Scilab Textbook Companion for Radio Frequency Circuit Design by R. Ludwig And G. Bogdanov[1]

Created by
Monisha Dhirmalani
Engineering
Others
Mumbai University
College Teacher
Monali Chaudhari
Cross-Checked by

May 20, 2016

---

[1] Funded by a grant from the National Mission on Education through ICT, http://spoken-tutorial.org/NMEICT-Intro. This Textbook Companion and Scilab codes written in it can be downloaded from the "Textbook Companion Project" section at the website http://scilab.in

Book Description

Title: Radio Frequency Circuit Design

Author: R. Ludwig And G. Bogdanov

Publisher: Pearson Education India, New Delhi

Edition: 2

Year: 2011

ISBN: 978-81-317-6218-9

Scilab numbering policy used in this document and the relation to the above book.

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

AP Appendix to Example(Scilab Code that is an Appednix to a particular Example of the above book)

For example, Exa 3.51 means solved example 3.51 of this book. Sec 2.3 means a scilab code whose theory is explained in Section 2.3 of the book.

Contents

List of Scilab Codes

List of Figures

Chapter 1

Introduction

Scilab code Exa 1.1 Intrinsic wave impedance

```
mu0=4*%pi*10^-7;// defining permeability of free
   space
epsilon0=8.85*10^-12;// defining permittivity of
   free space
z0=sqrt(mu0/epsilon0);// calculating intrinsic
   impedance
epsilonr=4.6;// defining relative permittivity
vp=1/sqrt(mu0*epsilon0*epsilonr);// calculating
   phase velocity
f1=30*10^6;
f2=3*10^9;
lambda1=vp/(f1);
lambda2=vp/(f2);
disp('metre',lambda1,'Wavelength corresponding to f1
   ');//displaying wavelengths
disp('metre',lambda2,"Wavelength corresponding to f2
   ");//displaying wavelengths
```

Scilab code Exa 1.2 Comparing Inductances at different frequencies

```
mu0=4*%pi*10^-7;
a=8*2.54*10^-5; //radius of copper wire
sigmac=64.5*10^6; //conductivity of copper
l=2*10^-2; //length of wire
rdc=l/(%pi*a*a*sigmac);
f1=100*10^6;
f2=2*10^9;
f3=5*10^9;
skindepth1=1/sqrt(%pi*mu0*f1*sigmac);
skindepth2=1/sqrt(%pi*mu0*f2*sigmac);
skindepth3=1/sqrt(%pi*mu0*f3*sigmac);
Lin1=(a*rdc)/(2*skindepth1*2*%pi*f1); //internal
   inductance
Lin2=(a*rdc)/(2*skindepth2*2*%pi*f2); //internal
   inductance
Lin3=(a*rdc)/(2*skindepth3*2*%pi*f3); //internal
   inductance
temp=log(2*l/a)/log(%e);
Lex=mu0*l*(temp-1)/(2*%pi); //external inductance
disp("metre",skindepth1,"Skin depth at f1");
disp("metre",skindepth2,"Skin depth at f2");
disp("metre",skindepth3,"Skin depth at f3");
disp("Henry",Lin1,"Internal inductance at f1");
disp("Henry",Lin2,"Internal inductance at f2");
disp("Henry",Lin3,"Internal inductance at f3");
disp("Henry",Lex,"External inductance");
```

Scilab code Exa 1.3 Frequency response of high frequency resistor

```
f=10^4:10^5:10^10;
w=2*%pi.*f;
mu0=4*%pi*10^-7;
l=2*2.5*10^-2;
```

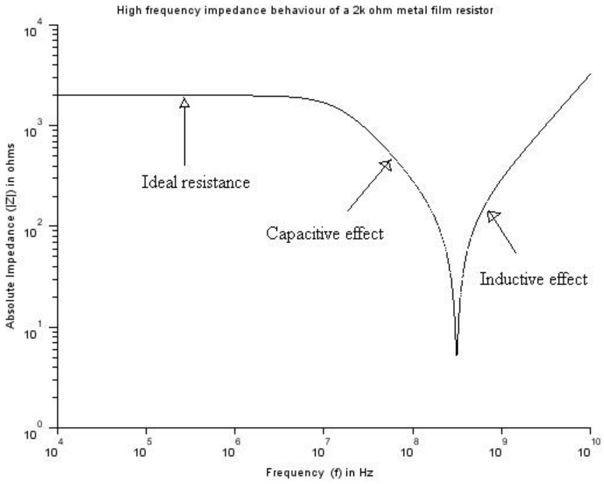

Figure 1.1: Frequency response of high frequency resistor

```
a=2.032*10^-4;
temp=log(2*l/a)/log(%e);
lex=mu0*l*(temp-1)/(2*%pi); //external inductance
r=2*10^3;                   // resistance
c=5*10^-12;                 //capacitance
z=w*lex*%i+1 ./(w*c*%i+1/r); //impedance
plot2d("gll",f,abs(z));
title("High frequency impedance behaviour of a 2k
   ohm metal film resistor ");
xlabel('Frequency  (f) in Hz');
ylabel('Absolute Impedance (|Z|) in ohms');
```

Scilab code Exa 1.4 Frequency response of high frequency capacitor

```
f=10^6:10^7:10^10;
rs=(4.8*10^-6).*sqrt(f);
```

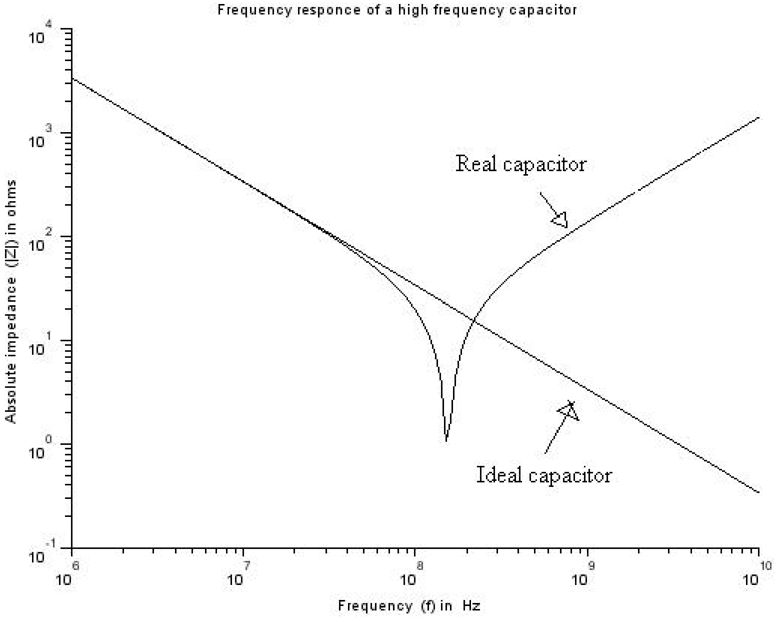

Figure 1.2: Frequency response of high frequency capacitor

```
re=(33.9*10^12) ./f;
c=47*10^-12;
w=2*%pi.*f;
l=2*1.25*10^-2;
a=2.032*10^-4;
temp=log(2*l/a)/log(%e);
lex=mu0*l*(temp-1)/(2*%pi);          //external inductance
z=1 ./(1 ./re +w*c*%i)+rs+w.*lex*%i; // impedance of frequency dependent capacitor
zideal=1 ./(w*c*%i);    //impedance of an ideal capacitor
plot2d("gll",f,abs(z));
plot2d(f,abs(zideal));
title("Frequency responce of a high frequency capacitor");
xlabel('Frequency  (f) in  Hz');
ylabel('Absolute impedance  (|Z|) in ohms');
```

Scilab code Exa 1.5 frequency response of high frequency inductor

```
f=10^7:10^8:10^10;
w=2*%pi.*f;
N=3.5;        //number of turns
rad=0.05*0.0254;
len=0.05*0.0254;       //length of wire
a=(5*0.0254*10^-3)/2;
u0=4*%pi*10^-7;
sig_cu=64.516*10^6;
e0=8.854*10^-12;
l=(%pi*rad^2*u0*(N^2))/len;
c=(e0*4*%pi*rad*(N^2)*a)/len;
r=(2*rad*N)/(sig_cu*(a^2));
```

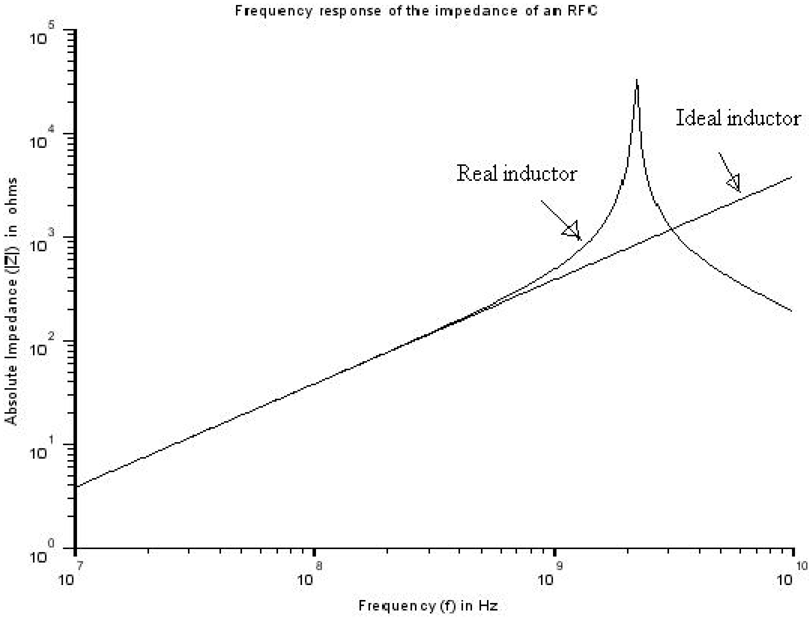

Figure 1.3: frequency response of high frequency inductor

```
z=1 ./((1 ./(r+w*%i*l))+w*%i*c); //impedance
zideal=w*%i.*l;                  //impedance of an ideal inductor
plot2d("gll",f,abs(z));
plot2d(f,abs(zideal));
title("Frequency response of the impedance of an RFC");
xlabel('Frequency (f) in Hz');
ylabel('Absolute Impedance (|Z|)  in  ohms');
```

Chapter 2

Transmission line analysis

Scilab code Exa 2.1 Magnetic field inside and outside infinitely long current carrying wire

```
I=5; //current in infinitely long wire
a=0.005; //radius of infinitely long wire
r_max=10*a;
N=100;
r=(0:N)/N*r_max;
for k=1:N+1
if(r(k)<=a)
H(k)=I*r(k)/(2*%pi*a*a);
else
H(k)=I/(2*%pi*r(k));
end;
end;
plot(r*1000,H);
plot([a a]*1000,[0 160],’r:’);
title(”Magnetic field distribution vs. distance from the center”);
xlabel(”Distance from the center of the wire ,mm”);
ylabel(”Magnetic field ,A/m”);
```

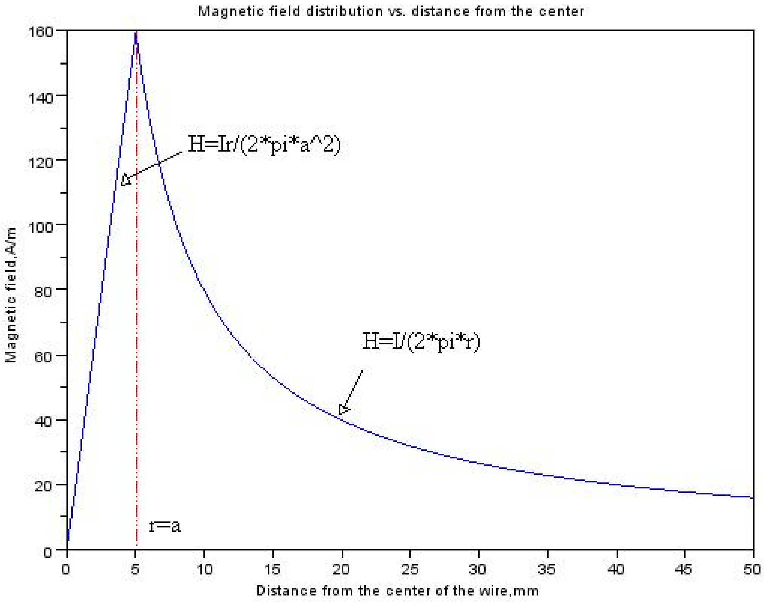

Figure 2.1: Magnetic field inside and outside infinitely long current carrying wire

---

Scilab code Exa 2.3 Transmission line parameters of a parallel copper plate transmission line

```
f=1*10^9;
w=6*10^-3; //width
d=1*10^-3; //seperation
epsilonr=2.25;
epsilon0=8.85*10^-12;
sigma_diel=0.125;
sigma_cond=64.5*10^6;
mu0=4*%pi*10^-7;
skindepth=1/sqrt(%pi*sigma_cond*mu0*f);
r=2/(w*sigma_cond*skindepth);
L=2/(w*sigma_cond*2*%pi*f*skindepth);
c=epsilon0*epsilonr*w/d;
G=sigma_diel*w/d;
disp("R,L,G,C parameters of a parallel copper plate
   transmission line ")
disp(r,"Resistance in ohm/m");
disp(L,"Inductance in Henry/m");
disp(c,"Capacitance in Farad/m");
disp(G,"Conductance in mS/m");
```

Scilab code Exa 2.5 Phase velocity and Wavelength of PCB material

```
epsilonr=4.6;
f=2*10^9;
z0=50; //line impedance
mu0=4*%pi*10^-7;
epsilon0=8.85*10^-12;
zf=sqrt(mu0/epsilon0); //free space impedance
```

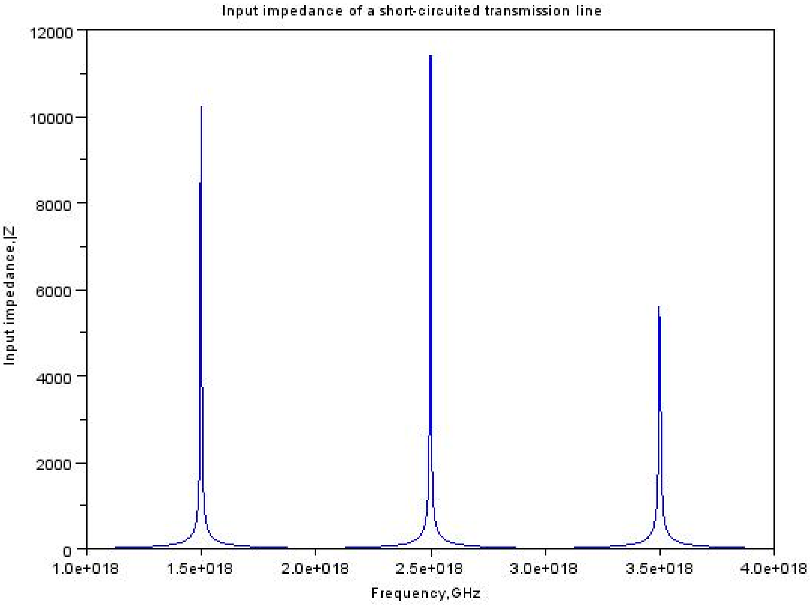

Figure 2.2: Input Impedance for a short circuited transmission line

```
temp=((epsilonr-1)/(epsilonr+1))*(0.23+(0.11/
   epsilonr));
temp1=2*%pi*(z0/zf)*sqrt((epsilonr+1)/2);
A=temp+temp1;
wtoh=(8*%e^A)/((%e^2*A)-2);
Eff=(epsilonr+1)/2+(epsilonr-1)/2*1/(sqrt(1+12*(1/(
   wtoh))));
vp=3*10^8/sqrt(Eff);
lambda=vp/f;
disp("metre/second",vp,"Phase velocity");
disp("metre",lambda,"Wavelength");
```

Scilab code Exa 2.6 Input Impedance for a short circuited transmission line

```
L=209.4*10^-9; //line inductance in H/m
C=119.5*10^-12; //line capacitance in F/m
vp=1/sqrt(L*C); // phase velocity
Z0=sqrt(L/C); // characteristic line impedance
d=0.1; // line length
N=500; // number of sampling points
f=1*10^9+3*10^9*(0:N)/N; // set frequency range
Z=tan(2*%pi*f*d/vp); // short circuit impedance
plot(f/1*10^9,abs(Z0*Z));
title('Input impedance of a short-circuited transmission line');
xlabel("Frequency,GHz");
ylabel("Input impedance,|Z");
```

Scilab code Exa 2.7 Input impedance of open circuited transmission line

```
L=209.4*10^-9; //line inductance in H/m
C=119.5*10^-12; //line capacitance in F/m
vp=1/sqrt(L*C); // phase velocity
Z0=sqrt(L/C); // characteristic line impedance
d=0.1; // line length
N=500; // number of sampling points
f=1e9+4e9*(0:N)/N; // set frequency range
Z=cotg(2*%pi*f*d/vp); // short circuit impedance
plot(f/1e9,abs(Z0*Z));
title('Input impedance of an open-circuited line');
xlabel('Frequency , GHz');
ylabel('Input impedance |Z|, {\Omega}');
```

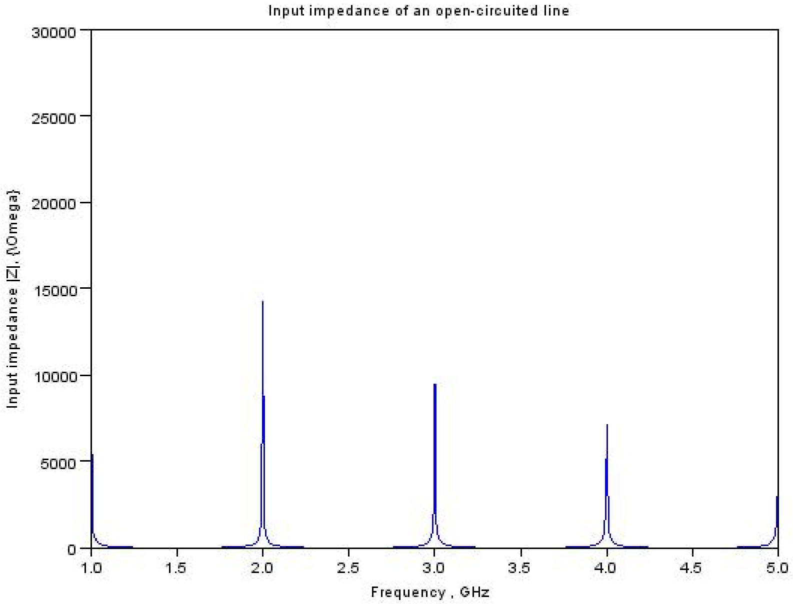

Figure 2.3: Input impedance of open circuited transmission line

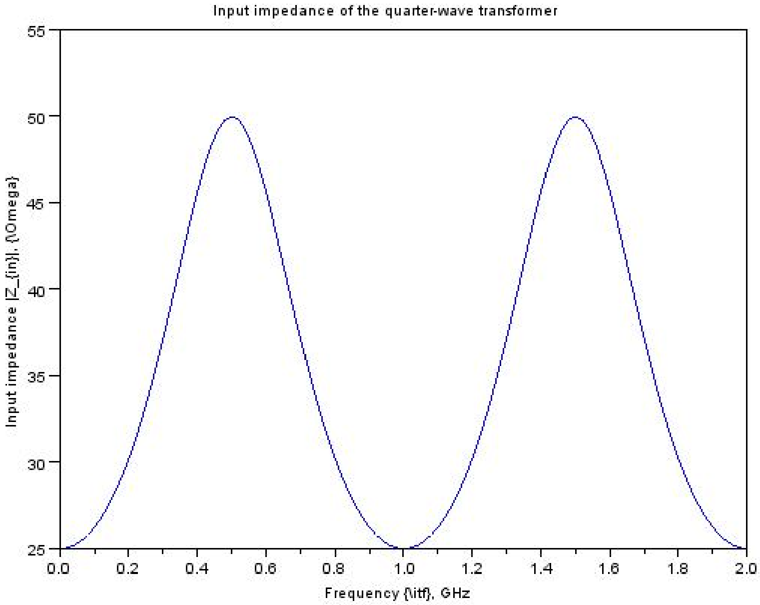

Figure 2.4: Quarter wave parallel plate line transformer

Scilab code Exa 2.8 Quarter wave parallel plate line transformer

```
ZL=25; //input impedance
Z0=50; //characteristic impedance
epsilonr=4;
dp=0.001;
f0=500e6;
mu0=4*%pi*1e-7;
epsilon0=8.85e-12;
Zline=sqrt(Z0*ZL); //line impedance
w=dp/Zline*sqrt(mu0/epsilon0/epsilonr);
```

```
L=mu0*dp/w;  //inductance
C=epsilon0*epsilonr*w/dp;  //capacitance
vp=1/sqrt(L*C);   //phase velocity
Z0=sqrt(L/C);
d=1/(4*f0*sqrt(L*C));
N=100;
f=2e9*(0:N)/N;
betta=2*%pi*f/vp;
Z=Zline*((ZL+%i*Zline*tan(betta*d))./(Zline+%i*ZL*tan(betta*d)));
plot(f/1e9,real(Z));
title('Input impedance of the quarter-wave transformer');
xlabel('Frequency {\itf}, GHz');
ylabel('Input impedance |Z_{in}|, {\Omega}');
```

Scilab code Exa 2.9 Power considerations of a transmission line

```
Zg=50; //generator impedance
Zo=75; //intrinsic impedance
Zl=40; //line impedance
Vg=5; //generator voltage
Ts=(Zg-Zo)/(Zg+Zo); //reflection coefficient at source
To=(Zl-Zo)/(Zl+Zo); //reflection coefficient at load
temp=1-(To^2);
temp1=(1-Ts)^2;
temp2=(1-Ts*To)^2;
Pin=((Vg)^2*temp1*temp2)/(8*Zo*temp); //input power
Pl=Pin; //power delivered to the load
disp("Watts",Pl,"The Power delivered to the load is same as that at the input-->");
```

Scilab code Exa 2.10 Return Loss of Transmission line section

```
RL=20; //load resistance
Zo=50; //intrinsic impedance
Rin=50; //input resistance
Tin=10^(-RL/20); //reflection coefficient at input
Rg1=Rin*(1+Tin)/(1-Tin);
Rg2=Rin*(1-Tin)/(1+Tin);
disp("Ohms",Rg1,"Source resistance for positive Tin=
   ");
disp("Ohms",Rg2,"Source resistance for negative Tin=
   ");
```

Chapter 3

The Smith Chart

Scilab code Exa 3.2 Input Impedance

```
Zl=30+%i*60; //load impedance
Z0=50; // intrinsic impedance
d=2*10^-2; //length of wire
f=2*10^9;
c=3*10^8;
T0=((Zl-Z0)/(Zl+Z0)); //load reflection coefficient
beta=((2*%pi*f)/(0.5*c));
T=-0.32-%i*0.55;
Zin=Z0*((1+T)/(1-T)); //input impedance
disp(”Ohms”,Zin,”Input impedance—>”);
```

Scilab code Exa 3.4 SWR circles

```
Z0=50; //define 50 Ohm characteristic impedance
Z=[50 48.5 75+%i*25 10-%i*5]; //define impedances
   for this example
```

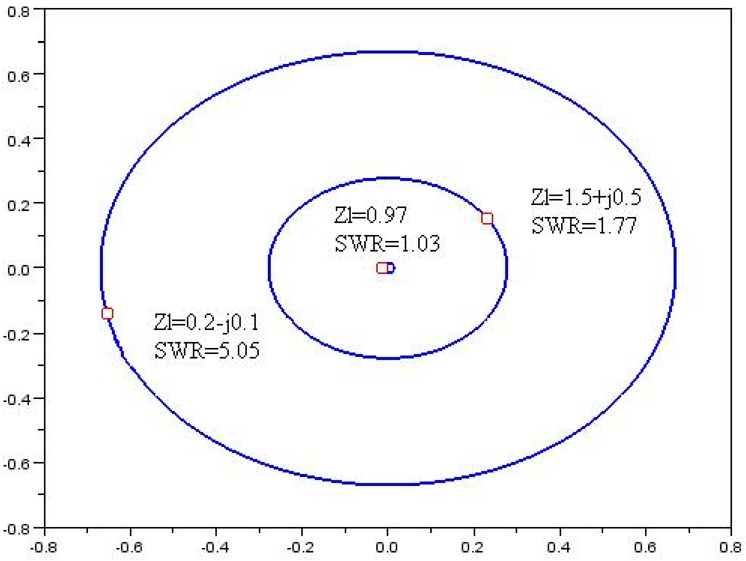

Figure 3.1: SWR circles

```
Gamma=(Z-Z0)./(Z+Z0) //compute corresponding reflection coefficients
SWR=(1+abs(Gamma))./(1-abs(Gamma)); //find the SWRs
a=0:0.01:2*%pi;
for n=1:length(Z)

plot(abs(Gamma(n))*cos(a),abs(Gamma(n))*sin(a),'b','linewidth',2);
plot(real(Gamma(n)), imag(Gamma(n)),'ro');
end;

for n=1:length(Z)
   if n~=1
   end;
end;
```

Chapter 4

Single and Multiport Networks

Scilab code Exa 4.3 Internal resistances and current gain of BJT

```
hie=5*10^3; //input impedance
hre=2*10^-4; //voltage feedback ratio
hfe=250; // small signal current gain
hoe=20*10^-6; //output admittance
rbc=hie/hre; // calculating base-collector
   resistance
rbe=hie/(1-hre); //calculating base-emitter
   resistance
beta=(hre+hfe)/(1-hre); //c calculating urrent gain
rce=hie/(hoe*hie-hre*hfe-hre); //collector-emitter
   resistance
disp("Ohms",rbc,"base collector resistance");
disp("Ohms",rbe,"base emitter resistance");
disp("Ohms",rce,"collector emitter resistance");
disp(beta,"current gain");
```

Scilab code Exa 4.7 S parameters and resistive elements of T network

```
Zin=50; //input impedance
Z0=50;
// defining scattering parameters
S11=0;
S22=0;
S21=1/sqrt(2);
S12=1/sqrt(2);
R1=((sqrt(2)-1)/(sqrt(2)+1))*Z0;
R2=R1;
R3=2*sqrt(2)*Z0;
disp(S21,S12,S22,S11,"Scattering parameters");
disp("Ohms",R3,"Ohms",R2,"Ohms",R1,"Resistance
   values R1,R2,R3:");
```

Chapter 5

An Overview of RF Filter Design

Scilab code Exa 5.1 Resonance frequency of a Bandpass filter

```
stacksize('max');
C=2*10^-12;
L=5*10^-9;
R=20;
Z0=50;
//f=[10^7:10^8:10^11];
//define frequency range
f_min=10e6;  //lower frequency limit
f_max=100e9; // upper frequency limit
N=100;       // number of points in the graph
f=f_min*((f_max/f_min).^((0:N)/N)); // compute
   frequency points on log scale
w=2*%pi.*f;
A=(w.*w*L*C-1)/(w*C);
S21=2*Z0./(2*Z0+R+%i*A);
f0=1./(2*%pi*sqrt(L*C));
disp("Hertz",f0,"Resonance frequency");
```

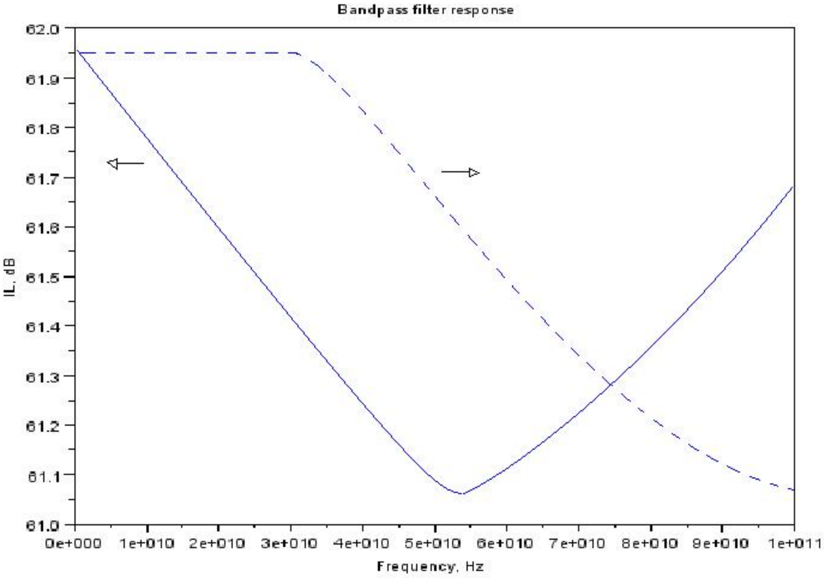

Figure 5.1: Resonance frequency of a Bandpass filter

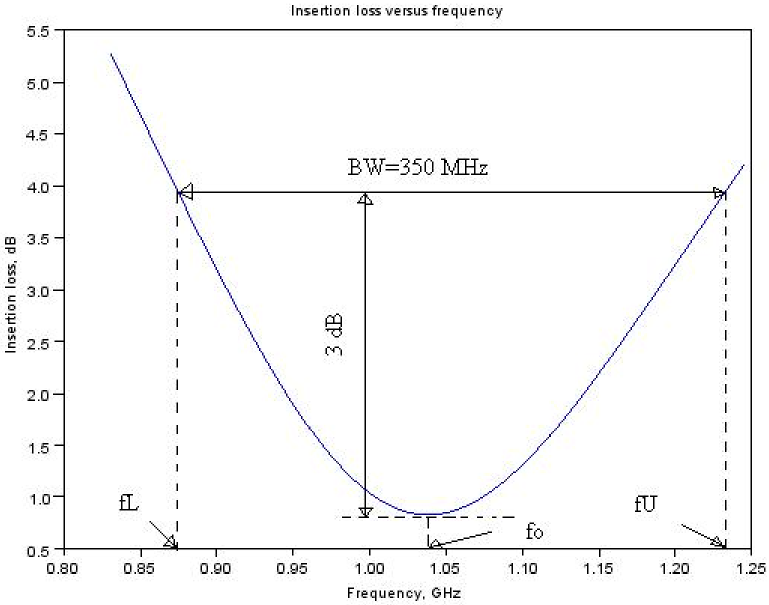

Figure 5.2: Quality factors of a filter

Scilab code Exa 5.2 Quality factors of a filter

```
//define problem parameters

Z0=50; //characteristic line impedance
ZG=50; //source impedance
ZL=50; //load impedance

//series RLC filter parameters
R=10;
```

```
L=50e-9;
C=0.47e-12;

VG=5; //generator voltage

//compute series resonance frequency
w0=1/sqrt(L*C);
f0=w0/(2*%pi);

//define a frequency range
delta=0.2;
w=((1-delta):2*delta/1000:(1+delta))*w0;

//compute quality factors
Q_LD=w0*L/(R+2*ZL) //loaded quality factor
Q_F=w0*L/R //filter quality factor
Q_E=w0*L/(2*ZL) //external quality factor

// compute Bandwidth
BW=f0/Q_LD

//compute input and load power
P_in=VG^2/(8*Z0)
P_L=P_in*Q_LD^2/Q_E^2

//compute insertion loss and load factor
epsilon=w/w0-w0./w;
LF=(1+epsilon.^2*Q_LD^2)/(1-Q_LD/Q_F)^2;
IL=10*log10(LF);

disp(Q_LD,"Loaded Quality Factor");
disp(Q_F,"Filter Quality Factor");
disp(Q_E,"External Quality Factor");
disp("Watts",P_in,"Input Power");
disp("Watts",P_L,"Power delivered to the load");
disp("Hertz",f0,"resonance frequency of the filter");
disp("Hertz",BW,"Bandwidth of the filter");
```

```
plot(w/2/%pi/1e9,IL);
title('Insertion loss versus frequency');
xlabel('Frequency, GHz');
ylabel('Insertion loss, dB');
```

Chapter 6

Active RF Components

Scilab code Exa 6.1 Conductivity of Si and Ge and GaAs

```
//define physical constants
q=1.60218e-19;
k=1.38066e-23;

// define material properties
Nc_300=[1.04e19 2.8e19  4.7e17];
Nv_300=[6e18    1.04e19 7e18];
mu_n=  [3900    1500    8500];
mu_p=  [1900    450     400];
Wg=    [0.66    1.12    1.424];

T0=273;
T=-50:250; // temperature range in centigrade

sigma=zeros([3 length(T)]);

for s=1:3 //loop through all semi conductor
   materials
   Nc=Nc_300(s)*((T+T0)/300).^(3/2);
```

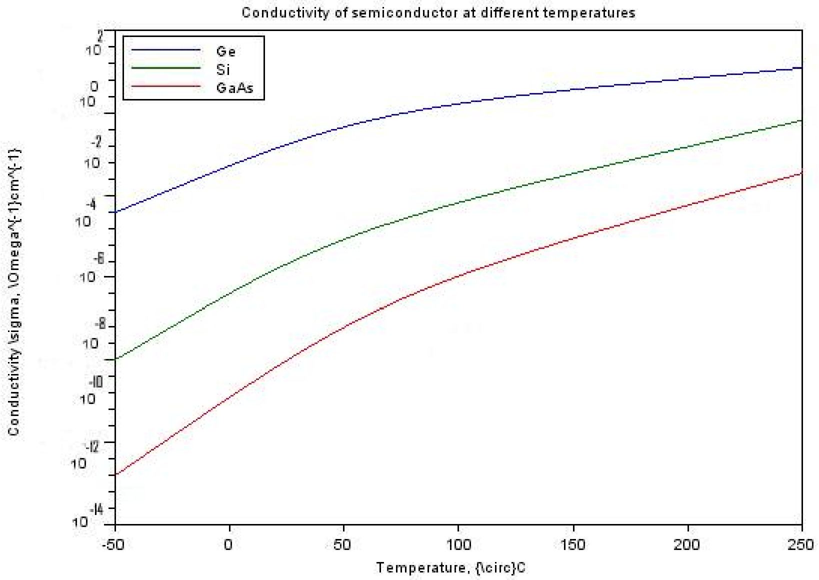

Figure 6.1: Conductivity of Si and Ge and GaAs

```
   Nv=Nv_300(s)*((T+T0)/300).^(3/2);
sigma=[q*sqrt(Nc.*Nv).*(exp(-Wg(s)./(2*k*(T+T0)/q)))
   *(mu_n(s)+mu_p(s))];
end;

plot(T,sigma(1),T,sigma(2),T,sigma(3));
legend('Ge','Si','GaAs',2);
title('Conductivity of semiconductor at different
   temperatures');
xlabel('Temperature, {\circ}C');
ylabel('Conductivity \sigma, \Omega^{-1}cm^{-1}');
```

Scilab code Exa 6.2 Barrier Voltage of a pn Junction

```
// doping concentrations
Na=1*10^18;
Nd=5*10^15;
//intrinsic concentrations
ni=1.5*10^10;
T=300;
term=(Na*Nd)/(ni*ni);
k=1.38*10^-23;
q=1.6*10^-19;
Vdiff=(k*T)*log(term)/q;
disp("Volts",Vdiff,"Barrier voltage");
```

Scilab code Exa 6.3 Depletion Layer Capacitance of a pn Junction

```
//define problem parameters

```

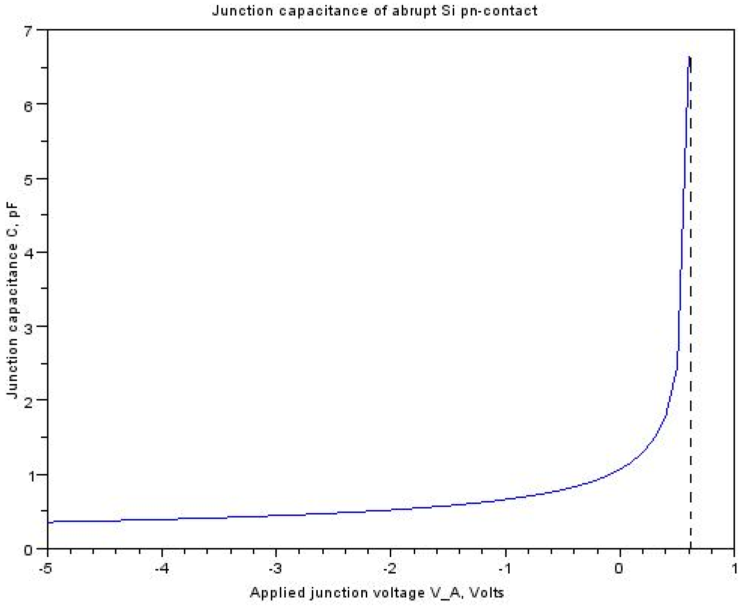

Figure 6.2: Depletion Layer Capacitance of a pn Junction

```
ni=1.5e10*1e6; //intrinsic carrier concentration in Si [m^(-3)]
Na=1e15*1e6; //acceptor doping concentration [m^(-3)]
Nd=5e15*1e6; //donor concentration [m^(-3)]
A=1e-4*1e-4; //cross sectional area [m^2]
eps_r=11.9;  //cross sectional area [m^2]

//define physical constants (SI units)
q=1.60218e-19; //electron charge
k=1.38066e-23; //Boltzmann's constant
eps0=8.85e-12; //permittivity of free space

eps=eps_r*eps0;

T=300; //temperatuure

//compute diffusion barrier voltage
Vdiff=k*T/q*log(Na*Nd/ni^2)

//junction capacitance at zero applied voltage
C0=A*sqrt(q*eps/(1/Na+1/Nd)/2/Vdiff)

//extents of the space charge region
dn=sqrt(2*eps*Vdiff/q*Na/Nd/(Na+Nd));
dp=sqrt(2*eps*Vdiff/q*Nd/Na/(Na+Nd));

//define range for applied voltage
VA=-5:0.1:Vdiff;

//compute junction capacitance
C=C0*(1-VA/Vdiff).^(-1/2);

plot(VA,C/1e-12);
title('Junction capacitance of abrupt Si pn-contact');
xlabel('Applied junction voltage V_A, Volts');
ylabel('Junction capacitance C, pF');
```

Scilab code Exa 6.4 Parameters of a Schottky diode

```
//doping concentrations
Nc=2.8*10^19;
Nd=1*10^16;
term=Nc/Nd;
k=1.38*10^-23; //Boltzman’s constant
q=1.6*10^-19; //charge
Vc=(k*T)*log(term)/q;
Vm=5.1; //workfunction
X=4.05; //affinity
Vd=(Vm-X)-Vc; //Barrier Voltage
Epsilon=11.9*8.854*10^-12;
ds=sqrt((2*Epsilon*Vd)/(q*Nd));
A=1*10^-4; //cross-sectional area
Cj=(A*Epsilon)/(ds); //junction capacitance
disp(”Volts”,Vc,”Conduction Band potential”);
disp(”Volts”,Vd,”Built in Barrier Voltage”);
disp(”metre”,ds,”Space Charge Width”);
disp(”Farads”,Cj,”Junction Capacitance”);
```

Scilab code Exa 6.7 Maximum forward current gain of bipolar junction transistor

```
Ndemitter=1*10^19; // donor concentration in emitter
Nabase=1*10^17; //acceptor concentration in base
de=0.8*10^-6; //spatial extent of the emitter
db=1.2*10^-6; //spatial extent of the base
alpha=2.8125;
beta=(alpha*Ndemitter*de)/(Nabase*db);
disp(beta,”Maximum forward current gain”);
```

Scilab code Exa 6.8 Thermal analysis involving a BJT mounted on a heat sink

```
Tj=150;
Ts=25;
Pw=15;
Rthjs=(Tj-Ts)/Pw; //Junction-to-solder point
   resistance
Rthca=2;
Rthhs=10;
Ta=60;
Rthtot=Rthjs+Rthca+Rthhs; //total thermal resistance
Pth=(Tj-Ta)/(Rthtot); //dissipated power
disp("Watts",Pth,"Maximum dissipated power");
```

Scilab code Exa 6.9 Drain saturation current in a MESFET

```
//define problem parameters
Nd=1e16*1e6;
d=0.75e-6;
W=10e-6;
L=2e-6;
eps_r=12;
Vd=0.8;
mu_n=8500e-4;
Vgs=0:-0.01:-4;

//define physical constants
q=1.60218e-19;// electron charge
eps0=8.85e-12;// permittivity of free space
```

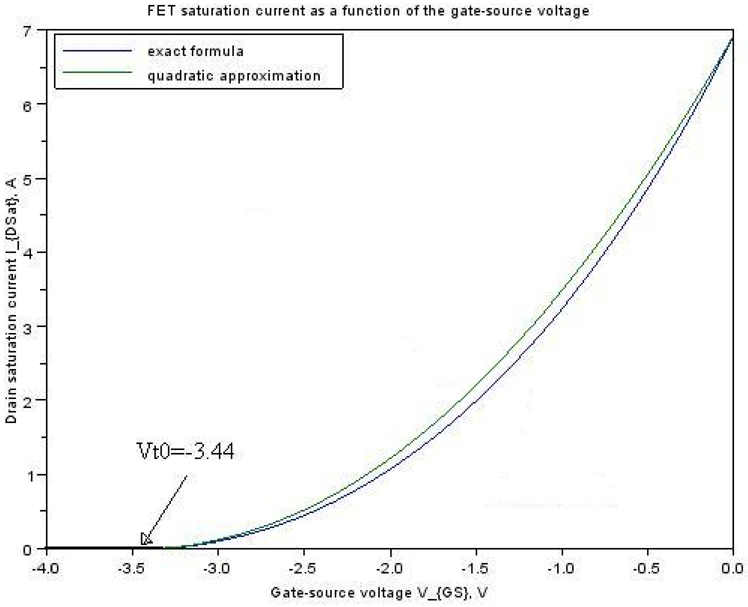

Figure 6.3: Drain saturation current in a MESFET

```

eps=eps_r*eps0;

//pinch-off voltage
Vp=q*Nd*d^2/(2*eps)

//threshold voltage
Vt0=Vd-Vp

//conductivity of the channel
sigma=q*mu_n*Nd

//Channel conductance
G0=q*sigma*Nd*W*d/L

//saturation current using the exact formula
Id_sat=G0*(Vp/3-(Vd-Vgs)+2/(3*sqrt(Vp))*(Vd-Vgs).^(3/2)).*(1-(Vgs<Vt0));
Idss=Id_sat(1)

//saturation current using the quadratic law approximation
Id_sat_square=Idss*(1-Vgs/Vt0)^2;

plot(Vgs,Id_sat,Vgs,Id_sat_square);
legend('exact formula', 'quadratic approximation',2);
title('FET saturation current as a function of the gate-source voltage');
xlabel('Gate-source voltage V_{GS}, V');
ylabel('Drain saturation current I_{DSat}, A');
```

Scilab code Exa 6.10 Current Voltage characterisitcs of a MESFET

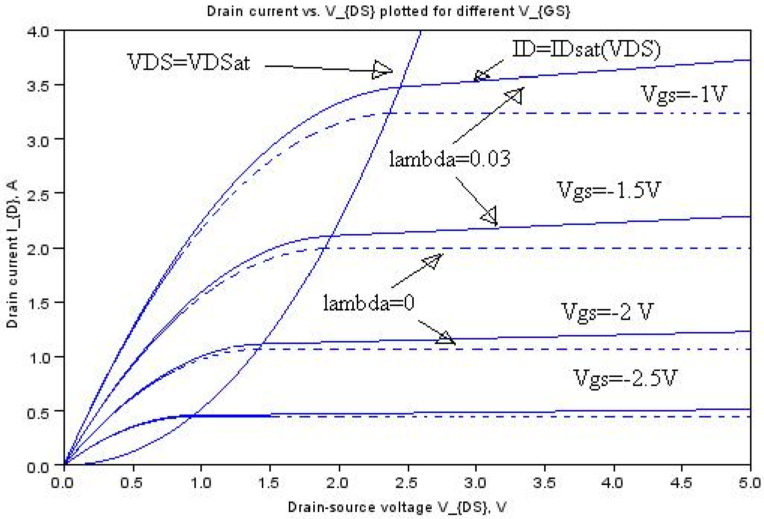

Figure 6.4: Current Voltage characterisitcs of a MESFET

```
//define problem parameters
Nd=1e16*1e6;
d=0.75e-6;
W=10e-6;
L=2e-6;
eps_r=12;
Vd=0.8;
mu_n=8500*1e-4;
lambda=0.03;

//define physical constants
q=1.60218e-19; //electron charge
eps0=8.85e-12; //permittivity of free space

eps=eps_r*eps0;

// pinch-off voltage
Vp=q*Nd*d^2/(2*eps)

//threshold voltage
Vt0=Vd-Vp

//conductivity of the channel
sigma=q*mu_n*Nd

//channel conductance
G0=q*sigma*Nd*W*d/L

//define the range for gate source voltage
Vgs_min=-2.5;
Vgs_max=-1;
Vgs=Vgs_max:-0.5:Vgs_min;

//drain source voltage
Vds=0:0.01:5;

//compute drain saturation voltage
Vds_sat=Vgs-Vt0;
```

```

//first the drain current is taken into account the channel length modulation
for n=1:length(Vgs)
   if Vgs(n)>Vt0
      Id_sat=G0*(Vp/3-(Vd-Vgs(n))+2/(3*sqrt(Vp))*(Vd-Vgs(n))^(3/2));
   else
      Id_sat=0;
   end;

   Id_linear=G0*(Vds-2/(3*sqrt(Vp)).*((Vds+Vd-Vgs(n)).^(3/2)-(Vd-Vgs(n))^(3/2))).*(1+lambda*Vds);
   Id_saturation=Id_sat*(1+lambda*Vds);
   Id=Id_linear.*(Vds<=Vds_sat(n))+Id_saturation.*(Vds>Vds_sat(n));
   plot(Vds,Id);
set(gca(),"auto_clear","off");
end;

//next the channel length modulation is not taken into account
for n=1:length(Vgs)
   if Vgs(n)>Vt0
      Id_sat=G0*(Vp/3-(Vd-Vgs(n))+2/(3*sqrt(Vp))*(Vd-Vgs(n))^(3/2));
   else
      Id_sat=0;
   end;

   Id_linear=G0*(Vds-2/(3*sqrt(Vp)).*((Vds+Vd-Vgs(n)).^(3/2)-(Vd-Vgs(n))^(3/2)));
   Id_saturation=Id_sat;
   Id=Id_linear.*(Vds<=Vds_sat(n))+Id_saturation.*(Vds>Vds_sat(n));
   plot(Vds, Id);
end;

```

```
//computation of drain saturation current

Vgs=0:-0.01:-4;
Vds_sat=Vgs-Vt0;

Id_sat=G0*(Vp/3-(Vd-Vgs)+2/(3*sqrt(Vp))*(Vd-Vgs).^(3/2)).*(1+lambda*Vds_sat).*(1-(Vgs<Vt0));

plot(Vds_sat, Id_sat);

mtlb_axis([0 5 0 4]);
title('Drain current vs. V_{DS} plotted for different V_{GS}');
xlabel('Drain-source voltage V_{DS}, V');
ylabel('Drain current I_{D}, A');
```

Scilab code Exa 6.11 Computation of HEMT related electric characteristics

```
//define problem parameters
Nd=1e18*1e6;
Vb=0.81;
eps_r=12.5;
d=50e-9;
dWc=3.5e-20;
W=10e-6;
L=0.5e-6;
mu_n=8500*1e-4;

//define physical constants
q=1.60218e-19;//electron charge
eps0=8.85e-12;//permittivity of free space

```

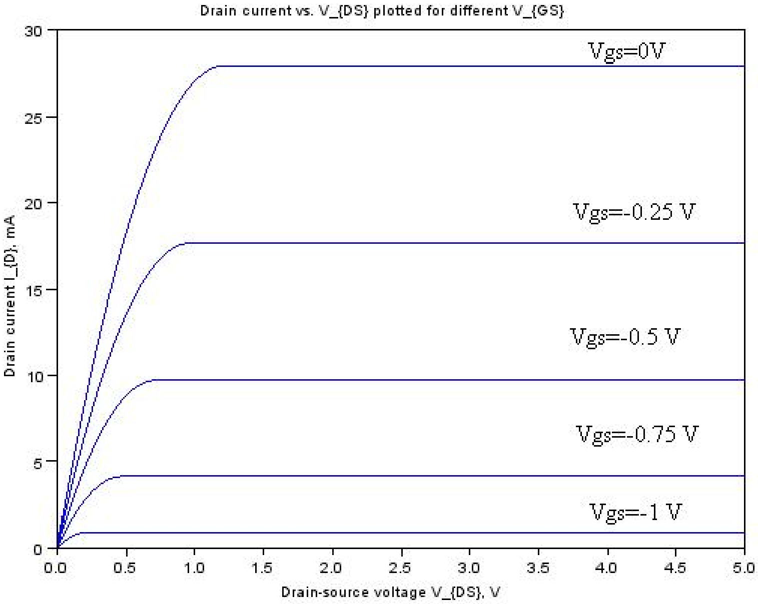

Figure 6.5: Computation of HEMT related electric characteristics

```
eps=eps_r*eps0;

//pinch-off voltage
Vp=q*Nd*d^2/(2*eps)

//threshold voltage
Vth=Vb-dWc/q-Vp

//drain-source applied voltage range
Vds=0:0.01:5;

//gate-source voltages
Vgs_r=-1:0.25:0;




for n=1:length(Vgs_r)
   Vgs=Vgs_r(n);
   Id=mu_n*W*eps/(L*d)*((Vds*(Vgs-Vth)-Vds.*Vds/2).*(1-(Vds>(Vgs-Vth)))+1/2*(Vgs-Vth)^2*(1-(Vds<=(Vgs-Vth))));
   plot(Vds,Id/1e-3);
   set(gca(),"auto_clear","off");
end;


title('Drain current vs. V_{DS} plotted for different V_{GS}');
xlabel('Drain-source voltage V_{DS}, V');
ylabel('Drain current I_{D}, mA');
```

Chapter 7

Active RF Component Modelling

Scilab code Exa 7.1 Small signal pn diode model

```
//define problem parameters
TT=500e-12; // transit time
T0=300;     //temperature
Is0=5e-15;  // reverse saturation current at 300K
Rs=1.5;     // series resistance
nn=1.16;    //emission coefficient

// parameters needed to describe temperature behavior of
// the band-gap energy in Si
alpha=7.02e-4;
beta=1108;
Wg0=1.16;
pt=3;

// quiescent current
Iq=50e-3;
```

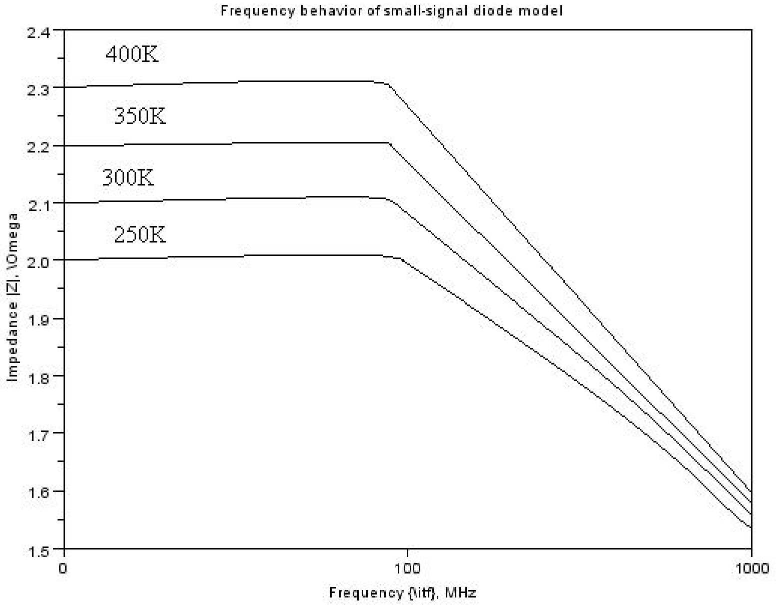

Figure 7.1: Small signal pn diode model

```

// frequency range 10MHz to 1GHz
f_min=10e6;   // lower limit
f_max=1e9;    //upper limit
N=300;        // number of points in the graph
f=f_min*((f_max/f_min).^((0:N)/N)); // compute frequency points on log scale

// temperatures for which analysis will be performed
T_points=[250 300 350 400];

// define physical constants
q=1.60218e-19; // electron charge
k=1.38066e-23; // Boltzmann's constant

for n=1:length(T_points)
   T=T_points(n);
   s=sprintf('T=%.f\n',T);
   Vt=k*T/q;

   Wg=Wg0-alpha*T^2/(beta+T);
   s=sprintf('%s   Wg(T)=%f\n',s,Wg);

   Is=Is0*(T/T0)^(pt/nn)*exp(-Wg/Vt*(1-T/T0));
   s=sprintf('%s   Is(T)=%e\n',s,Is);

   Vq=nn*Vt*log(1+Iq/Is);
   s=sprintf('%s   Vq(T)=%f\n',s,Vq);

   Rd=nn*Vt/Iq;
   s=sprintf('%s   Rd(T)=%f\n',s,Rd);

   Cd=Is*TT/nn/Vt*exp(Vq/nn/Vt);
   s=sprintf('%s   Cd(T)=%fpF\n',s,Cd/1e-12)

   Zc=1./(%i*2*%pi*f*Cd);

   Zin=Rs+Rd*Zc./(Rd+Zc);
```

```

    plot(f/1e6,abs(Zin));
    set(gca(),"auto_clear","off");
end;

title('Frequency behavior of small-signal diode model');
xlabel('Frequency {\itf}, MHz');
ylabel('Impedance |Z|, \Omega');
```

Scilab code Exa 7.4 Parameters of BJT

```
//first we define all parameters for the transistor and the circuit
Z0=50; //characteristic imedance of the system

Vcc=3.6; //power supply voltage
Vce=2; //collector voltage
Ic=10e-3; //collector current

T=300; //ambient temperature (300K)

//transistor parameters (they are very similar to BFG403W)
beta=145;    // current gain
Is=5.5e-18;  // saturation current
VAN= 30;     // forward Early voltage
tau_f=4e-12; // forward transition time
rb=125;      // base resistance
rc=15;       // collector resistance
re=1.5;      // emitter resistance
Lb=1.1e-9;   // base inductance
Lc=1.1e-9;   // collector inductance
Le=0.5e-9;   // emitter inductance
Cjc=16e-15;  // collector junction capacitance at
```

```
zero applied voltage
mc=0.2;       // collector junction grading coefficient
Cje=37e-15;   // emitter junction capacitance at zero applied voltage
me=0.35;      // emitter junction grading coefficient
phi_be=0.9;   // base-emitter diffusion potential
phi_bc=0.6;   // base-collector diffusion potential
Vbe=phi_be;   // base-emitter voltage

// some physical constants
k=1.38e-23;    // Boltzmann's constant
q=1.6e-19;     // elementary charge
VT=k*T/q;      // thermal potential

disp('DC biasing parameters');

Ib=Ic/beta;
disp("Amperes",Ib,"Base current");

Rc=(Vcc-Vce)/Ic;
disp("Ohms",Rc,"Collector resistance");

Rb=(Vcc-Vbe)/Ib;
disp("Ohms",Rb,"Base resistance");


r_pi=VT/Ib;
disp("Ohms",r_pi,"Rpi");

r0=VAN/Ic;
disp("Ohms",r0,"R0");

gm=beta/r_pi;
disp("Mho",gm,"Gm");

Vbc=Vbe-Vce;
Cmu=Cjc*(1-Vbc/phi_bc)^(-mc);
```

```
disp("Farads",Cmu,"base collector capacitance");

if(Vbe<0.5*phi_be)
   Cpi_junct=Cje*(1-Vbe/phi_be)^(-me);
else
   C_middle=Cje*0.5^(-me);
   k_middle=1-0.5*me;
   Cpi_junct=C_middle*(k_middle+me*Vbe/phi_be);
end;

disp("Farads",Cpi_junct,"Junction Capacitance");

Cpi_diff=Is*tau_f/VT*exp(Vbe/VT);
disp("Farads",Cpi_diff,"Differential capacitance");

Cpi=Cpi_junct+Cpi_diff;
disp("Farads",Cpi,"Total Capacitance");

C_miller=Cmu*(1+gm*r_pi/(r_pi+rb)*Z0*r0/(r0+rc+Z0));
disp("Farads",C_miller,"Miller Capacitance");

C_input=Cpi+C_miller;
disp("Farads",C_input,"Total input capacitance");
```

Scilab code Exa 7.5 Cutoff frequency of GaAs MESFET

```
l=1*10^-6; //length
w=200*10^-6; //width
d=0.5*10^-6; //depth
E0=8.854*10^-12;
Er=13.1;
q=1.6*10^-19; //electron charge
Nd=1*10^16; //doping concentration
mun=8500;
Vp=(q*Nd*d^2)/(2*Er*E0);
```

```
G0=(q*mun*Nd*w)/l;
gm=0.0358;
Cap=(E0*Er*w*l)/d;
fT=gm/(2*%pi*Cap);
disp("Hertz",fT,"Cut off frequency");
```

Scilab code Exa 7.6 Small signal Hybrid pi parameters without Miller Effect

```
Icq=6*10^-3;
Ibq=40*10^-6;
Van=30; //Early voltage
q=1.6*10^-19;
k=1.38*10^-23;
T=300;
fT=37*10^9; //Transition frequency
gm=(Icq*q)/(k*T);
beta0=Icq/Ibq;
r0=Van/Icq;
rpi=beta0/gm;
Cpi=(beta0)/(2*%pi*fT*rpi);
disp("Hybrid pi parametrs without Miller effect");
disp("Mho",gm,"gm");
disp("Ohms",rpi,"Rpi");
disp("Farads",Cpi,"Cpi");
disp("Ohms",r0,"R0");
disp(beta0,"Beta0");
```

Chapter 8

Matching and biasing networks

Scilab code Exa 8.11 Efficiency of different types of amplifiers

```
theta=(1:1:360)/180*%pi; //define conduction angle

//compute efficiency
nu=-1/2*(theta-sin(theta))./(theta.*cos(theta/2)-2*
   sin(theta/2));

plot(theta/%pi*180,nu*100,'r','linewidth',2);
set(gca(),"auto_clear","off");
plot([0 180],[%pi/4*100 %pi/4*100],'b:');
plot([180 180],[0 %pi/4*100],'b:');
plot(180,%pi/4*100,'bo');
plot(360,50,'bo');
mtlb_axis([0 360 50 100]);
title('Maximum theoretical efficiency of the
   amplifier');
xlabel('Conduction angle \Theta_0, deg.');
ylabel('Efficiency \eta, %');
```

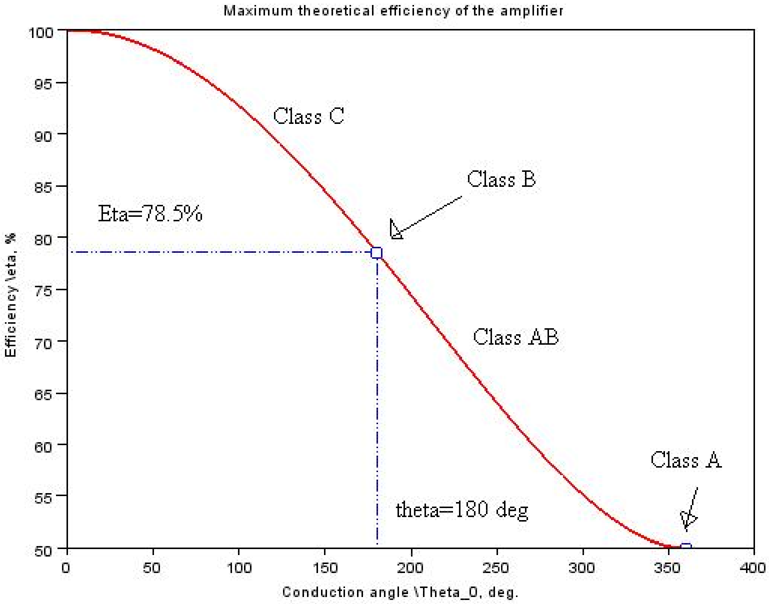

Figure 8.1: Efficiency of different types of amplifiers

Scilab code Exa 8.12 Design of passive biasing networks for a BJT in CE config

```
Ic=10*10^-3; //Collector current
Vce=3;
Vcc=5;
beta=100; //current gain
Vbe=0.8;
I1=Ic+Ic/beta;
R1=(Vcc-Vce)/I1;
R2=(Vce-Vbe)/(Ic/beta);
Vx=1.5;
R3=(Vx-Vbe)/(Ic/beta);
Ix=10*(Ic/beta);
R11=(Vx/Ix);
R22=(Vcc-Vx)/(Ix+(Ic/beta));
R4=(Vcc-Vce)/Ic;
disp("Amperes",I1,"I1","Ohms",R1,"R1","Ohms",R2,"R2"
    ,"Ohms",R3,"R3","Ohms",R11,"R11","Ohms",R22,"R22"
    ,"Ohms",R4,"R4");
```

Chapter 9

RF Transistor Amplifier Design

Scilab code Exa 9.1 Power relations for an RF amplifier

```
//defining scattering parameters
S11=0.102-%i*0.281;
S21=0.305+%i*3.486;
S12=0.196-%i*0.03471;
S22=0.2828-%i*0.2828;

Vs=5;
Zs=40;
Zl=73;
Z0=50;

Ts=(Zs-Z0)/(Zs+Z0);
Tl=(Zl-Z0)/(Zl+Z0);
Tin=S11+(S21*S12*Tl)/(1-S22*Tl);
Tout=S22+(S12*S21*Ts)/(1-S11*Ts);

a=S21^2;
b=1-Ts^2;
c=1-Tl^2;

Gt=(c*a*b)/((1-Tl*Tout)^2*(1-S11*Ts)^2);
```

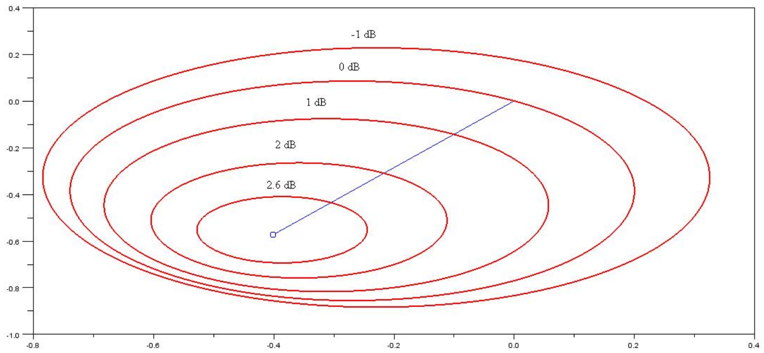

Figure 9.1: Computation of source gain circles for a unilateral design

```
Gtu=(c*a*b)/((1-Tl*S22)^2*(1-S11*Ts)^2);
Ga=(a*b)/((1-Tout)^2*(1-S11*Ts)^2);
G=(a*c)/((1-Tin)^2*(1-S22*Tl)^2);

d=abs(Gt);

Pin=(Z0*(Vs)^2)/((Zs+Z0)^2*(1-Tin*Ts)^2*2);
pinR=real(Pin);
pinI=imag(Pin);
Pinc=sqrt(pinR^2+pinI^2);
PA=78.1*10^-3;
Pl=PA*d;
disp(Pl,"Power delivered to load in watts");
```

Scilab code Exa 9.7 Computation of source gain circles for a unilateral design

```
//define s11 parameter of the transistor
s11=0.7*exp(%i*(125)/180*%pi);
```

```

//compute the maximum gain achievable by the input matching network
Gs_max=1/(1-abs(s11)^2);
Gs_max_dB=10*log10(Gs_max)

//find the reflection coefficient for the maximum gain
Gs_opt=conj(s11);

//draw a straight line connecting Gs_opt and the origin
set(gca(),"auto_clear","off");
plot([0 real(Gs_opt)],[0 imag(Gs_opt)],'b');
plot(real(Gs_opt),imag(Gs_opt),'bo');

//specify the angle for the constant gain circles
a=(0:360)/180*%pi;

//plot source gain circles
gs_db=[-1 0 1 2 2.6];
gs=exp(gs_db/10*log(10))/Gs_max;

for n=1:length(gs)
   dg=gs(n)*conj(s11)/(1-abs(s11)^2*(1-gs(n)));
   rg=sqrt(1-gs(n))*(1-abs(s11)^2)/(1-abs(s11)^2*(1-gs(n)));
   plot(real(dg)+rg*cos(a),imag(dg)+rg*sin(a),'r','linewidth',2);
end;
```

Scilab code Exa 9.8 Design of 18 dB single stage MESFET amplifier

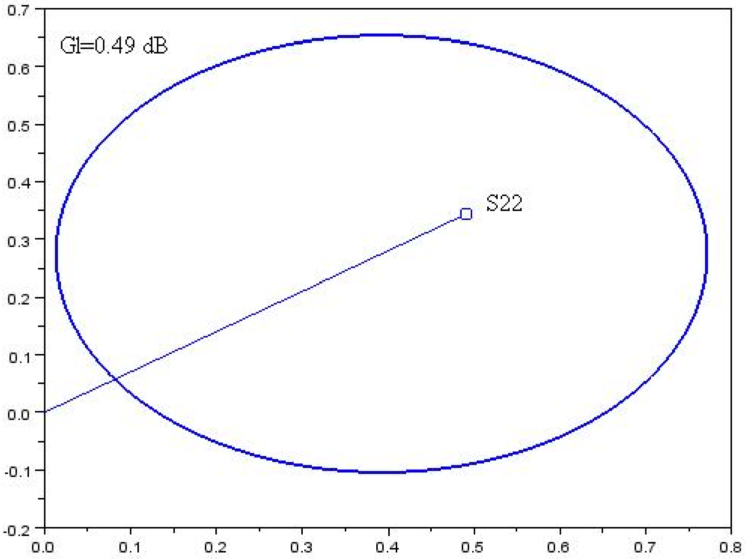

Figure 9.2: Design of 18 dB single stage MESFET amplifier

```
s11=0.5*exp(%i*(-60)/180*%pi);
s12=0.02*exp(%i*(-0)/180*%pi);
s21=6.5*exp(%i*(+115)/180*%pi);
s22=0.6*exp(%i*(-35)/180*%pi);

Gs_max=1/(1-abs(s11)^2);
Gl_max=1/(1-abs(s22)^2);

G0=abs(s21)^2;

Gmax=Gs_max*G0*Gl_max;
Gs_max_dB=10*log10(Gs_max)
Gl_max_dB=10*log10(Gl_max)
G0_dB=10*log10(G0)
Gmax_dB=10*log10(Gmax)
Ggoal_dB=18;
Gload_dB=Ggoal_dB-G0_dB-Gs_max_dB;
Gl_opt=conj(s22);

set(gca(),”auto_clear”,”off”);
plot([0 real(Gl_opt)],[0 imag(Gl_opt)],’b’);
plot(real(Gl_opt),imag(Gl_opt),’bo’);
a=(0:360)/180*%pi;
gl=exp([Gload_dB]/10*log(10))/Gl_max;
dg=gl*conj(s22)/(1-abs(s22)^2*(1-gl));
rg=sqrt(1-gl)*(1-abs(s22)^2)/(1-abs(s22)^2*(1-gl));
plot(real(dg)+rg*cos(a),imag(dg)+rg*sin(a),’b’,’
   linewidth’,2);
```

Scilab code Exa 9.13 Amplifier design using the constant operating gain circles

```
//define the S—parameters of the transistor
```

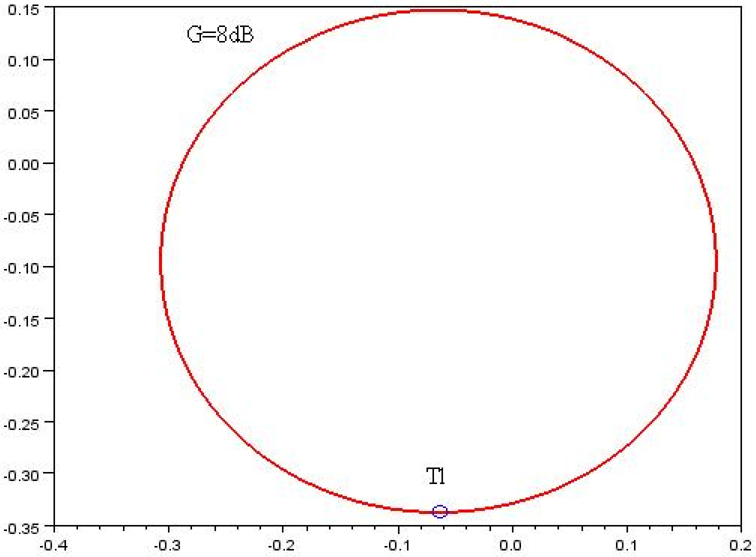

Figure 9.3: Amplifier design using the constant operating gain circles

```
s11=0.3*exp(%i*(+30)/180*%pi);
s12=0.2*exp(%i*(-60)/180*%pi);
s21=2.5*exp(%i*(-80)/180*%pi);
s22=0.2*exp(%i*(-15)/180*%pi);

K=1.18

//find the maximum gain
Gmax=abs(s21/s12)*(K-sqrt(K^2-1));
Gmax_dB=10*log10(Gmax)

//specify the target gain
G_goal_dB=8; //would like to build an amplifier with 8dB gain
G_goal=10^(G_goal_dB/10); //convert from dB to normal units

//find constant operating power gain circles
go=G_goal/abs(s21)^2;

//find the center of the constant operating power gain circle
dgo=go*conj(s22-conj(s11))/(1+go*(abs(s22)^2));


//find the radius of the circle
rgo1=sqrt(1-2*K*go*abs(s12*s21)+go^2*abs(s12*s21)^2);
rgo=rgo1/abs(1+go*(abs(s22)^2));

//plot a circle in the Smith Chart
a=(0:360)/180*%pi;

set(gca(),"auto_clear","off");
plot(real(dgo)+rgo*cos(a),imag(dgo)+rgo*sin(a),'r','linewidth',2);

//choose the load reflection coefficient
```

```
zL=1-%i*0.53
GL=(zL-1)/(zL+1);

plot(real(GL),imag(GL),'bo');

[Ro,Theta]=polar(atan(imag(Gs),real(Gs)));
Gin=s11+s12*s21*GL/(1-s22*GL);
Gs=conj(Gin);
Gs_abs=abs(Gs)
Gs_angle=(Theta/%pi)*180;

zs=(1+Gs)/(1-Gs);
```

Scilab code Exa 9.14 Design of small signal amplifier for minimum noise figure and specified gain

```
global Z0;
Z0=50;

//define the S-parameters of the transistor
s11=0.3*exp(%i*(+30)/180*%pi);
s12=0.2*exp(%i*(-60)/180*%pi);
s21=2.5*exp(%i*(-80)/180*%pi);
s22=0.2*exp(%i*(-15)/180*%pi);

//pick the noise parameters of the transistor
Fmin_dB=1.5
Fmin=10^(Fmin_dB/10);
Rn=4;
Gopt=0.5*exp(%i*45/180*%pi);

//compute a noise circle
Fk_dB=1.6;
```

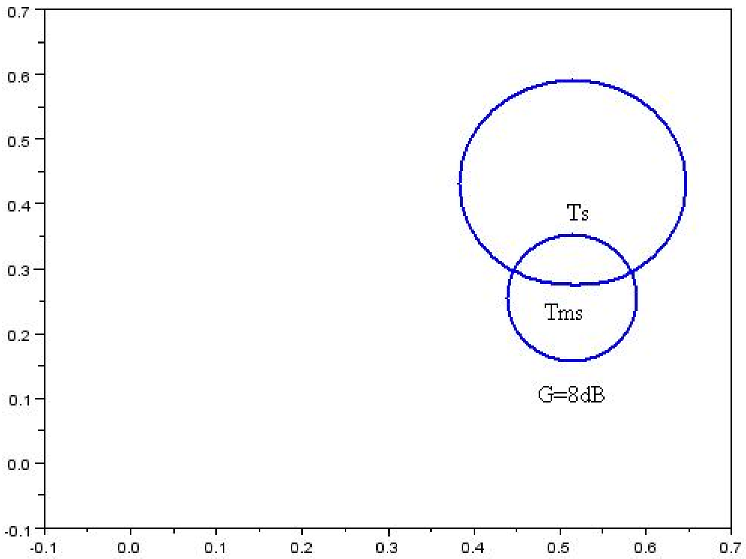

Figure 9.4: Design of small signal amplifier for minimum noise figure and specified gain

```
Fk=10^(Fk_dB/10);


Qk=abs(1+Gopt)^2*(Fk-Fmin)/(4*Rn/Z0) //noise circle parameter
dfk=Gopt/(1+Qk); //circle center location
rfk=sqrt((1-abs(Gopt)^2)*Qk+Qk^2)/(1+Qk) //circle radius


//plot a noise circle
a=[0:360]/180*%pi;
set(gca(),”auto_clear”,”off”);
plot(real(dfk)+rfk*cos(a),imag(dfk)+rfk*sin(a),'b','linewidth',2);

// plot optimal reflection coefficient
plot(real(Gopt),imag(Gopt),'bo');


//specify the desired gain
G_goal_dB=8;
G_goal=10^(G_goal_dB/10);

//find the constant operating power gain circles
go=G_goal/abs(s21)^2; // normalized the gain
dgo=go*conj(s22-conj(s11))/(1+go*(abs(s22)^2)); //center

rgo=sqrt(1-2*K*go*abs(s12*s21)+go^2*abs(s12*s21)^2);
rgo=rgo/abs(1+go*(abs(s22)^2));

//map a constant gain circle into the Gs plane
rgs=rgo*abs(s12*s21/(abs(1-s22*dgo)^2-rgo^2*abs(s22)^2));
dgs=((1-s22*dgo)*conj(s11-dgo)-rgo^2*s22)/(abs(1-s22*dgo)^2-rgo^2*abs(s22)^2);

```

```
//plot a constant gain circle in the Smith Chart
set(gca(),"auto_clear","off");
plot(real(dgs)+rgs*cos(a),imag(dgs)+rgs*sin(a),'r','linewidth',2);



//choose a source reflection coefficient Gs
Gs=dgs+%i*rgs;
plot(real(Gs), imag(Gs), 'ro');
//text(real(Gs)-0.05,imag(Gs)+0.08,'\bf\Gamma_S');

//find the actual noise figure
F=Fmin+4*Rn/Z0*abs(Gs-Gopt)^2/(1-abs(Gs)^2)/abs(1+Gopt)^2;

//print out the actual noise figure
Actual_F_dB=10*log10(F)
```

Scilab code Exa 9.15 Constant VSWR design for given gain and noise figure

```
global Z0;
Z0=50;

//define the S-parameters of the transistor
s11=0.3*exp(%i*(+30)/180*%pi);
s12=0.2*exp(%i*(-60)/180*%pi);
s21=2.5*exp(%i*(-80)/180*%pi);
s22=0.2*exp(%i*(-15)/180*%pi);

//noise parameters of the transistor
Fmin_dB=1.5
Fmin=10^(Fmin_dB/10);
Rn=4;
```

```
Gopt=0.5*exp(%i*45/180*%pi);


//compute a noise circle
Fk_dB=1.6;//desired noise performance
Fk=10^(Fk_dB/10);

Qk=abs(1+Gopt)^2*(Fk-Fmin)/(4*Rn/Z0); //noise circle parameter
dfk=Gopt/(1+Qk); //circle center location
rfk=sqrt((1-abs(Gopt)^2)*Qk+Qk^2)/(1+Qk); //circle radius


//plot a noise circle
a=[0:360]/180*%pi;
set(gca(),"auto_clear","off");
plot(real(dfk)+rfk*cos(a),imag(dfk)+rfk*sin(a),'b','linewidth',2);

//specify the goal gain
G_goal_dB=8;
G_goal=10^(G_goal_dB/10);


//find constant operating power gain circles
go=G_goal/abs(s21)^2;  //normalized gain
dgo=go*conj(s22-delta*conj(s11))/(1+go*(abs(s22)^2)); //center

rgo=sqrt(1-2*K*go*abs(s12*s21)+go^2*abs(s12*s21)^2);
rgo=rgo/abs(1+go*(abs(s22)^2)); //radius

//map a constant gain circle into the Gs plane
rgs=rgo*abs(s12*s21/(abs(1-s22*dgo)^2-rgo^2*abs(s22)^2));
dgs=((1-s22*dgo)*conj(s11-delta*dgo)-rgo^2*s22)/(abs(1-s22*dgo)^2-rgo^2*abs(s22)^2);
```

```

//plot constant gain circle in the Smith Chart
set(gca(),"auto_clear","off");
plot(real(dgs)+rgs*cos(a),imag(dgs)+rgs*sin(a),'r','linewidth',2);


//choose a source reflection coefficient Gs
Gs=dgs+%i*rgs;

//find the corresponding GL
GL=(s11-conj(Gs))/(delta-s22*conj(Gs));

//find the actual noise figure
F=Fmin+4*Rn/Z0*abs(Gs-Gopt)^2/(1-abs(Gs)^2)/abs(1+Gopt)^2;

//% print out the actual noise figure
Actual_F_dB=10*log10(F)

//find the input and output reflection coefficients
Gin=s11+s12*s21*GL/(1-s22*GL);
Gout=s22+s12*s21*Gs/(1-s11*Gs);


//find the VSWRin and VSWRout
Gimn=abs((Gin-conj(Gs))/(1-Gin*Gs));
Gomn=abs((Gout-conj(GL))/(1-Gout*GL));

VSWRin=(1+Gimn)/(1-Gimn); //VSWRin should be unity since we used the constant operating gain approach
VSWRout=(1+Gomn)/(1-Gomn);

//specify the desired VSWRin
VSWRin=1.5;

//find parameters for constant VSWR circle
```

```
Gimn=(1-VSWRin)/(1+VSWRin)
dvimn=(1-Gimn^2)*conj(Gin)/(1-abs(Gimn*Gin)^2); //circle center
rvimn=(1-abs(Gin)^2)*abs(Gimn)/(1-abs(Gimn*Gin)^2); //circle radius

//plot VSWRin=1.5 circle in the Smith Chart
plot(real(dvimn)+rvimn*cos(a),imag(dvimn)+rvimn*sin(a),'g','linewidth',2);


//plot a graph of the output VSWR as a function of the Gs position on the constant VSWRin circle
Gs=dvimn+rvimn*exp(%i*a);
Gout=s22+s12*s21*Gs./(1-s11*Gs);

//find the reflection coefficients at the input and output matching networks
Gimn=abs((Gin-conj(Gs))./(1-Gin*Gs));
Gomn=abs((Gout-conj(GL))./(1-Gout*GL));

//and find the corresponding VSWRs
VSWRin=(1+Gimn)./(1-Gimn);
VSWRout=(1+Gomn)./(1-Gomn);

figure; //open new figure for the VSWR plot
plot(a/%pi*180,VSWRout,'r',a/%pi*180,VSWRin,'b','linewidth',2);
legend('VSWR_{out}','VSWR_{in}');
title('Input and output VSWR as a function of \Gamma_S position');
xlabel('Angle \alpha, deg.');
ylabel('Input and output VSWRs');
mtlb_axis([0 360 1.3 2.3])


//choose a new source reflection coefficient
Gs=dvimn+rvimn*exp(%i*85/180*%pi);
```

```

//find the corresponding output reflection
   coefficient
Gout=s22+s12*s21*Gs./(1-s11*Gs);

//compute the transducer gain in this case
GT=(1-abs(GL)^2)*abs(s21)^2.*(1-abs(Gs).^2)./abs(1-
   GL*Gout).^2./abs(1-Gs*s11).^2;
GT_dB=10*log10(GT)

//find the input and output matching network
   reflection coefficients
Gimn=abs((Gin-conj(Gs))./(1-Gin*Gs));
Gomn=abs((Gout-conj(GL))./(1-Gout*GL));

//and find the corresponding VSWRs
VSWRin=(1+Gimn)./(1-Gimn)
VSWRout=(1+Gomn)./(1-Gomn)

//also compute the obtained noise figure
F=Fmin+4*Rn/Z0*abs(Gs-Gopt)^2/(1-abs(Gs)^2)/abs(1+
   Gopt)^2;
F_dB=10*log10(F)
```

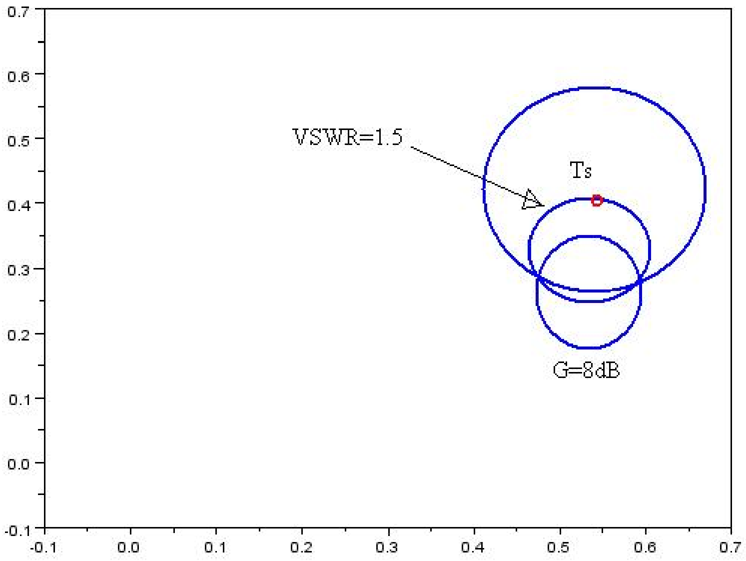

Figure 9.5: Constant VSWR design for given gain and noise figure

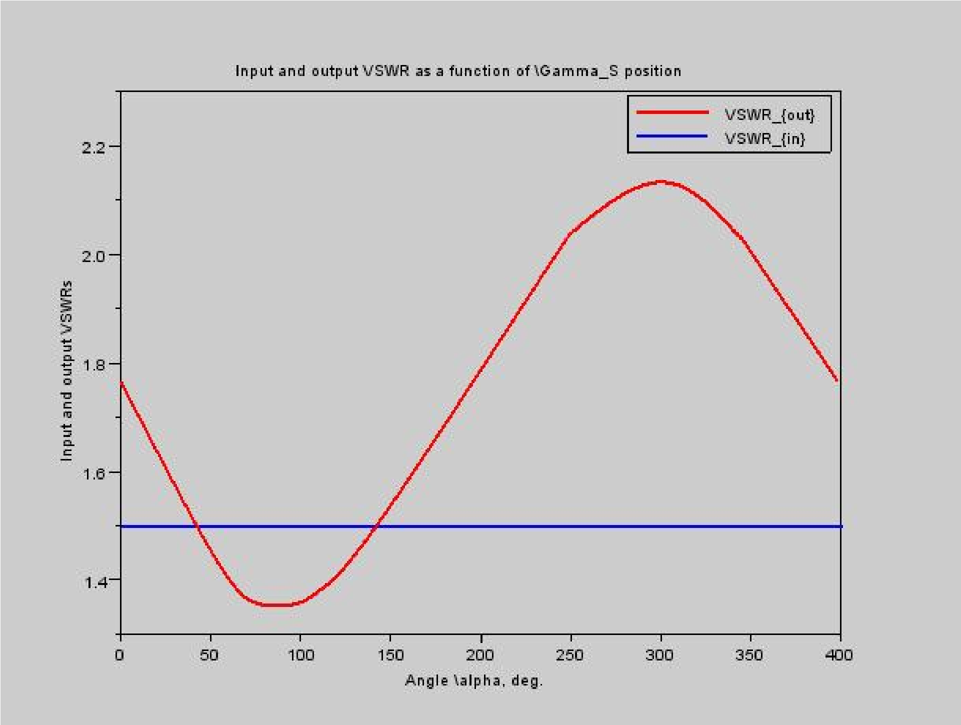

Figure 9.6: Constant VSWR design for given gain and noise figure

Chapter 10

Oscillators and Mixers

Scilab code Exa 10.1 Design of a Colpitt oscillator

```
fo=200*10^6;
Vce=3;
Ic=3*10^-3;

Cbc=0.1*10^-15;
rBE=2*10^3;
rCE=10*10^3;
Cbe=100*10^-15;
L3=50*10^-9;
L=50*10^-9;
gm=0.11666;

disp("DC values of Hparameters are");
h11=rBE;
h12=0;
h21=rBE*gm;
h22=1/rCE;

disp("Mho",h22,"h22",h21,"h21",h12,"h12","Ohms",h11,
   "h11");
k=h21/(h11*h22-h21*h12);
```

```
A=(1+k)/L;
B=A^2;
C=16*k*(%pi)^2*fo^2*(h22/h11);
D=8*k*(%pi)^2*fo^2;
C2=(A+sqrt(B+C))/D;
C1=k*C2;

disp("H parameters at resonance frequency");
w=2*%pi*fo;
E=1+%i*w*(Cbe+Cbc)*rBE;

hie=rBE/E;
hre=(%i*w*Cbc*rBE)/E;
hfe=(rBE*(gm-%i*w*Cbc))/E;
hoe=h22+(%i*w*Cbc*(1+gm*rBE+%i*w*Cbe*rBE))/E;
disp("Mho",hoe,"hoe",hfe,"hfe",hre,"hre","Ohms",hie,
    "hie");
```

Scilab code Exa 10.2 Prediction of resonance frequencies of quartz crystal

```
stacksize("max");
//define crystal parameters
Lq=0.1;
Rq=25;
Cq=0.3*10^-12;
C0=1*10^-12;

//find series resonance frequency
ws0=1/sqrt(Lq*Cq);
disp(ws0);
ws=ws0*(1+Rq^2/2*C0/Lq);
fs=ws/2/%pi

//find parallel resonance frequency
wp0=sqrt((Cq+C0)/(Lq*Cq*C0));
```

```
wp=wp0*(1-Rq^2/2*C0/Lq);
fp=wp/2/%pi

//define frequency range for this plot
f=(0.9:0.00001:1.1)*1e6;
w=2*%pi*f;

//find abmittance of the resonator
Y=%i.*w*C0+1./(Rq+%i*(w*Lq-1./(w*Cq)));

plot(f/1e6,abs(imag(Y)));
mtlb_axis([0.9 1.1 1e-10 1e-1]);
title('Admittance of the quartz crystal resonator');
xlabel('Frequency {\itf}, MHz');
ylabel('Susceptance |B|, \Omega');
```

Scilab code Exa 10.3 Adding a positive feedback element to initiate oscillations

```
Z0=50;
//oscillation frequency
f=2*10^9;
w=2*%pi*f;
//transistor S-parameters at oscillation frequency

s_tr=[0.94*exp(%i*174/180*%pi),0.013*exp(-%i*98/180*
   %pi);1.9*exp(-%i*28/180*%pi),1.01*exp(-%i*17/180*
   %pi)];
s11=ss2tf(1,1);
s12=ss2tf(1,2);
s21=ss2tf(2,1);
s22=ss2tf(2,2);

```

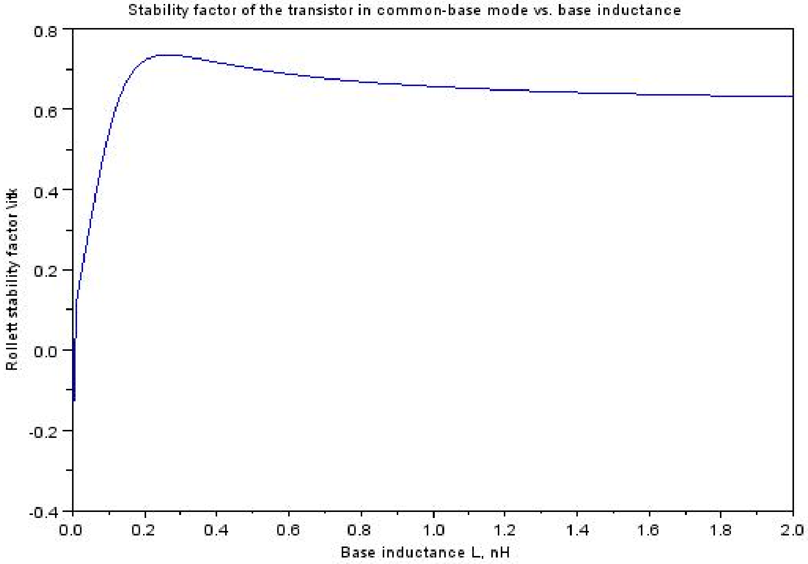

Figure 10.1: Adding a positive feedback element to initiate oscillations

```
//find the Z-parameters of the transistor
z_tr=ss2tf(s_tr,Z0);

//attempt to add inductor to base in order to increase instability
L=(0:0.01:2)*1e-9;

Z_L=%i*w*L;
z_L=[1,1;1,1];

N=length(L);

//create variables for the S_parameters of the transistor with the inductor
s11=zeros([1 N]);
s12=zeros([1 N]);
s21=zeros([1 N]);
s22=zeros([1 N]);

//Rollett stability factor
K=zeros([1 N]);

for n=1:N
   z_total=z_tr+z_L*Z_L(n);
   s_total=ss2tf(z_total,Z0);
   s11(n)=s_total(1,1);
   s12(n)=s_total(1,2);
   s21(n)=s_total(2,1);
   s22(n)=s_total(2,2);
   K(n)=(1-abs(s11(n))^2-abs(s22(n))^2+abs(det(s_total))^2)/2/abs(s12(n)*s21(n));
end;

plot(L/1e-9,K);
title('Stability factor of the transistor in common-base mode vs. base inductance');
xlabel('Base inductance L, nH');
ylabel('Rollett stability factor \itk')
```

Scilab code Exa 10.6 Dielectric resonator oscillator design

```
//define the S-paramters of the transistor at resonance frequency
s11=1.1*exp(%i*(170)/180*%pi);
s12=0.4*exp(%i*(-98)/180*%pi);
s21=1.5*exp(%i*(-163)/180*%pi);
s22=0.9*exp(%i*(-170)/180*%pi);

s=[s11,s12;s21,s22];

//define oscillation frequency
f0=8e9;
w0=2*%pi*f0;

//define parameters of the dielectric resonator
Z0=50;
beta=7;
R=beta*2*Z0;
Qu=5e3;

//compute equivalent L and C
L=R/(Qu*w0);
C=1/(L*w0^2);

//find output reflection coefficient of the DR
Gout_abs=beta/(1+beta);
Gout_angle=-atan(imag(s11),real(s11))/%pi*180;

//compute electrical length of the transmission line for the DR
theta0=-1/2*Gout_angle
Gout=Gout_abs*exp(%i*Gout_angle*%pi/180);

```

```
//find the output impedance of the DR
Zout=Z0*(1+Gout)/(1-Gout)


// find the equivalent capacitance (it will be necessary for the computation of the oscillator without DR)
CC=-1/(w0*imag(Zout))

Rs=50;

//define the frequency for the plot
delta_f=0.05e9; //frequency range
f=f0-delta_f/2 : delta_f/100 : f0+delta_f/2;
w=2*%pi*f;

if theta0<0
   theta0=360+theta0;
end;

theta=theta0*f/f0/180*%pi;

//repeat the same computations as above, but for specified frequency range
Gs=(Rs-Z0)/(Rs+Z0);
G1=Gs*exp(-%i*2*theta);
R1=Z0*(1+G1)./(1-G1);
Zd=1./(1/R+1./(%i*w*L+%i*w*C));
R1d=R1+Zd;
G1d=(R1d-Z0)./(R1d+Z0);
G2=G1d.*exp(-%i*2*theta);

//compute the output reflection coefficient (we have oscillations if |Gout|>1)
Gout=s22+s12*s21*G2./(1-s11*G2);

figure;
plot(f/1e9,abs(Gout),'b','linewidth',2);
```

```
title('Output reflection coefficient of the oscillator with DR');
xlabel('Frequency f, GHz');
ylabel('Output reflection coefficient |\Gamma_{out}|');
mtlb_axis([7.975 8.025 0 14]);


//Redefine the frequency range (we have to increase it in order to be able to observe any variations in the response
delta_f=5e9;
f=f0-delta_f/2 : delta_f/100 : f0+delta_f/2;
w=2*%pi*f;

//Compute the output reflection coefficient of the oscillator but with DR replaced by a series combination of resistance and capacitance
ZZ2=real(Zout)+1./(%i*w*CC);
GG2=(ZZ2-Z0)./(ZZ2+Z0);
GG=s22+s12*s21*GG2./(1-s11*GG2);

figure;
plot(f/1e9,abs(GG),'r','linewidth',2);
title('Output reflection coefficient of the oscillator without DR');
xlabel('Frequency f, GHz');
ylabel('Output reflection coefficient |\Gamma_{out}|');
```

Scilab code Exa 10.8 Local oscillator frequency selection

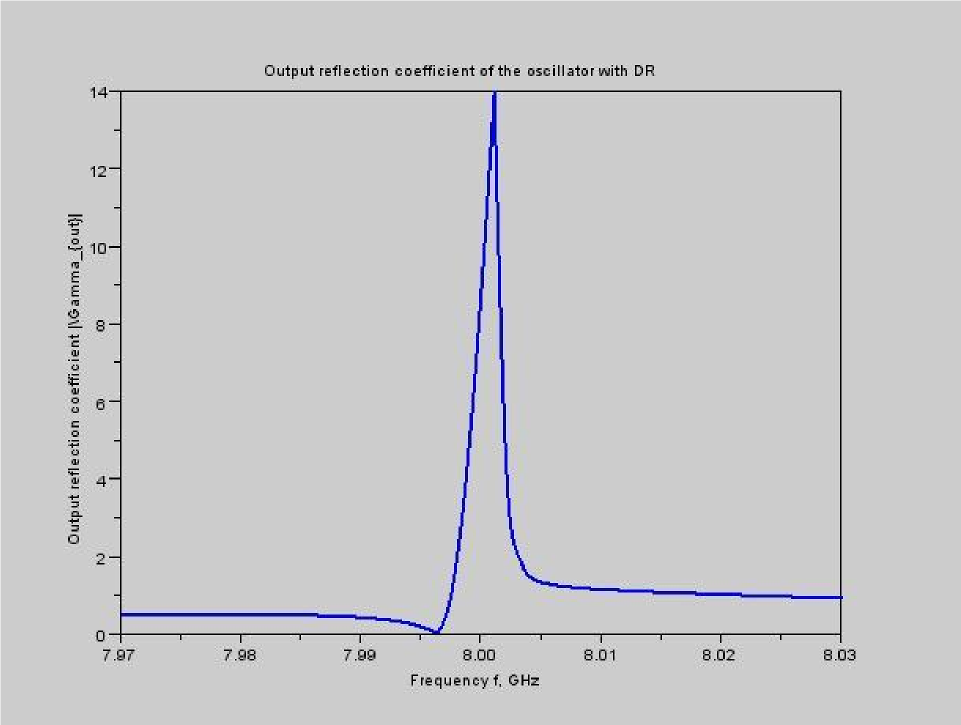

Figure 10.2: Dielectric resonator oscillator design

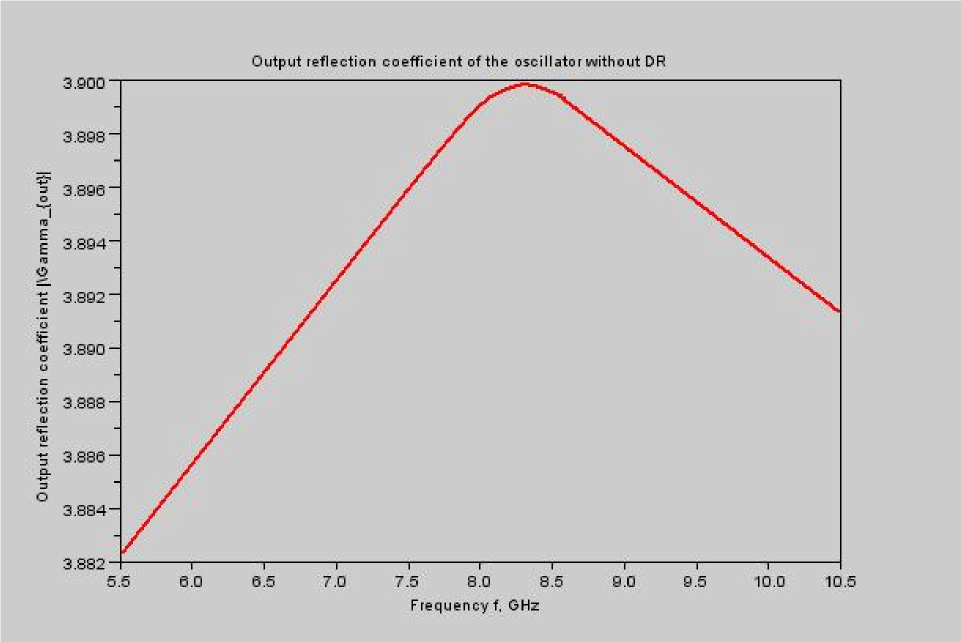

Figure 10.3: Dielectric resonator oscillator design

```
fRF=1.89*10^9; //RF frequency
BW=20*10^6; //Bandwidth
fIF=200*10^6; //Intermediate Frequency
flo=fRF+fIF; //Local oscillator frequency
Q=fIF/BW; //Quality factotr
disp(Q,”Quality Factor”);
```
