/scratch/paramesh/sim_nmosc[illegible]m_ft_v2/spectre/schematic" )
modelFile(
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_bip")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_dio")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_dio_33")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_mim")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_dio_dnw")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_dio_18")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_bip_npn")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_33")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_rfrtmom")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_mos_cap_25")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_18")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_disres")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_res")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_dio_na")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_rfmos_33")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_dio_hvt")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_rfmvar")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_rfmos_18")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_na")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_dio_na33")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_dio_lvt")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_rfres_sa")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_na33")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_dio_esd")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_rfmim")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_dio_25od33")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_25od33")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_mos_cap")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_dio_25")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_dio_25ud18")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_25")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_rfmos")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_25ud18")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_dio_na25od33")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_rfjvar")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_rfmos_25")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_rtmom")
--:**-  paramTool.ocn    Top L3     (Fundamental)----------------------------------------
```

RAW Ocean Script (2)

emacs@rfic2.andrew.cmu.edu

File Edit Options Buffers Tools Help

```
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_rfind")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_rfmvar_25")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_na25")
    '("/afs/ece.cmu.edu/project/rfic/DesignKits/TSMC/TSMC65/tsmcN65/../models/spectre/crn65gplus_2d5_lk_v1d0.scs" "tt_lvt")
)
analysis('dc ?saveOppoint t  )
analysis('ac ?start "30"  ?stop "30G"  ?dec "40"  )
desVar(   "wf" 1u       )
desVar(   "Afb" 1000    )
desVar(   "Iden" 50n    )
desVar(   "kk" 1        )
desVar(   "vdd" 1.2     )
desVar(   "vds" 0.3     )
desVar(   "Id" "Iden*kk*wf/1e-6"        )
envOption(
        'analysisOrder  list("dc" "ac")
)
option( 'gmin  "1e-15"
        'iabstol  "1e-14"
        'vabstol  "1e-8"
        'reltol  "1e-5"
)
saveOption( ?simOutputFormat "sst2" )
saveOption( 'useprobes "yes" )
saveOption( 'currents "all" )
converge( 'nodeset "/vd" "0.5" )
temp( 27 )
paramAnalysis("kk" ?values '(1 2 3 4 5 6 7 8 9 10 )
  paramAnalysis("Iden" ?values '(1.5e-08 1.88838811769125e-08 2.37733978869167e-08 2.99289347245332e-08 3.76782964726437e-08 4.743416490
25257e-08 5.97160755830246e-08 7.51780850440909e-08 9.46436016720291e-08 1.19149235208642e-07 1.5e-07 1.88838811769125e-07 2.37733978869
167e-07 2.99289347245332e-07 3.76782964726437e-07 4.74341649025257e-07 5.97160755830247e-07 7.51780850440909e-07 9.46436016720291e-07 1.
19149235208642e-06 1.5e-06 1.88838811769125e-06 2.37733978869167e-06 2.99289347245332e-06 3.76782964726438e-06 4.74341649025258e-06 5.97
160755830247e-06 7.5178085044091e-06 9.46436016720292e-06 1.19149235208643e-05 1.5e-05 1.88838811769126e-05 2.37733978869168e-05 2.99289
347245333e-05 3.76782964726438e-05 4.74341649025258e-05 5.97160755830247e-05 7.5178085044091e-05 9.46436016720292e-05 0.0001191492352086
43 0.00015 )
))
paramRun()
Av_ac = dB20(value(abs((VF("/vd") / VF("/vg"))) 10000.0))
plot( Av_ac ?expr '( "Av_ac" ) )
Av_op = db20((pv("M0" "gm" ?result "dcOpInfo") / pv("M0" "gds" ?result "dcOpInfo")))
plot( Av_op ?expr '( "Av_op" ) )
```

--:**- paramTool.ocn   Bot L75   (Fundamental)---------------

Analysis details

ADE parameters

This sometimes helps DC convergence

Length and Iden parametrization

ADE outputs

Modifying the script

```
Iden_list = '(1.5e-07 1.88838811769125e-07 2.37733978869167e-07
2.99289347245332e-07 3.76782964726437e-07 4.74341649025257e-07
5.97160755830246e-07 7.51780850440909e-07 9.46436016720291e-07
1.19149235208642e-06 1.5e-06 1.88838811769125e-06 2.37733978869167e-06
2.99289347245332e-06 3.76782964726437e-06 4.74341649025257e-06
5.97160755830247e-06 7.51780850440909e-06 9.46436016720291e-06
1.19149235208642e-05 1.5e-05 1.88838811769125e-05 2.37733978869167e-05
2.99289347245332e-05 3.76782964726438e-05 4.74341649025258e-05
5.97160755830247e-05 7.51780850440911e-05 9.46436016720292e-05
0.000119149235208643 0.00015 )

kk_list = '(1 1.25 1.5 2 2.5 3 4 5 6 7)

;;Iden_list = '(1e-6 10e-6)
;;kk_list = '(7)

fp = outfile("/afs/ece.cmu.edu/usr/paramesh/cadence_afs/tsmc65RF/pch_lvt_char.csv",
"w" )
fprintf(fp,"W,L,Id,Vgs,Vds,Vth,gm,gds,Cgg,Cgs,Csg,Cgd,Cdg,Cjd,Cjs,gmb,region\n");

k1 = 0
foreach( a1 kk_list
        kkx = nth(k1,kk_list)
        sprintf( kkx, "%g" kkx*1.0 )
        desVar( "kk",kkx )
        k2 = 0
        foreach( a2 Iden_list
                printf("k1 = %d\tk2 = %d\n" k1 k2)
                Idenx = nth(k2,Iden_list)
                sprintf( Idenx "%g" Idenx*1.0 )
                desVar( "Iden",Idenx )
                run()

                wx = pv("M0" "w" ?result "instance")
                lx = pv("M0" "l" ?result "instance")
```

List of current density values

Open file to write information to. We will use csv format

Write header of tabulated values

Extract width and length info of transistor from simulation output. This is only for error checking.

- I usually modify the RAW ocean script to directly extract the information I need
- Instead of a parametric analysis, I put the run() statement in a for loop as shown

Modifying the script

- Hot tip: How do I find the syntax of a command? For example, the pv command?
- Simple.
  - I have run the simulation once previously from the ADE GUI.
  - Then I open the "results browser"
  - Then I find the parameter I want to extract in the GUI, and send it to the calculator
  - The command appears in the calculator, and I can copty/paste if from there into the OCEAN script

Extract all parameter values from simulation output.

```
Id_dc = -pv("M0" "id" ?result "dcOpInfo")

vgs = -pv("M0" "vgs" ?result "dcOpInfo")
vds = -pv("M0" "vds" ?result "dcOpInfo")
Vth = -pv("M0" "vth" ?result "dcOpInfo")

gm = pv("M0" "gm" ?result "dcOpInfo")
gmb = pv("M0" "gmb" ?result "dcOpInfo")
gds = pv("M0" "gds" ?result "dcOpInfo")

Cgg = pv("M0" "cgg" ?result "dcOpInfo")
Csg = (-1 * pv("M0" "csg" ?result "dcOpInfo"))
Cdg = (-1 * pv("M0" "cdg" ?result "dcOpInfo"))
Cgd = (-1 * pv("M0" "cgd" ?result "dcOpInfo"))
Cgs = (-1 * pv("M0" "cgs" ?result "dcOpInfo"))

Cjd =pv("M0" "cjd" ?result "dcOpInfo");
Cjs =pv("M0" "cjs" ?result "dcOpInfo");
region = pv("M0" "region" ?result "dcOpInfo")

printf("vgs=%g\n" vgs)

                fprintf(fp,"%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,",wx lx Id_dc vgs
vds Vth gm gds Cgg Cgs Csg Cgd Cdg)
                        fprintf(fp,"%g,%g,%g,%d\n",Cjd,Cjs,gmb,region)
                        k2 = k2+1
                )
                k1 = k1+1
)

close(fp)
```

Write all extracted values to file.

Run simulation for next length value

Processing the data in Matlab

- You will need the rgb and loadcolors scripts in the following Matlab script.
- You can find them in the class afs folder

Processing the data in Matlab

- I read the csv file into Matlab to plot the results

Modify this path. This should point to where the scripts “rgb” and “loadcolors” are stored.

Open csv file where you wrote the sim data

Should have same format as the data you wrote into the csv file

```
clear
close all
addpath('/afs/ece.cmu.edu/usr/paramesh/Matlab/Utilities')
loadcolors;
numcolors = length(colornames);
dbm = inline('10*log10(x/1e-3)', 'x');

fid = fopen('/afs/ece/usr/paramesh/Matlab/TransistorChar/TransistorData/tsmc65_nch_rvt_v2.csv');
data1 = textscan(fid, '%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f', 'Delimiter', ',' ,'HeaderLines', 1);

widths  = data1{1};
lengths = data1{2};
devData = cell2mat( data1(3:16) );

[wids, nval1] = unique(widths);
[lens, nval2] = unique(lengths);
if ( ( length(wids) ~= length(lens) ) && ( nval1 ~= nval2 ))
    disp('Something wrong with input data');
end
nvalx = nval1;
nvalx = [nval1; length(devData)+1];

numDevices = length(wids);
```

Processing the data in Matlab

```
for k = 1:numDevices
    device_nch(k).Type = 'nch';
    device_nch(k).W = wids(k);
    device_nch(k).L = lens(k);
    device_nch(k).data = devData( nvalx(k):nvalx(k+1)-1, : );
end

hh = figure(1)
hh.Color = [1 1 1];
xfig = 0; yfig = 0;
set(hh, 'Position', [xfig, yfig, 1600, 750]);

%for k = 1:numDevices
for k = 1:3:10
    idx =  device_nch(k).data(:,1) ;
    vgsx = device_nch(k).data(:,2);
    vdsx = device_nch(k).data(:,3);
    Vthx = device_nch(k).data(:,4);
    gmx =  device_nch(k).data(:,5);
    gdsx =  device_nch(k).data(:,6);
    Cggx = device_nch(k).data(:,7);
    Cgsx = device_nch(k).data(:,8);
    Csgx = device_nch(k).data(:,9);
    Cgdx = device_nch(k).data(:,10);
    Cdgx = device_nch(k).data(:,11);
    Cjdx = device_nch(k).data(:,12);
    Cjsx = device_nch(k).data(:,13);
```

I am plotting data for all device lengths here. You can easily modify this to plot data only for device lenghts you are interested in.

Open figure and set size and position

Read all parameters from csv file.

Processing the data in Matlab

```
    wx = device_nch(k).W;
    lx = device_nch(k).L;
    iden = idx/wx;
    kx = k+8;


figure(1)

    subplot(241), h0 = plot(vgsx, log10(idx)); hold on; grid on;
    set(h0, 'Color',rgb(colornames{mod(kx,numcolors)+3}), 'LineWidth',2, 'LineStyle', '-');
    ylabel('log10(Id) (A)'), xlabel('Vgs (V)'); grid on;
    %axis([-0.5 1.5 -36 -6])

    N_vgsx = length(vgsx);
    S_subthreshold = 1./(diff(log10(idx))./diff(vgsx));
    subplot(245),
    h0 = plot(iden,vgsx);
     ylabel('Vgs (V)'), xlabel('Iden (uA/um)'); grid on;
          hold on; grid on;
    set(h0, 'Color',rgb(colornames{mod(kx,numcolors)+3}), 'LineWidth',2, 'LineStyle', '-');

    subplot(242), h0 = plot(iden,gmx./wx); hold on; grid on;
    set(h0, 'Color',rgb(colornames{mod(kx,numcolors)+3}), 'LineWidth',2, 'LineStyle', '-');
    ylabel('Transconductance per Unit Width (uS/um)'), xlabel('Iden (uA/um)');

    subplot(243), h0 = semilogx(iden,gmx./idx); hold on; grid on;
    set(h0, 'Color',rgb(colornames{mod(kx,numcolors)+3}), 'LineWidth',2, 'LineStyle', '-');
    ylabel('gm/Id (mS/mA)'), xlabel('Iden (uA/um)'); grid on;
    xlim([1.0e-2 1.5e2]);
```

Plot everything.

Processing the data in Matlab

- You can use this code to learn how to make professional-looking graphs in Matlab.
- See how all axes are labeled, label fonts sized to be readable etc.

```
subplot(247), h0 = semilogx(iden,20*log10(gmx./gdsx)); hold on; grid on;
set(h0, 'Color',rgb(colornames{mod(kx,numcolors)+3}), 'LineWidth',2, 'LineStyle', '-');
ylabel('Intrinsic gain gm/gds (dB)'), xlabel('Iden (uA/um)'); grid on;
 xlim([1.0e-2 1.5e2]);

subplot(246), h0 = plot(iden,gmx./(Cggx*2*pi)/1e9); hold on; grid on;
set(h0, 'Color',rgb(colornames{mod(kx,numcolors)+3}), 'LineWidth',2, 'LineStyle', '-');
ylabel('Transit frequency fT (GHz)'), xlabel('Iden (uA/um)'); grid on;

K=1e3;
subplot(244)
h0 = plot(iden,K*Cggx/(wx*lx)); hold on;
set(h0, 'Color', rgb(colornames{mod(kx,numcolors)+3}), 'LineWidth',3, 'LineStyle', '-');
ylabel('Capacitance (fF/sq. um)'), xlabel('Iden (uA/um)'); grid on;
```

Processing the data in Matlab

```
 K2 = 1e-9;
    subplot(248)
    h0 = plot( iden,Cjdx/K2/wx, iden,Cjsx/K2/wx ); hold on;
    set(h0, 'Color', rgb(colornames{mod(kx,numcolors)+3}), 'LineWidth',3, 'LineStyle', '-');
    ylabel('Capacitance (fF/um)'), xlabel('Iden (uA/um)'); grid on;


end

lx = []; Vthx = [];
for k = 1:numDevices
    lx(k) = device_nch(k).L;
    Vthx(k) = mean(device_nch(k).data(:,4));
end

hh = figure(2)
hh.Color = [1 1 1];
h0 = plot(lx/1e-6, Vthx); grid on;
set(h0, 'LineWidth',3, 'LineStyle', '-');
ylabel('Threshold voltage (V)'), xlabel('Length (um)'); grid on;

rmpath('/afs/ece.cmu.edu/usr/paramesh/Matlab/Utilities');
```