YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MİKRODALGA TRANSİSTORUN YAPAY SİNİR AĞI İLE PERFORMANS ANALİZİ ve MODELLENMESİ

Yük. Müh. Cemal TEPE

F.B.E Elektronik ve Haberleşme Mühendisliği Anabilim Dalında Haberleşme Programında Hazırlanan

DOKTORA TEZİ

Tez Savunma Tarihi	: 03-Temmuz-2000
Tez Danışmanı	: Prof. Dr. Filiz GÜNEŞ (YTÜ)
Jüri Üyeleri	: Prof. Dr. Osman PALAMUTÇUOĞLU (İTÜ)
	: Prof. Dr. Hasan DİNÇER (Kocaeli Ünv.)

İSTANBUL, 2000

İÇİNDEKİLER

SİMGE LİSTESİ

G_T	Transduser güç kazancı
$G_{T\max}$	Maksimum transduser güç kazancı
$G_{T\min}$	Minimum transduser güç kazancı
G_{Treq}	İstenilen transduser kazancı ($G_{T\min} \le G_{Treq} \le G_{T\max}$)
$F_{\min}$	Bir mikrodalga transistorun minimum gürültü faktörü
F	Gürültü faktörü ($F \ge F_{\min}$)
F_{req}	İstenilen gürültü faktörü ($F_{req} \ge F_{\min}$)
$\lvert\rho_i\rvert$	Giriş yansıma katsayısı
V_i	Giriş VSWR ($V_i \ge 1$)
V_{ireq}	İstenilen giriş VSWR ($V_{ireq} \ge 1$)
f	Çalışma frekansı
V_{DS}, V_{CE}	Bir mikrodalga transistorun kutuplama gerilimi
I_{DS}, I_C	Bir mikrodalga transistorun kutuplama akımı
CT	Konfigürasyon tipi
S_{ij}	Transistorun S-parametreleri (i,j=1,2)
$\lvert S_{ij}\rvert$	Transistorun S-parametreleri genliği (i,j=1,2)
$<S_{ij}$	Transistorun S-parametreleri açısı (i,j=1,2)
Γ_{opt}	Optimum Kaynak Yansıma Katsayısı
$\lvert\Gamma_{opt}\rvert$	Optimum Kaynak Yansıma Katsayısı genliği
$<\Gamma_{opt}$	Optimum Kaynak Yansıma Katsayısı açısı
R_n	Bir mikrodalga transistorun eşdeğer gürültü Direnci
$Z_L = R_L + jX_L$	Reel ve imajiner kısımlarıyla iki-kapılı devrenin yük empedansı
$Z_{L\max} = Z_L$	$G_{T\max}$'a karşılık gelen yük empedansı
$Z_S = R_S + jX_S$	Reel ve imajiner kısımlarıyla iki-kapılı devrenin kaynak empedansı
$Z_{S\max} = Z_S$	$G_{T\max}$'a karşılık gelen kaynak empedansı
$Z_{Lreq} = R_{Lreq} + jX_{Lreq}$	Reel ve imajiner kısımlarıyla iki-kapılı devrenin G_{Treq} için yük empedansı
$Z_{Sreq} = R_{Sreq} + jX_{Sreq}$	Reel ve imajiner kısımlarıyla iki-kapılı devrenin G_{Treq} için kaynak empedansı

$Z_{in} = R_{in} + jX_{in}$	İki-kapılı devrenin giriş empedansı
Z_{cin}	Zi-düzlemindeki dairelerin merkez fazörü
r_{in}	Zi-düzlemindeki dairelerin yarıçapı
Z_{cS}	Zs-düzlemindeki dairelerin merkez fazörü
r_s	Zs-düzlemindeki dairelerin yarıçapı
T1 ve T2	Kaynak düzleminde gürültü faktörü dairesi ile girişi VSWR dairesi arasındaki pozisyonların Zi-düzleminde oluşturduğu bölgeleri sınırlayan daireler
$Z_{ct1,2}$	Zi-düzlemindeki T1 ve T2 dairelerin merkez fazörü
$r_{ct1,2}$	Zi-düzlemindeki T1 ve T2 dairelerin yarıçapı
Z_{cg}	Zi-düzlemindeki kazanç dairesinin merkez fazörü
r_g	Zi-düzlemindeki kazanç dairesinin yarıçapı
$Z_{cn} = R_{cn} + jX_{cn}$	Kaynak düzleminde reel ve imajiner kısımlarıyla gürültü faktörü dairesinin merkez fazörü
r_n	Kaynak düzleminde gürültü faktörü dairesinin yarıçapı
$Z_{cv} = R_{cv} + jX_{cv}$	Kaynak düzleminde reel ve imajiner kısımlarıyla Giriş VSWR dairesinin merkez fazörü
r_v	Kaynak düzleminde Giriş VSWR dairesinin yarıçapı
$Z_{opt} = R_{opt} + jX_{opt}$	Reel ve imajiner kısımlarıyla iki-kapılı devrenin optimum kaynak empedansı
$z_{11}, z_{12}, z_{21}, z_{22}$	Kullanılan transistore ilişkin iki kapılının z-parametreleri
r_{11}, x_{11}; r_{12}, x_{12}; r_{21}, x_{21}; r_{22}, x_{22}	z-parametrelerinin sırasıyla reel ve sanal kısımları

KISALTMA LİSTESİ

CSSC	Kompleks Eşlenik Kaynak Kararlılık Dairesi (Complex Conjugate Source Stability Circle)
CT	Konfigürasyon tipi (Configuration Type)
ISC	Giriş Kararlılık Dairesi (Input Stability Circle)
NaN	Sayısal Veri Bulunamıyor Anlamında Kullanılıyor (Not a Number)
VSWR	Gerilim Duran Dalga Oranı (Voltage Standing Wave Ratio)
YSA	Yapay Sinir Ağı

ŞEKİL LİSTESİ

ÇİZELGE LİSTESİ

ÖNSÖZ

Bu tezin ortaya çıkmasında katkısı olan herkese en içten duygularımla sonsuz teşekkürler ederim.

Ayrıca doktoraya başlamamda ve tezin ortaya çıkmasında en fazla ve her zaman desteğini hissettiğim, gerekli bilgi, kaynak ve yönlendirmeyi sayelerinde her an hazır olarak bulduğum sevgili ve saygıdeğer Prof. Dr. Filiz GÜNEŞ hocama şükranlarımı kelimelerle ifade edebilmem mümkün değildir.

Aynı zamanda, her problemimizde bize gerekli bilgiyi ve zamanı ayıran, değerli bilgilerinden yararlanmamızı sağlayan sevgili hocam Prof. Dr. Fikret GÜRGEN'e, en zor günlerimizde bize çıkış yolları önererek değerli bilgilerini bizlerle paylaşan sayın hocam Prof. Dr. Binboğa Sıddık YARMAN'a içten teşekkürlerimi sunarım.

Tez süresince desteklerini hiç eksik etmeyen Yrd. Doç. Dr. Hamid TORPİ, Dr. Bedri A. ÇETİNER, Tolga Bazan, Arzu Tepe ve Kerime Gül'e değerli katkılarından dolayı teşekkürü bir borç bilirim.

Tüm öğrenimim süresince bana gerekli desteği ve ortamı sağlayan ailemin her bir bireyine ayrı ayrı teşekkür ederim.

Cemal Tepe

BABAMA ve ANNEME ...

ÖZET

Yapılan bu doktora tez çalışmasında, bir mikrodalga transistorun Yapay Sinir Ağı (YSA) ile performans analiz ve giriş/çıkış uydurma devrelerinin tasarımına yönelik yeni bir yöntem sunulmuştur. Dolayısıyla bu tez; YSA, performans analizi ve uydurma devre tasarımı olmak üzere üç ana bölümden oluşmaktadır.

Aktif devre elemanı (transistor) üretici verileriyle, YSA optimum sürede eğitilmektedir. Yapay sinir ağı eğitildikten sonra, klasik kestirim yöntemleri yerine kullanılarak, istenilen veriler kolay bir şekilde ve hızlıca elde edilebilmektedir. Böylelikle önerilen YSA modeli, optimizasyon prosedürü esnasında elemanın fizik denklemlerinin tekrar tekrar çözülmesini gerektirmeyecek ve klasik optimizasyon tekniklerinin ortak sorunu olan "başlangıç değer kestirimi" problemini ortadan kaldırarak tercih edilir duruma gelmektedir. Modelleme esnasında bir döngü içerisinde defalarca optimizasyona ihtiyaç duyulduğu düşünüldüğünde, önerilen yöntemin başarısı kolaylıkla ortaya çıkmaktadır.

Önerilen modelde, herhangi bir mikrodalga transistorun S ve gürültü parametreleri yapay sinir ağı çıkışı, frekans, kutuplama akımı ve gerilimi ile konfigürasyon tipi yapay sinir ağı girişini oluşturmaktadırlar. Böylelikle YSA, frekans, kutuplama ve konfigürasyon tipi parametreleriyle, S ve gürültü parametreleri arasında bir eşleşme yaratarak; eleman denklemleri çözülmeden optimizasyon işlemlerini yapma olanağı sağlamaktadır. Bu çalışmada, YSA üretici tarafından verilmeyen gürültü parametrelerini kestirebilmek için kullanılmıştır.

Transistorun performans analizinde kullanılan ve GÜNEŞ metoduna dayanan optimizasyon problemi, $F_{req} - F(R_S, X_S) = 0$ ve $V_{ireq} - V_i(R_S, X_S, R_L, X_L) = 0$ olmak üzere ($G_{T\max}$, maksimum kazanç ve G_{Treq} istenilen kazanç), $G_{T\max} - G_T(R_S, X_S, R_L, X_L) = 0$ ve ayrıca $G_{Treq} - G_T(R_S, X_S, R_L, X_L) = 0$ olacak şekilde iki ayrı durum için kararlı bölgede kalmak koşuluyla mümkün tüm $Z_S = R_S + jX_S$ ve $Z_L = R_L + jX_L$ kompleks sonlandırmalarını geometriksel ve matematiksel olarak belirlemek şeklinde ifade edilebilir. Bu durumda bu işlemler sonucunda; $\{F_{req}, V_{ireq}, G_{T\max}[f, V_{CE}, I_C]\} \Leftrightarrow Z_{L\max}[f, V_{CE}, I_C]$, $Z_{S\max}[f, V_{CE}, I_C]$ ve $\{F_{req}, V_{ireq}, G_{T\min} \le G_{Treq} \le G_{T\max}[f, V_{CE}, I_C]\} \Leftrightarrow Z_{Lreq}[f, V_{CE}, I_C]$, $Z_{Sreq}[f, V_{CE}, I_C]$ fonksiyonlarına ulaşılmış olacaktır. Burada f; frekansı ve V_{CE} ile I_C (veya V_{DS} ile I_{DS}) kutuplama noktasını dolayısıyla transistorun gürültü ve S-parametrelerini ifade etmektedir.

Bu tür yüksek dereceli polinomların içerildiği optimizasyon problemlerin çözümüne yönelik

bir takım yöntemler literatürde sıkça yer almaktadır ancak tanımladığımız problemin hem reel hem de sanal kısma sahip olması ve ayrıca çok hesaplama zamanına ihtiyaç duymaları önerilen geometriksel yöntemi daha avantajlı duruma getirmektedir. Bunun yanında bu tezde tüm çözümler hem analitik hem`de geometriksel olarak ifade edilmişlerdir. Kullanılan yöntem adımları ilgili bölümlerde açıklanmıştır.

Performans analizi sonucunda elde edilen sonlandırmalara ilişkin giriş/çıkış uydurma devrelerin tasarımına yönelik birçok yöntem bulunmaktadır. Biz burada ileriki bölümlerde açıklanan empedans yaklaşım yöntemini kullanacağız. Bu yöntemde veriler, pozitif reel bir z(s) empedans fonksiyonu ile modellendiği zaman bu z(s) empedans fonksiyonu sentez edilebilir olacaktır. Ölçülen verilerdeki reel ve sanal kısımları ayrı ayrı modellemek, Empedans Yaklaşımı ile Modelleme yöntemimizin esasını oluşturur. Reel ve sanal kısımları ayrı ayrı modellemek için, uygulanan empedans yaklaşımına ilişkin adımlar ileriki bölümlerde anlatılacaktır.

Bir mikrodalga transistorun YSA ile performans analizi ve modellenmesi konusunda ilk defa bu tezde ortaya konulan noktalar aşağıda sıralanmıştır.

a) Transistorun, klasik küçük-işaret eşdeğer devresi yerine YSA eşdeğeriyle, çalışma koşullarının (konfigürasyon tipi (CT), kutuplama akım (I_C) ve gerilimi (V_{CE}), frekans (f)) fonksiyonu olarak davranışının anında tayini ve bunun uygulamada kullanılması.

b) $F_{req} \geq F_{\min}$, $V_{ireq} \geq 1$ ve $G_{T\min} \leq G_{Treq} \leq G_{T\max}$ koşulları altında herhangi bir $(F_{req},\ V_{ireq},\ G_{Treq})$ uyumlu performans üçlüsü ile birlikte, $G_{T\max}$'ye karşılık gelen $Z_{L\max} = Z_L = R_L + jX_L$, $Z_{S\max} = Z_S = R_S + jX_S$ ve G_{Treq}'a karşılık gelen $Z_{Lreq} = R_{Lreq} + jX_{Lreq}$, $Z_{Sreq} = R_{Sreq} + jX_{Sreq}$ sonlandırmalarının, transistorun çalışma koşullarının (konfigürasyon tipi (CT), kutuplama akım (I) ve gerilimi (V), frekans (f)) fonksiyonu olarak tayin edilip hesaplanması.

c) Uyumlu performans üçlüsünü gerçekleyen kararlı bölgelerde istenilen adım aralığıyla mümkün tüm sonlandırmaların sistematik olarak tayin edilip hesaplanması.

d) Bulunan sonlandırmaların sistematik bir yöntemle (empedans parametreleri yaklaşımı) giriş/çıkış uydurma devreleri tasarımında kullanılması.

e) Bir mikrodalga transistorun için YSA, performans analizi ve giriş/çıkış uydurma devrelerinin elde edilmesi şeklinde bir blok yapının Aktif Mikrodalga devre sentezinde

kullanılmak üzere sunulması.

Tüm çalışmalar bilgisayar programlarıyla yapılmıştır ve programlarla çıktıları ilgili bölümlerde verilmiştir.

ABSTRACT

In this Ph.D. thesis, signal and noise behaviors of microwave transistors are modeled through the neural network approach for the whole operating ranges including frequency, bias and configuration types to provide input data for the method of graphical design of a low-noise, low VSWR microwave stable amplifier and design input and output matching circuits of this amplifier. So this thesis consists of three main chapters: Neural Network, performance Analysis of a Microwave transistor and designing input/output matching circuits.

Here, the transistor is modeled by a black box whose small-signal and noise parameters are evaluated through a neural network, based upon the fitting of both of these parameters for multiple bias and configuration type with the target space. The concurrent modeling procedure does not require to solve any equation of the physics of the device during the optimization procedure. After having trained the network in an optimum time interval compared with the other modeling techniques, inputting the required configuration type (CT), bias condition (V_{DS},I_{DS} or V_{CE},I_C) and operating frequency, one can obtain the corresponding eight signal parameters which are the scattering parameters in our study, and four noise parameters.

In another chapter of this study, maximum transducer power gain of a stable microwave transistor has been analytically expressed in terms of noise figure, input VSWR and the two-port parameters. The analysis has been based on GÜNEŞ method with the geometrical representations of constant noise, input VSWR and gain circles in the impedance planes and the solution has been obtained in the input impedance plane. The corresponding terminations are also derived analytically. The conditionally stable transistor has also been discussed and the same reasoning has been performed on the computer-plotted graphs and contours clearly. Different curve groups depending on VSWR-Gain-Noise parameters are plotted on computer. In the program, to help the designer an option for plotter outputs of curves is included for the user.

Finally, input and output matching circuits of the amplifier have been designed by using Impedance parameter approach explained in the following chapters. As know by a theorem, any impedance can be stated as a sum of minimum reactance function and a foster function. The characteristic of the minimum reactance function is that its real part is non-negative and it can be fully obtained from either real or imaginary part. The required steps of this method are given in the following chapters.

In this Ph.D. thesis the concepts that are new in the literature are presented as follows;

a) The transistor is modeled by a black box whose small-signal and noise parameters are immediately evaluated through a neural network, based upon the fitting of both of these parameters for multiple bias, configuration type and frequency.

b) Concerning $F_{req} \geq F_{\min}$, $V_{ireq} \geq 1$ and $G_{T\min} \leq G_{Treq} \leq G_{T\max}$; any given $(F_{req},\ V_{ireq},\ G_{Treq})$ performance triplets that provides $Z_{L\max} = Z_L = R_L + jX_L$, $Z_{L\max} = Z_L = R_L + jX_L$ termination couple corresponding to $G_{T\max}$ and $Z_{Lreq} = R_{Lreq} + jX_{Lreq}$, $Z_{Sreq} = R_{Sreq} + jX_{Sreq}$ termination couple corresponding to G_{Treq} are evaluated in terms of operation conditions such as configuration type, bias conditions and frequency. So that microwave amplifier design can be easily achieved over whole operation frequency band.

c) Identifying all possible terminations systematically realizing performance triplets with a parametric step size over the stable regions.

d) Defining input/output matching circuits by concerning the termination calculated with a "impedance parameter approach" procedure.

e) Proposing the first block structure based on Neural Network, performance analysis of a Microwave transistor and designing input/output matching circuits that will be used the synthesis of Active Microwave circuits.

All the work is done by computer programs and the outputs of programs are given in the related chapters.

1. GİRİŞ

1.1 Tezin Kapsamı

Yapılan bu doktora tez çalışmasında, bir mikrodalga transistorun Yapay Sinir Ağı (YSA) ile performans analiz ve giriş/çıkış uydurma devrelerinin tasarımına yönelik yeni bir yöntem sunulmuştur. Dolayısıyla bu tez; YSA, performans analizi ve uydurma devre tasarımı olmak üzere üç ana bölümden oluşmaktadır.

Bilgisayar dünyasındaki baş döndürücü gelişmeler, bilgisayarın mikrodalga devre tasarım ve optimizasyon işlemlerinde kullanımı kaçınılmaz hale gelmesine neden olmuştur. Devre modelinin fiziksel elemanlarla gerçekleştirilmeden bilgisayarla simule edilebiliyor olması tasarımcıya zaman, maliyet ve işgücü açısından büyük avantajlar sağlamaktadır. Bu tezde ise, aktif mikrodalga devrelerinde, programlanmış yapay sinir ağı (YSA) yardımıyla yapılan doğru ve etkin yeni bir analiz ve modelleme yöntemi ortaya konulmaktadır. Burada YSA aktif devre elemanı (transistor) simule etmek için kullanılmıştır.

Yapılan çalışmada, aktif devre elemanı (transistor) üretici verileriyle, YSA optimum sürede eğitilmektedir. Yapay sinir ağı eğitildikten sonra, klasik kestirim yöntemleri yerine kullanılarak, istenilen veriler kolay bir şekilde ve hızlıca elde edilebilmektedir. Böylelikle önerilen YSA modeli, optimizasyon prosedürü esnasında elemanın fizik denklemlerinin tekrar tekrar çözülmesini gerektirmeyecek ve klasik optimizasyon tekniklerinin ortak sorunu olan "başlangıç değer kestirimi" problemini ortadan kaldırarak tercih edilir duruma gelmektedir. Modelleme esnasında bir döngü içerisinde defalarca optimizasyona ihtiyaç duyulduğu düşünüldüğünde, önerilen yöntemin başarısı kolaylıkla ortaya çıkmaktadır.

Önerilen modelde, herhangi bir mikrodalga transistorun S ve gürültü parametreleri yapay sinir ağı çıkışı, frekans, kutuplama akımı ve gerilimi ile konfigürasyon tipi yapay sinir ağı girişini oluşturmaktadırlar. Böylelikle YSA, frekans, kutuplama ve konfigürasyon tipi parametreleriyle, S ve gürültü parametreleri arasında bir eşleşme yaratarak; eleman denklemleri çözülmeden optimizasyon işlemlerini yapma olanağı sağlamaktadır. Bu çalışmada, YSA üretici tarafından verilmeyen gürültü parametrelerini kestirebilmek için kullanılmıştır.

YSA`dan elde edilen veriler mikrodalga transistorun performans analizi işleminde kullanılacaktır. Son yıllarda düşük gürültülü mikrodalga transistor kuvvetlendirici ve onların tasarım tekniklerinde artan bir ilgi oluşmaktadır. Temel talep, belirli bant genişliği içinde

minimum gürültü (Fmin), minimum giriş VSWR (=1) ve yüksek kazançtır.

Transistorun performans analizinde kullanılan ve GÜNEŞ metoduna dayanan optimizasyon problemi, $F_{req} - F(R_S, X_S) = 0$ ve $V_{ireq} - V_i(R_S, X_S, R_L, X_L) = 0$ olmak üzere ($G_{T\max}$, maksimum kazanç ve G_{Treq} istenilen kazanç), $G_{T\max} - G_T(R_S, X_S, R_L, X_L) = 0$ ve ayrıca $G_{Treq} - G_T(R_S, X_S, R_L, X_L) = 0$ olacak şekilde iki ayrı durum için kararlı bölgede kalmak koşuluyla mümkün tüm $Z_S = R_S + jX_S$ ve $Z_L = R_L + jX_L$ kompleks sonlandırmalarını geometriksel ve matematiksel olarak belirlemek şeklinde ifade edilebilir. Bu durumda bu işlemler sonucunda; $\{F_{req}, V_{ireq}, G_{T\max}[f, V_{CE}, I_C]\} \Leftrightarrow Z_{L\max}[f, V_{CE}, I_C]$, $Z_{S\max}[f, V_{CE}, I_C]$ ve $\{F_{req}, V_{ireq}, G_{T\min} \le G_{Treq} \le G_{T\max}[f, V_{CE}, I_C]\} \Leftrightarrow Z_{Lreq}[f, V_{CE}, I_C]$, $Z_{Sreq}[f, V_{CE}, I_C]$ fonksiyonlarına ulaşılmış olacaktır. Burada f; frekansı ve V_{CE} ile I_C (veya V_{DS} ile I_{DS}) kutuplama noktasını dolayısıyla transistorun gürültü ve S-parametrelerini ifade etmektedir.

Bu tür yüksek dereceli polinomların içerildiği optimizasyon problemlerin çözümüne yönelik bir takım yöntemler literatürde sıkça yer almaktadır ancak tanımladığımız problemin hem reel hem de imajiner kısma sahip olması ve ayrıca çok hesaplama zamanına ihtiyaç duymaları önerilen geometriksel yöntemi daha avantajlı duruma getirmektedir. Bunun yanında bu tezde tüm çözümler hem analitik hem de geometriksel olarak ifade edilmişlerdir.

Kullanılan yöntem aşağıdaki adımlardan oluşmaktadır:

a) Verilen bir transistor ilişkin F=sabit, V_i=sabit ve G_T=sabit dairelerine yönelik Zs-düzleminde analiz yapılır. Bu adımda gürültü ve VSWR dairelerinin Zs-düzlemindeki konumları belirlenmiş olur. Bu dairelerin durumları $G_{T\max}, G_{Treq}, Z_S$ ile Z_L değerlerini belirlenmesine yardımı olacaktır.

b) Zi-düzleminde yapılacak analizlerde kararlılık bölgeleri belirlenir. Ayrıca burada kararlı bölgede belirlenecek Z_i değeri bir Z_L'e ve Zs-düzleminde F ve V_i dairelerinin durumuna göre bir veya iki Z değerine karşılık düşürülecektir.

c) $G_{T\max}$ ve G_{Treq} için tüm mümkün Z_S ve Z_L değerleri kararlı bölgede kalacak şekilde belirlenir.

d) Bu işlemler tüm çalışma bandı frekanslarında yapılarak analiz tanımlanmış olur .

Yukarıda anlatılan adamlarla modellemeye temel oluşturacak veriler elde edilmiş olmaktadır.

Haberleşme sistemlerinin tasarımında sıkça karşılaşılan temel problemlerden birisi, sayısal veriler ile tanımlanmış bulunan fiziksel eleman ve yapıların devre modellerinin çıkarılmasıdır. Özellikle mikrodalga frekanslarında çalışan sistemlerde, yüksek frekans transistorları, anten, yükselteç ve güç kaynağı gibi ön, ara ve son kat birimlerinin karakteristikleri genelde sadece ölçüm verileri ile tanımlanmaktadır. Bu tür birimleri içeren yüksek frekans sistemlerinde, katlar arasında güç transferini kontrol etmek için ihtiyaç duyulan filitre ve uydurma devreleri tasarımlarında, bilinen analitik ve yarı analitik yöntemlerin kullanılması, sayısal veriler ile tanımlı elemanların, gerçekleşebilir devre fonksiyonları veya devre elemanları ile modellenmiş olmasını gerektirmektedir. Bu anlamda, mevcut analitik tasarım yöntemlerinin uygulanabilmesi için gerek haberleşme sistemlerinde gerekse elektronik mühendisliğinin diğer disiplinlerinde karşılaşılan sayısal verilerle tanımlanmış yapıların, kayıplı veya kayıpsız devre elemanları ile modellenmesi büyük önem taşımaktadır.

Sayısal verilerden devre modellenmesine ihtiyaç duyulan tipik uygulamalardan bir tanesi klasik geniş bantlı uydurma devre tasarımıdır. Bu problemde,verilen bir kaynak ile yük arasında, mümkün olduğunca geniş bir frekans bandı üzerinde, maksimum güç transferini sağlayacak kayıpsız elemanlardan oluşan bir, iki kapılı uydurma devresinin belirlenmesi söz konusudur. Burada kaynak ve yük empedansları uygulamaya bağlı olarak bizim durumumuzda olduğu gibi sayısal veriler ile tanımlanmış olabilmektedir.

Literatürde veri modelleme çalışmaları çok eskilere uzanmaktadır. Ancak, mevcut yaklaşımlar ile modellemenin güçlüğü ve sonuçların söz konusu uydurma problemlerinde istenen hassasiyet açısından yetersiz kalması nedeni ile, çalışmalar doğrudan sayısal veriler ile tanımlı yüklerden çalışabilen alternatif saf sayısal veya yarı-analitik bilgisayar destekli yaklaşımlar üzerine yoğunlaşmıştır. (Carlin H.J., Yarman S.B., 1983) tarafından başlatılan ve Reel Frekans Teknikleri diye adlandırılan bu yeni yarı-analitik çözüm yöntemleri daha sonra birçok araştırmacı tarafından geliştirilmiş ve analitik yöntemlerde kaçınılmaz olarak ihtiyaç duyulan veri modelleme problemindeki güçlükleri ortadan kaldırmıştır. Ancak, verimli ve hassas veri modelleme yöntemlerinin geliştirilmesi ve analitik yöntemler ile uydurma problemlerinin çözümü halen araştırmacılar tarafından çözüm bekleyen temel problemlerden birini teşkil etmektedir.

Pratikte, bir yük, ilgilenilen frekans bölgesinde ölçülmüş empedansın reel-sanal kısımları olarak tanımlanabilmektedir. Bu şekilde tanımlı sayısal yük verilerinin bir devre yapısı ile modellenmesi, temel olarak devre fonksiyonlarının gerçeklenme koşulları sağlanacak şekilde elde edilmesini içermektedir. Bu çalışmada, sayısal veriler ile tanımlı fiziksel bir yapının

kayıplı bir tek kapılı devre olarak veya kayıpsız elemanlardan oluşan bir iki kapılı devre ile modellenmesi incelenmiştir.

Verilen bir giriş empedansının modellenmesinde iki yöntem çok sık olarak kullanılmıştır.

1). Bir devre topolojisi belirlenerek verilere en iyi uyan eleman değerlerinin belirlenmesi.

2). Verileri en iyi temsil eden empedans veya saçılma parametrelerinin belirlenmesi ve bu parametrelerin sentezi ile eşdeğer devrenin bulunması.

Modelleme yapmak için bir devre topolojisi seçildiğinde, bir optimizasyon paketi yardımıyla verilere en iyi uyan eleman değerleri belirlenir. Bu çok basit ve düz bir yöntem olmasına rağmen bazı zorluklar içermektedir: Yapılacak optimizasyon işlemi eleman değerleri açısından oldukça lineerlikten uzaktır bu nedenle bölgesel bir minimuma ulaşılabilir veya optimizasyon hiç yakınsamayabilir. Böylesi zor lineer olmayan optimizasyon problemlerinin iyi sonuç verebilmesi için başlangıç değerinin çok iyi seçilmiş olması gerekir. Oysa bu başlangıç değerini bulmak o kadar kolay değildir. En önemlisi, problemin doğasına uygun olarak topoloji nasıl seçilecektir? Bu belirgin değildir. Bu nedenle modelleme yapan kişi deneme yanılma yöntemiyle bir çok topoloji seçecek, sonuçta en iyi optimizasyonu veren topolojiyi model olarak alacaktır veya tesadüfen de olsa uygun topoloji denk düşmezse problem çözümsüz kalacaktır.

Verileri en iyi temsil eden empedans parametrelerini bulmak için birçok yöntem uygulanmıştır. Bizim durumumuzda bu değerlerin transistorun performans analizi programından geldiğini daha önce belirtmiştik. Bu veriler, pozitif reel bir z(s) empedans fonksiyonu ile modellendiği zaman bu z(s) empedans fonksiyonu sentez edilebilir. Ölçülen verilerdeki reel ve sanal kısımları ayrı ayrı modellemek, Empedans Yaklaşımı ile Modelleme yöntemimizin esasını oluşturur. Reel ve sanal kısımları ayrı ayrı modellemek için, uygulanan empedans yaklaşımına ilişkin adımlar ileriki bölümlerde anlatılacaktır.

Bir mikrodalga transistorun YSA ile performans analizi ve modellenmesi konusunda ilk defa bu tezde ortaya konulan noktalar aşağıda sıralanmıştır.

a) Transistorun, klasik küçük-işaret eşdeğer devresi yerine YSA eşdeğeriyle, çalışma koşullarının (konfigürasyon tipi (CT), kutuplama akım (I_C) ve gerilimi (V_{CE}), frekans (f)) fonksiyonu olarak davranışının anında tayini ve bunun uygulamada kullanılması.

b) $F_{req} \geq F_{\min}$, $V_{ireq} \geq 1$ ve $G_{T\min} \leq G_{Treq} \leq G_{T\max}$ koşulları altında herhangi bir

$(F_{req},\ V_{ireq},\ G_{Treq})$ uyumlu performans üçlüsü ile birlikte, $G_{T\max}$'ye karşılık gelen $Z_{L\max} = Z_L = R_L + jX_L$, $Z_{S\max} = Z_S = R_S + jX_S$ ve G_{Treq}'a karşılık gelen $Z_{Lreq} = R_{Lreq} + jX_{Lreq}$, $Z_{Sreq} = R_{Sreq} + jX_{Sreq}$ sonlandırmalarının, transistorun çalışma koşullarının (konfigürasyon tipi (CT), kutuplama akım (I_C) ve gerilimi (V_{CE}), frekans (f)) fonksiyonu olarak tayin edilip hesaplanması.

c) Uyumlu performans üçlüsünü gerçekleyen kararlı bölgelerde istenilen adım aralığıyla mümkün tüm sonlandırmaların sistematik olarak tayin edilip hesaplanması.

d) Bulunan sonlandırmaların sistematik bir yöntemle (empedans parametreleri yaklaşımı) giriş/çıkış uydurma devreleri tasarımında kullanılması.

e) Bir mikrodalga transistorun için YSA, performans analizi ve giriş/çıkış uydurma devrelerinin elde edilmesi şeklinde bir blok yapının Aktif Mikrodalga devre sentezinde kullanılmak üzere sunulması.

Tüm çalışmalar bilgisayar programlarıyla yapılmıştır ve programlarla çıktıları ilgili bölümlerde verilmiştir.

1.2 Kaynak Araştırması

1970'lerden başlamak üzere günümüze değin bilgisayar destekli mikrodalga tasarımında önemli gelişmeler sağlanmıştır. Artık bilgisayar kullanımı mikrodalga devre tasarımı süresince kaçınılmaz bir araç durumuna gelmiştir. Bilgisayar kullanımıyla, zaman ve maliyet probleminden kurtulup, devreyi simule etme olanağına kavuşulmaktadır (Blaavw, Gerrit A. ve Brodes, Frederic P).

Bilgisayar bu kadar yaygın ve kullanışlı olmadığı zamanlarda da optimizasyona ihtiyaç duyulmuştur. Bu durumda araştırmacılar optimizasyon işlemine yönelik birçok yöntemler önermişlerdir (Kenneth W. Cattermale, 1985), (Arthur E. Bryons, Jr. Mu-Chi-Ho (1975), (Kenneth W. Cattermule, Sohn &.O'Resilly, 1984) ve (Björn Albinson, 1990). Bilgisayarın optimizasyon işleminde kullanılabileceği ortaya çıktığında, bilgisayar yardımıyla tasarım ve analiz konusuna ağırlık verilmeye başlanmıştır (Dobrowolski Janusz A., 1991) , (Stephen A. Maas, 1988). Bilgisayarla parametre ve nümerik analiz yöntemleri üzerinde bazı tezler yazılmıştır (Tepe Cemal, 1991) ve (Tepe Cemal, 1994).

Gelişen teknoloji karşısında bilgisayarla yapılan optimizasyon işlemlerinin birtakım eksikliklerinin olduğunun ortaya çıkması, araştırmaların yeni metotlarla geliştirilmesine

zorlamıştır. Biyolojik sinir ağının çalışma prensibinin programlanabileceği ortaya çıktığında, optimizasyon işleminin Yapay Sinir Ağı (YSA) ile yapılabileceği noktasına gelinmiştir (Cichocki, A ve Unbehauen R., 1993), (Hykin Simon, 1994) ve (Werbos, P. John, 1994). Zaabab A.Hafid, Zhang Qi-Jun, Nakhla Michel, (1995) önerdiği devre optimizasyonuna yönelik YSA yaklaşımıyla, çalışmalar hızlanmış ve bu konuda birçok yayın ortaya çıkmıştır (Güneş, F., Gürgen, F., Torpi, H., 1996), (Torpi Hamid, 1997), (Bandler John, Shauhuacher W., 1998), (Veluswami Anand, Nakhla Michel S. and Zhang Qi-Jul, 1997), (Zaabab A. Hafid, Zhong Qi-Jun, Nakhla Michel S., 1995) ve (Wamg Fang, Zhang Qi-Jun, 1997).

Bu gelişmelere paralel olarak mikrodalga devrelerde de birtakım gelişmeler kaydedilmiştir. (Güneş Macit, 1980) tarafından geliştirilen mikrodalga transistorun geometriksel yöntemle performans analizi üzerinde yapılan çalışmalarla söz konusu bu yöntem geliştirilmiştir (Güneş Filiz, Güneş M., Fidan M., 1994) ve (Güneş F., Çetiner B.A., 1998).

YSA ve mikrodalga alanındaki bu gelişmelerle birlikte veri modellemesi ve uydurma devre tasarımında da yeni gelişmeler kaydedilmiştir (Carlin H.J., Yarman S.B., 1983) tarafından ortaya konulan uydurma devre tasarımı yöntemlerini (Yarman S.B., Feltuvies A., 1990), (Yarman S.B., Akşen A., 1992) ve (Kılınç Ali, 1995) izlemiş ve böylelikle yeni gelişmeler ortaya konulmuştur.

1.3 İçerik Düzenlemesi

Tezde bulunan bölümlerde içerilen bilgiler aşağıda kısaca özetlenmiştir.

Bölüm 1'de tez konusunda, şu ana kadar yayınlanmış kaynaklardan bahsedilerek konuya giriş yapılmıştır.

Bölüm 2'de Yapay Sinir Ağları hakkında genel bilgiler verilerek tezde kullanılan YSA tanıtılmıştır.

Bölüm 3'de çözümü hedeflenmiş optimizasyon problemi ifade edilmiş ve mikrodalga transistorun performans analizi ortaya konarak sonlandırmaların bulunabilmesi için kullanılan yöntemlerden söz edilmiştir. Yine bu bölümde transistorun kararlılık durumundan ve çalışma bölgelerinden bahsedilmiştir.

Bölüm 4'de 3. Bölümde bahsedilen sonlandırma verilerinden uydurma devrelerini elde edebilmek için kullanılan yöntem tanıtılmıştır.

Bölüm 5'de Bölüm 2-4'de verilen yöntemlerin uygulanabilmesi için geliştirilen programlar

tanıtılmış ve tezin kapsamı blok yapılarla ifade edilerek geliştirilen ve kullanılan sistem ortaya konulmuştur.

Bölüm 6'da üç farklı transistor verileri için uygulama yapılmış ve sonuçlar verilmiştir.

Daha sonraki bölümlerde ise sonuçlar ve önerilerden bahsedilerek kaynaklar, ekler ve özgeçmiş verilmiştir.

2. YAPAY SİNİR AĞLARI

2.1 Genel Bilgiler

2.1.1 Yapay Sinir Ağı Tanımı

Yapay sinir ağı (YSA) yeni bir bilgi işleme tekniğidir. YSA bilgisayar tabanlı sinir sistemi olarak düşünülebilir ve klasik hesaplama tekniklerinden oldukça farklı çalışır. Matematiksel bir tabana sahiptir ve eğitmek için kullanılacak geçerli bir istatiksel kümeye ihtiyacı vardır. Biyolojik Sinir Ağları sayesinde akıllı sistemlerde sayısız gelişmeler olmuştur. Birçok bilim dalındaki araştırmacılar örüntü (pattern) tanıma, kestirme, optimizasyon, ilişkili bellek (associative memory) ve kontrol konularında bazı problemleri çözebilmek için YSA tasarlamaya çalışmaktadırlar (Simon, 1994).

Bu tür problemleri çözmek için bazı klasik yaklaşımlar önerilmiştir. Bazı durumlarda bu tür problemlerin çözümüne yönelik birçok başarılı uygulama bulunmasına rağmen, hiçbiri kendi domenleri dışında yeterince esnek değildir. Bu konuda YSA oldukça geniş bir esneklik sağlamaktadır. Aşağıdaki birçok karakteristik özellik, Von Neumann veya modern bilgisayarda yeterince içerilmemektedir (Marilyn ve Illigworth, 1991). Bu karakteristik özellikler yoğun paralellik (massive parallelism), dağıtılmış gösterilim ve hesaplama, öğrenme kabiliyeti, genelleştirme yeteneği, uyum sağlama, bilgi işleme, hata analizi ve düşük enerji tüketimi (kaynak sarfiyatı), vs. YSA dayalı cihazların yukarıdaki belirtilen karakteristik özelliklerden bazılarını işleyebileceği ümit edilmektedir. Modern sayısal bilgisayarlar, nümerik hesaplamalarda insanlardan daha üstün durumdadırlar. Diğer taraftan insanlar herhangi bir çaba göstermeksizin bir insanın yüzünü rahatlıkla algılayabilir. Bunu bilgisayarla gerçekleştirmek oldukça güçtür. Gösterilen performanstaki bu ciddi fark, yapay sinir ağı mimarisi ile Von Neumann mimarisinin tamamen farklı olmasından kaynaklanmaktadır (Çizelge 2.1)

Çizelge 2.1 YSA ile Von Neumann bilgisayarın karşılaştırılması

Özellik	**Von Neumann**	**YSA**
İşlemci	Karmaşık	Basit
	Yüksek hız	Düşük hız
	Bir yada birkaç tane	Çok miktarda
Hafıza	İşlemciden bağımsız	İşlemciye entegre

	Yerelleştirilmiş İçerik adreslenemeyen	Dağılmış İçerik Adreslenebilir
Hesaplama	Merkezileşmiş Sıralı Yüklü programlar	Dağılmış Paralel Kendiliğinden öğrenme
Güvenilirlik	Çok Hassas	Güçlü
Deneyim	Nümerik ve sembolik marifetli	Algılama problemleri
Çalışma ortamı	İyi tanımlanmış sınırlar iyi belirlenmiş	İyi tanımlanmamış sınırlandırılmamış

Merkezi mimarili Von Neumann'a dayalı programlara birçok akıl kazandırma çalışmalarına rağmen, bu çalışmalar genel bir akıllı program elde etme şeklinde sonuçlanmamıştır (Marilyn ve Illigworth, 1991). Biyolojik sinir ağı yardımıyla, YSA oldukça çok sayıda basit işlemciler ve bağlantılar sayesinde yoğun paralel hesaplamalar yapabilir duruma getirilmiştir.

Yapay sinir ağı (YSA) girişindeki bilgiyi çok boyutlu ve lineer olmayan bir fonksiyon üzerinden çıkıştaki bilgiye eşleştiren bir modeldir. Giriş uzayı X n-boyutlu bir vektör (devre eleman parametrelerinden oluşuyor olabilir) bu vektör çıkış uzayında, aradaki gizli katmalar üzerinden, p-boyutlu bir Y vektörüne (yine devre eleman parametreleri olabilir) karşılık düşürülüyor.

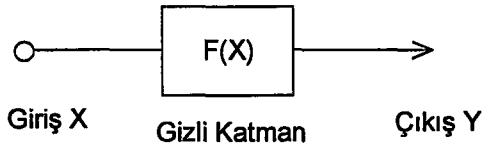
X ile Y arasındaki ilişki çok boyutlu ve lineer olmaması özelliğiyle YSA modelleri diğer eşleştirme metotlarına göre avantajlı olmaktadır.

Bu ilişki matematiksel olarak,

$$Y=F(X) \tag{2.1}$$

şeklinde tanımlanabilir. Buradaki F fonksiyonu normalde basit ve hesaplamalarda düşük belleğe ihtiyaç duyacak yapıda olmak zorundadır. Geleneksel modellemelerde ve simülasyon tekniklerinde bu F fonksiyonu nümerik hesaplamalar ve analizlerle hesaplanır. YSA yaklaşımında ise bu fonksiyon bir YSA kullanılarak gerçekleştirilir ve nümerik hesap ve analizlere ihtiyaç kalmaz.

Şekil 2.1'den de görülebileceği gibi n-boyutlu X giriş vektörü bir gizli katman üzerinden p-boyutlu Y çıkış vektörünü oluşturmaktadır.



Şekil 2.1 YSA blok gösterilim

2.1.2 Uygulama Alanları

Aşağıda YSA'la ilgili güncel problemler ve araştırma alanları sıralanmıştır (Marilyn ve Illigworth, 1991).

a) Örüntü Sınıflama (Pattern Classification); Örüntü sınıflama işleminde amaç bir giriş örüntüsünü (bir konuşma dalga formu veya bir el yazısı gibi) daha önceden belirlenmiş birçok sınıftan bir tanesine atamaktır (Şekil 2.2a).

b) Gruplama; gruplamada bilinen sınıf etiketine yönelik eğitim verisi olmaz. Ayrıca bu öğreticisiz (unsupervised) örüntü sınıflama olarakta bilinir. Gruplama algoritması örüntüler arasındaki benzerliklere bakarak benzer örüntüleri gruplar (Şekil 2.2b).

c) Fonksiyon Uydurma; n çiftten oluşan bir eğitim örüntüsü grubunun $\{(x_1,y_1),(x_2,y_2),.....(x_n,y_n)\}$ bilinmeyen bir $\mu(x)$ fonksiyonundan (gürültü ile ilgili) yaratıldığını düşünelim. Buradaki amaç bilinmeyen $\mu(x)$ fonksiyonunun belirli bir hata ile $\mu(x)$olarak tahmin edebilmektir. Birçok mühendislik ve bilimsel modelleme problemleri fonksiyon uydurmaya ihtiyaç duyarlar (Şekil 2.2c).

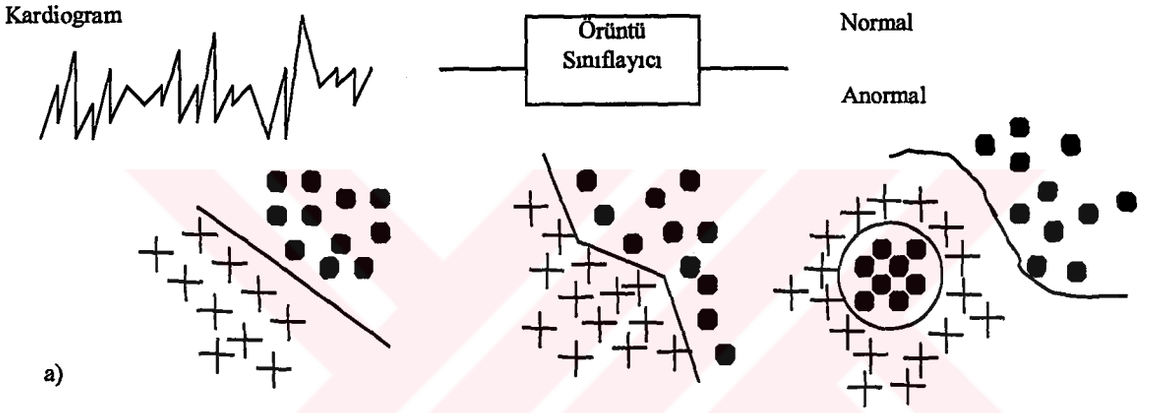
d) Kestirme (Prediction); bu tezin konusu olan kestirim de YSA kullanılmaktadır. $t_1,t_2,....t_n$ anlarında örnek değerler $y(t_1),y(t_2),.....y(t_n)$ olduğunu düşünürsek buradaki amaç gelecekte t_{n+1} anındaki $y(t_{n+1})$'ı kestirmektir (Şekil 2.2.d). Pazara yönelik stok kestirimi ve hava tahmini gibi durumlarda bu yöntem sıkça kullanılmaktadır.

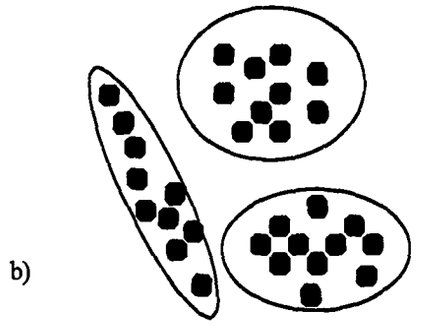
e) Optimizasyon; YSA optimizasyonundaki amaç, amaç fonksiyonunu maksimize veya minimize eden ve bir grup sınırlamaları sağlayan bir çözümü bulmaktır. Seyyar satıcı problemi (travelling salesman problem) klasik örneklerden birisidir (Şekil 2.2e).

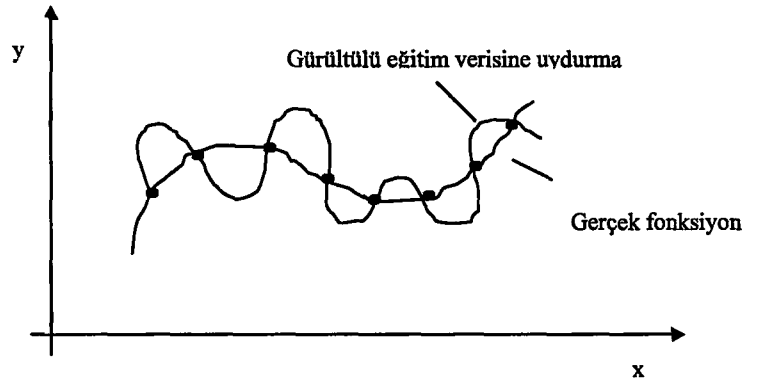
f) İçerik Adreslenebilir Bellek; Von Neumann hesaplama yapısında bellekteki bir bölgeye sadece adres yolundan ulaşılarak elde edilebilir ve adresin içeriğinden bağımsızdır. Adreslemede herhangi bir ufak hata tamamen farklı bir içeriğin elde edilmesine neden

olacaktır. İlişkili veya içerik adreslenebilir bellekte ise adından da anlaşılacağı gibi direkt içerik adreslemesi söz konusudur. Bellek içeriği girişin eksik olması veya içeriğin bozuk olması durumunda bile başarılı sonuç verebilir (Şekil 2.2f).

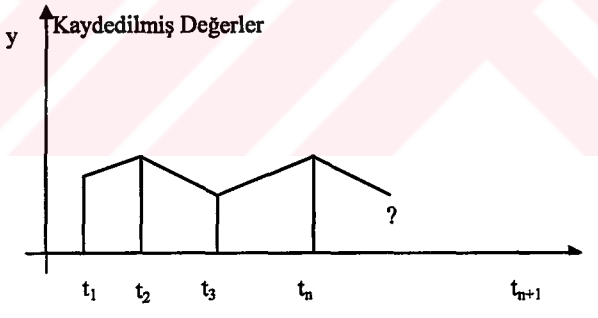
g) Kontrol; t anında y(t) çıkışı ve u(t) kontrol girişi ile tanımlanan bir dinamik sistemi düşünelim. Buradaki amaç referans modeli tarafından yaratılan, istenilen bir hedefe sistemi yönlendirecek bir kontrol girişi yaratmaktır (Şekil 2.2g). Motor hız kontrolü buna bir örnektir.



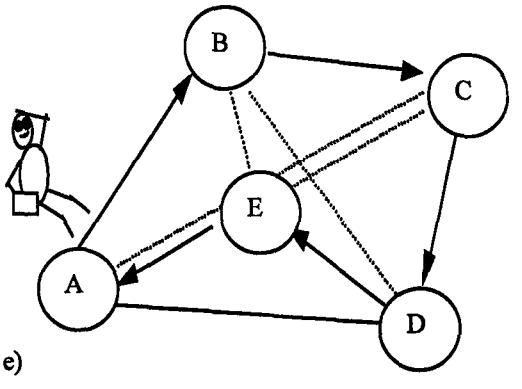



y
Gürültülü eğitim verisine uydurma
Gerçek fonksiyon
x

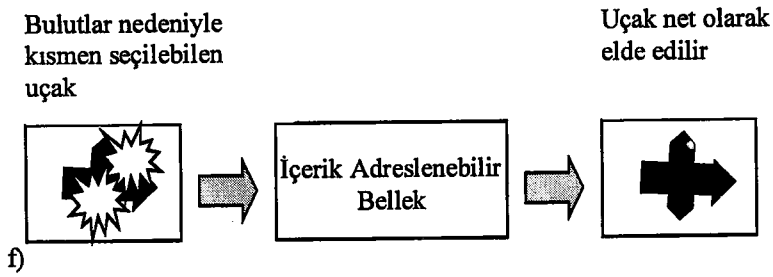
c)

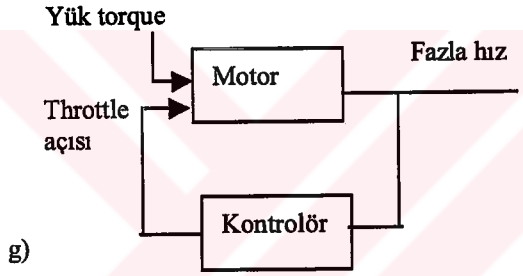

y
Kaydedilmiş Değerler
?
t1
t2
t3
tn
tn+1

d)


B
C
E
A
D

e)





Şekil 2.2 YSA'nın gerçekleştirebileceği görevler

a) Örüntü sınıflama

b) Gruplama

c) Fonksiyon uydurma

d) Kestirim

e) Optimizasyon

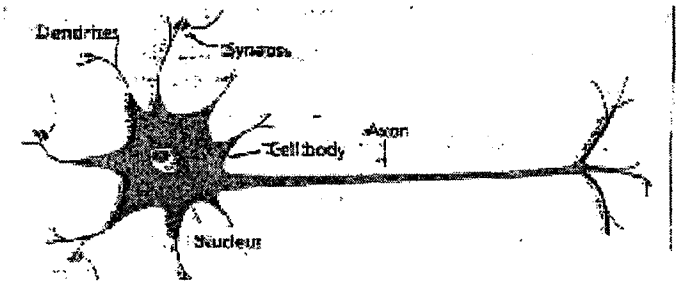
f) İçerik adreslenebilir bellek

g) Kontrol

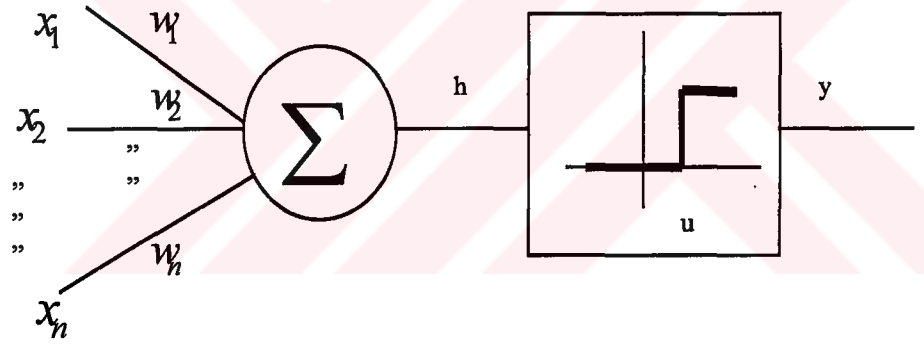
2.1.3 Yapay Sinir Ağı Model Yapısı ve Mimarisi

McCulloch ve Pitts (Simon, 1994), bir yapay sinir hücresi için hesaplama modeli olarak, bir binary threshold (ikili eşik) işlem elemanı önerdiler (Şekil 2.3.). Bu matematiksel sinir

hücresi girişindeki x_j (j=1,2,...,n) işaretlerini ağırlıklı toplama tabi tutarak, çıkışta 1 veya 0 üretir. Eğer toplam, u eşik değerinden büyük olursa 1, aksi taktirde 0 sonucuna ulaşılır.



a)



b)

Şekil 2.3 Yapay sinir hücresi

a) Biyolojik yapay sinir hücresi

b) Mc Culloch ve Pitts modeli

Matematiksel olarak;

$$y = \theta\left[\sum_{j=1}^{n} w_j x_j - u\right] \tag{2.2}$$

şeklindedir. Burada θ(.) birim basamak fonksiyonudur. Notasyon basitliği için genelde u eşik değeri, girişinde $x_0=1$ olan $\omega_0=-u$ şeklinde bir ağırlık düşünülür. Pozitif ağırlıklar excitatory (uyarıcı) sinaplara, eksi ağırlıklar ise inhabitary (yasaklıyıcı) sinaplara karşılık düşmektedir. McCulloch ve Pitts genel olarak uygun ağırlıkların seçilmesinin bu tür nöronların üniversal hesaplama yapabilmelerini sağladığını ispatladılar. Biyolojik sinir hücresi ile karşılaştırma yapıldığında iç bağlantıların aksonlar ve dendritleri modellendiği, ağırlıkların sinapları gösterdiği ve eşik fonksiyonunda somadaki aktiviteyi gerçekleştirdiği kolaylıkla görülebilir. McCulloch ve Pitts işleri basitleştirmek için bir takım kabullerde bulunduğundan biyolojik sinir hücresini tam olarak temsil edememektedir. McCulloch ve Pitts modeli birçok alanda genelleştirilmiştir. Bunlardan en çok bilineni eşik fonksiyonu yerine piecewise (parçalı) lineer, sigmoid ve gauss gibi bazı aktivasyon fonksiyonlarının kullanımıdır (Şekil 2.4). Sigmoid fonksiyonu YSA'da oldukça çok kullanılmaktadır. Standart sigmoid fonksiyonu aşağıdaki şekilde tanımlanır (Simon, 1994).

$$g(x) = 1/(1+\exp\{-\beta x\}) \tag{2.3}$$

β eğim parametresidir.

Şekil 2.4 Farklı aktivasyon fonksiyonları

a- Threshold (eşik)

b- Piecewise (parçalı) lineer

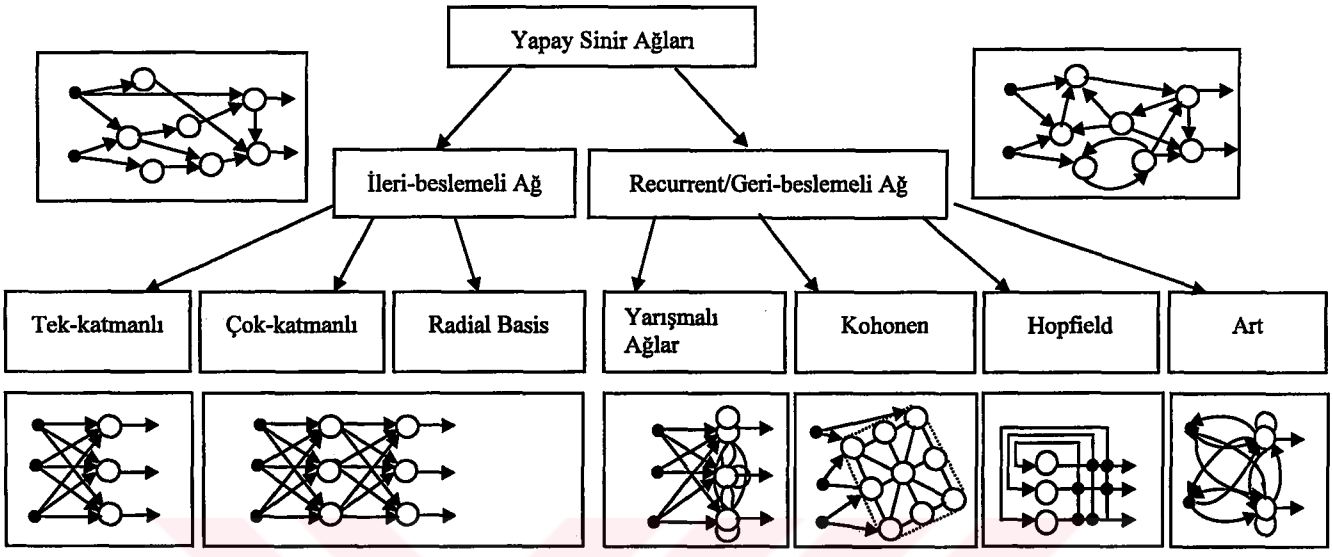
c- Sigmoid

d- Gaussian

YSA, girişi çıkışa bağlayan yapay sinir hücrelerinin oluşturduğu düğümlerle, düğümleri birbirine bağlayan ağırlıklardan oluşmaktadır (Şekil 2.3). Bağlantı yapısına bağlı olarak YSA iki grupta toplanabilirler (Şekil 2.5.) (Simon, 1994).

a) İleri-Beslemeli (Feed-forward) ağ; çevrim yok

b) Geri-Beslemeli (Feedback) , rekürsif (recurrent), çevrim var



Şekil 2.5 YSA'nın gruplandırılması

Çok katmanlı algılayıcı (Multi-layer perceptron) olarak bilinen ileri beslemeli YSA'da yapay sinir hücreleri aralarında tek yönlü bağlantı olacak şekilde düzenlenirler (Şekil 2.5). Her bir grup için tipik YSA da gösterilmektedir. Farklı bağlantılar farklı ağ davranışlarını ortaya koyar. Genel olarak bizimde konu edeceğimiz YSA statiktir. Yani verilen bir girişten arka arkaya birçok çıkış yerine sadece bir grup çıkış üretir. İleri-beslemeli YSA belleksizdir, yani bir girişe cevabı ağın daha önceki durumdan bağımsızdır. Diğer taraftan rekürsif YSA dinamik bir sistemdir. Her bir girişten sonra yeni bir çıkış üretilir ve geri-besleme yolunun olması nedeniyle ağda yeni bir duruma çekilir. Yine geri-besleme yolundan dolayı ağın bu yeni durumu bir sonraki çıkışı etkileyecektir. Farklı ağ yapıları, farklı öğrenme algoritmalarına ihtiyaç duyarlar. Bir sonraki bölümde öğrenme algoritmaları incelenecektir.

2.1.4 YSA Öğrenme Algoritmaları

YSA'daki öğrenme yeteneği akıllığın temelini oluşturmaktadır. Öğrenmeyi formüle etmenin oldukça zor olması yanında, YSA'da öğrenme işlemi, belirli bir görevi etkin bir şekilde yerine getirebilmek için ağ yapısının ve bağlantı ağırlıklarının güncelleştirilmesi şeklinde düşünülebilir. YSA genelde ağırlık değerlerini eğitme örüntülerinden öğrenir. Ağırlık

değerleri güncelleştirildikçe YSA performansı da artırılmış olur. YSA'nın örneklerden öğreniyor olması oldukça dikkat çekmesine neden olmuştur. Belirlenmiş birtakım kuralları izlemektense YSA giriş çıkış ilişkisinden gerekli bilgiyi öğrenerek problemi çözmeye çalışır. Bu durum YSA'yı klasik sistemlerden ayıran en önemli özelliktir. Bir öğrenme işlemi tanımlayabilmek için ilk olarak öğrenme algoritmasına (ağırlıkları ayarlayabilmek için hangi öğrenme kurallarının kullanılacağına) ihtiyaç vardır.

Üç temel öğrenme modeli vardır (Çizelge 2.2) (Marilyn ve Illigworth, 1991). Öğreticili (supervised), öğreticisiz (unsupervised) ve karışık (hybrid). Bu tezinde konu edindiği öğreticili öğrenimde YSA her bir giriş örüntüsü için doğru bir cevapla desteklenir. Ağırlıklar, YSA cevabının doğru cevaba mümkün olduğunca yakın olacağı şeklinde belirlenir. Diğer taraftan öğreticisiz öğrenme her bir giriş örüntüsü için doğru bir çıkışın YSA'na ek bir giriş olarak verilmesine ihtiyaç duymaz. Burada giriş örüntüleri arasındaki korelasyona (ilişkiye) bakılarak; çıkış oluşturulmaya çalışılır. Üçüncü tip öğrenme modeli olan karışık öğrenme, öğreticili ve öğreticisiz öğrenme modellerinin birleşiminden oluşmaktadır.

Çizelge 2.2 Öğrenme algoritmaları

Model	**Öğrenme Kuralı**	**Yapı**	**Öğrenme Algoritması**	**İşlem**
Öğreticili	Hata-düzeltme	Tek veya çok-katmanlı algılayıcı	Algılayıcılı öğrenme algoritmaları Geriye-yayılımlı Adaline ve Madaline	Örüntü Sınıflama Fonksiyon Uydurma Kestirme, kontrol
	Boltzmann	Geri-beslemeli	Boltzmann öğrenme algoritması	Örüntü sınıflama
	Hebbian	Çok-katmanlı ileri-beslemeli	Lineer ayrıştırma analizi	Veri analizi Örüntü sınıflama
	Yarışmalı	Yarışmalı ART Ağı	Vektör kuantalama Öğrenmesi ART Haritası	Sınıf içi gruplama Veri sıkıştırma Örüntü sınıflama

				Sınıf içi gruplama
Öğreticisiz	Hata-düzeltme	Çok-katmanlı İleri-beslemeli	Sammon öğrenmesi	Veri analizi
	Hebbian	İleri-beslemeli ya da yarışmalı	Temel eleman analizi	Veri analizi Veri sıkıştırma
		Hopfield Ağı	İçerik adreslenebilir bellek öğrenmesi	Adreslenebilir bellek
	Yarışmalı	Yarışmalı	Vektör kuantalama	Gruplama Veri sıkıştırma
		Kohonen SOM	Kohonen SOM	Gruplama Veri sıkıştırma
		ART ağları	ART1, ART2	Gruplama
Karışık	Hata-düzeltme ve yarışmalı	RBF ağı	RBF öğrenme algoritması	Örüntü sınıflama Fonksiyon uydurma Kestirme, kontrol

Öğrenme algoritması; YSA bir öğrenme algoritması kullanarak örneklerden öğrenirken; üç temel nokta önemli olmaktadır. Kapasite, örneklerin karmaşıklığı ve hesapların karmaşıklığı. Kapasite; kaç tane örüntünün kaydedileceği ve hangi fonksiyona ve sınırlara uyum sağlayacağının belirlenmesidir. Örnek karmaşıklığı; doğru bir sonucu garanti edebilmek için kaç tane eğitme örüntüsüne ihtiyaç olduğunu göstermektedir. Yeterince eğitme örüntüsünün olmaması YSA'nın istenilen sonucu vermesini engelleyebilir. Aynı zamanda gereğinden fazla eğitme örüntüsü de gereksiz yere zaman kaybına sebep olur.

Hesaplama Karmaşıklığı; Öğrenme algoritmasının eğitme örüntülerinden bir çözüm

çıkarabilmesi için ihtiyaç duyacağı zamanı göstermektedir. Varolan birçok öğrenme algoritmaları oldukça yüksek hesaplama karmaşıklığı içermektedirler. Günümüzde YSA için etkin algoritma tasarlama oldukça ilgi çeken bir konudur. Literatürde YSA'ya yönelik birçok öğrenme kuralı mevcuttur. Hata-düzeltme (Error Correction), boltzman, hebbian ve yarışmalı öğrenme (Competitive learning) gibi birçok öğrenme kuralı mevcuttur. Biz burada günümüzde sıkça kullanılan geriye yayılımlı (Back-propagation) algoritmasının da temeli olan hata-düzeltme kuralını konu edineceğiz. Hata-düzeltme kuralı bir öğreticili öğrenme modelidir. Öğrenme işlemi süresince YSA tarafından yaratılan y çıkışı, d istenilen çıkışa eşit olmayabilir. Hata-düzeltme öğrenme kuralının temel prensibi, ağırlıkları değiştirerek d-y hatasını azaltmaya dayanmaktadır.

Algılayıcı (Perceptron) Öğrenme ve Geriye-yayılımlı (Back-propagation) öğrenme kuralları, hata-düzeltme prensibine dayanmaktadır. Bir algılayıcı, ayarlanabilir ağırlıklar w_j (j=1,2,…,n) ve bir u eşik değeriyle birlikte tek bir sinir hücresi içermektedir. Bu durum Şekil 2.3'de görülmektedir.

Giriş vektörü $X=(x_1,x_2,.....,x_n)^t$ olması durumunda yapay sinir hücresindeki net giriş,

$$v=\sum_{j=1}^{n} w_j x_j - u \tag{2.4}$$

şeklinde olacaktır. Algılayıcıların çıkışı eğer $v>0$ ise +1 aksi taktirde ise 0'dır. Örneğin iki gruplu sınıflama probleminde algılayıcı, $y=1$ ise girişi bir gruba; $y=0$ ise girişi diğer gruba atayacaktır. Aşağıdaki eşitlik

$$\sum_{j=1}^{n} w_j x_j - u = 0 \tag{2.5}$$

uzayı ikiye bölen sınırları (decision boundary) belirler. Aşağıdaki verilen bir eğitme örüntü kümesine yönelik algılayıcı öğrenme algoritması görülmektedir.

Algılayıcı (Perceptron) öğrenme algoritması (Simon, 1994):

1- Ağırlıklara ve Eşik değerine rasgele küçük sayılar ata

2- $(x_1,x_2,.....,x_n)^t$ giriş vektör örüntüsünü YSA girişine ver ve çıkışı belirle

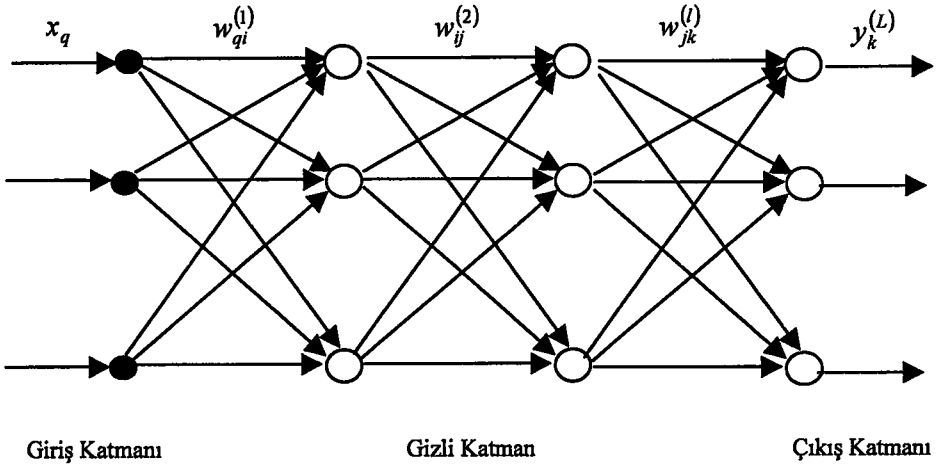
3- Ağırlıkları aşağıdaki şekilde güncelleştir

$$w_j(t+1) = w_j(t) + \eta(d-y)x_j \tag{2.6}$$

Burada d istenilen çıkış; t iterasyon sayısı; η (0<η<1) adım büyüklüğü, w_j (j=1,....,n) ağırlık değerleridir. Yukarıdaki algoritmada da görülebileceği gibi algılayıcı sadece hata anında öğrenmektedir. Algılayıcının birden fazla YSA katmanında kullanılmasıyla çok katmanlı algılayıcı elde edilir. Şekil-2.6'de tipik bir üç katmanlı algılayıcı görünmektedir. Genelde, standart bir l-katmanlı ileri-beslemeli YSA bir giriş seviyesi (bir gizli katman) ve bir çıkış katmanından oluşur. İleri-beslemeli özellik nedeniyle aynı katmanda işlem elemanları arası bağlantı olmaz ve geri-besleme bağlantısı bulunmaz. Çok katmanlı algılayıcı, çok katmanlı ileri-beslemeli YSA'nın popüler bir sınıfıdır. Her bir işlem elemanı ister eşik fonksiyonunu, isterse sigmoid fonksiyonunu kullanabilir. Geriye yayılımlı öğrenme algoritmasının geliştirilmesi ile YSA'da çok katmanlı algılayıcının kullanımı yaygınlaşmıştır. $w_{ij}(l)(l-1)$. katmandaki i. işlem elemanından l. katmandaki j. işlem elemanına bağlantıyı göstermektedir. $\{(x^{(1)},d^{(1)}),(x^{(2)},d^{(2)}),.....,(x^{(p)},d^{(p)})\}$ p tane eğitme örüntüsünden oluşan bir küme olsun (giriş çıkış çiftleri). Burada $x^{(i)} \in R^n$ ise n-boyutlu örüntü uzayında giriş vektörü ve $d^{(i)} \in [0,1]^m$ ise n-boyutlu hiperküptür. Sınıflama amacıyla m sınıf sayısı olmaktadır. Yapay Sinir Ağı literatüründe karesel hata fonksiyonu genelde aşağıdaki gibi ifade edilir.

$$E = \frac{1}{2}\sum_{i=1}^{p}\left\|y^{(i)} - d^{(i)}\right\|^2 \tag{2.7}$$

Geri-yayılımlı algoritma (2.7) denklemindeki karesel hata fonksiyonunu minimize etmek için kullanılır ve gradient-descent metoduna dayanmaktadır.



Şekil 2.6 Üç katmanlı algılayıcı

Aşağıda geri-yayılımlı (Back-propagation) algoritmanın adımları verilmiştir (Marilyn ve Illigworth, 1991).

1- Ağırlık değerlerine rasgele küçük sayılar ata.

2- Eğitim örneği kümesinden rasgele bir giriş örüntüsü $x^{(\mu)}$ seç.

3- İşareti YSA girişine ver.

4- Çıkış katmanı $\left(O_i = y_i^l\right)$ deki δ_i^l 'yi hesapla.

$$\delta_i^l = g'(h_i^l)\left[d_i^u - y_i^l\right] \tag{2.8}$$

h_i^l ; l. katmandaki i. işlem elemanına net girişi, g'; δ aktivasyon fonksiyonunun türevidir.

5- l=(l-1),.....,1 için hataları geriye doğru yayarak δ 'ları hesapla.

$$\delta_i^l = g'(h_i^l)\sum_j w_{ij}^{l+1}\delta_j^{l+1} \tag{2.9}$$

6- Ağırlıkları güncelle.

$$\Delta w_{ji}^l = \eta S_i^l y_j^{l-1} \tag{2.10}$$

7- Hata değeri daha önceden belirlenmiş bir eşik değerinden daha düşük değil veya maksimum iterasyon sayısına ulaşılmamış ise 2. adıma git, aksi taktirde dur.

Geri–yayılımlı öğrenme algoritması yukarıdan anlaşılacağı üzere, ileriye doğru ve geriye doğru olmak üzere iki aşama içermektedir. İlk aşamada örnek giriş YSA girişine verilir ve her bir işlem elemanının çıkışı hesaplanır. Bu çıkış istenilen çıkış değeri ile karşılaştırılır ve hata hesaplanır. Geriye doğru olarak adlandırılan ikinci aşamada ise çıkış katmanından, giriş katmanına doğru hatalar hesaplanır. Yeni bir giriş bu iki aşama tamamlandıktan sonra verilebilir. Bu işleme hata belli bir değerden küçük oluncaya kadar devam edilir.

2.2 Mikrodalga Transistorun YSA ile Modellenmesi

Bu tezde aktif mikrodalga elemanının küçük-işaret ve gürültü davranışı çoklu kutuplama/konfigürasyon tipi için YSA yaklaşımıyla modellenmiştir. Burada işaret ve gürültü parametreleri YSA eşdeğeri tarafından hesaplanan eleman bir kara kutuyla modellenmiş ve bu modelleme çoklu kutuplama ve farklı konfigürasyon tipleri için, bu iki

parametre kümesinin ölçüm uzayına uydurulmasına dayandırılmıştır. Burada anlatılacak olan Eşdeğer YSA modeli elemanın optimizasyon sırasında eleman fiziği denklemlerinin çözümünü gerektirmez. Diğer modelleme teknikleri ile karşılaştırıldığında YSA yaklaşımının lineer olmayan elemanlar için çok boyutlu bir model olarak kullanılabilme kapasitesine sahip olduğu görülecektir.

Bir mikrodalga transistorunun kutuplama noktası civarındaki küçük işaret ve gürültüye karşı olan davranışları çalışma bandı boyunca sırasıyla saçılma parametreleri S_{11},S_{12},S_{21},S_{22} ve gürültü parametreleri $F_{\min}$, Γ_{opt}, R_n ile belirlenebilir. S-parametreleri ve gürültü parametrelerinin her ikisi de frekansa, kutuplama noktasına, konfigürasyon tipine ve elemanın fiziksel özelliklerine bağlıdır. S-parametreleri, elemanın işaret güç kazançları ve giriş çıkış kapı uyumsuzluk kayıplarını tayin ederken, gürültü parametreleri de elemanın giriş çıkışı arasında sinyal/gürültü oranı kötüleşmesini tayin eder.

Bu çalışmanın hedefleri aşağıdaki gibi sıralanabilir.

(i) Eşdeğer işlevi görecek, tek gizli katmanlı, ileri beslemeli tipteki yapay sinir ağı devresini kurmak,

(ii) Geriye yayılım algoritması ve lineer olmayan tipteki aktivasyon fonksiyonları kullanarak, yapay sinir ağı devresini herhangi bir tipteki aktif eleman için (transistor gibi) çalışma bandı boyunca, çeşitli kutuplama noktalarında ve konfigürasyon tipleri için işaret-gürültü davranışının her ikisini de kullanarak, yapay sinir ağı devresini eğitmek,

(iii) Yapay sinir ağı devresinin performans ölçüsünü belirlemek,

(iv) Seçilen kutuplama noktası civarında, her hangi bir konfigürasyon tipi elemanın lineer olmayan karakteristiğine fonksiyon yaklaşımı yapan eğitilmiş yapay sinir ağı devresini kullanarak her hangi bir frekans için elemanın işaret-gürültü davranışını belirlemek.

Klasik modelleme bakış açısından küçük-işaret-gürültü eşdeğer devreleri öz (intrinsic) devre ve dış (extrinsic) devre olmak üzere iki kısımda incelenir. Bu fiziksel yaklaşımların sık sık karşılaşılan iki önemli sakıncası vardır: Birincisi eşdeğer devre parametrelerinin seçilen elemanın elektriksel davranışının yegane çözümü olamaması, ikincisi ideal olmayan etkilerin yeterince doğru modellenememesinden kaynaklanır. İkinci sakınca küçük ölçme hatalarına göz yumulması durumunda bile iyi olmayan bir uyuma neden olabilir.

Bilgisayar simülasyonu Tümleşik Devre tasarımında en önemli basamaklardan biridir.

Tümleşik devrelerin analiz ve tasarımının doğruluğu, devre elemanlarının bilgisayar simülasyonunun doğruluğu kadar olduğuna göre, yeterli doğrulukta modelleme Tümleşik Devre Tasarımında gerekli bir faktördür. Alçak frekanslarda göreceli basit olan pasif elemanları karekterize edilmesi bile mikrodalga frekans bölgesinde zor olabilir. Çok sayıda parametre kümesi ve aralarındaki karmaşık ilişkileriyle yüksek dereceli lineer olmayan modellerle karekterize edilebilen yarıiletken elemanlar halinde ise, parametrelerin uygun seçimi çok önemli olacaktır. Gerçekte, yetersiz bir uygulama, simülasyon sonuçlarını önemli ölçüde bozabilir. Elemanların lineer olmamaları nedeniyle, genellikle bu model parametreleri doğrudan ölçümlerle tayin edilemez.

Literatürdeki eleman modelleme, iki temele dayandırılılabilir.

(i) Genellikle küçük-işaret performansı çalışmaları, gürültü performansı çalışmalarından ayrılmıştır. Bu konudaki yayınlar ya sadece küçük-işaret modeli ya da mevcut küçük-işaret eşdeğer devrelerine dayandırılmış ve eleman gürültü özellikleriyle ilişkilendirilmemiş gürültü davranışı tasvirleridir (Torpi, 1997).

(ii) Bir tümleşik mikrodalga elemanı ve paketlenmesini karekterize etmenin standart yaklaşımı her bir bileşeni, elektriksel ölçmelerle uyum içinde olacak şekilde bir eşdeğer devreyle modellemektir. Böyle bir yaklaşımın, paketlemenin (kılıflama), elemanın paketlendiği durumdaki toplam elektriksel özelliklerine katkısını doğru olarak modellemediği, (Dobrowolski, 1991) da gösterilmiştir.

Ayrıca, klasik eşdeğer devre eleman değerlerini bulmakta kullanılan optimizasyon temelli çıkarım teknikleri, yegane çözüm takımı yerine bir çok alternatif çözüm takımları üretirler. Kısımlara ayırarak ölçme yöntemi (Sebastian, 1996) ve otomatik dekompozisyon (düzeltme) tekniğiyle (Minai ve Williams, 1994) bir kısım gelişmelere rağmen, başlangıç değerinden kaynaklanan belirsizlik hala mevcuttur.

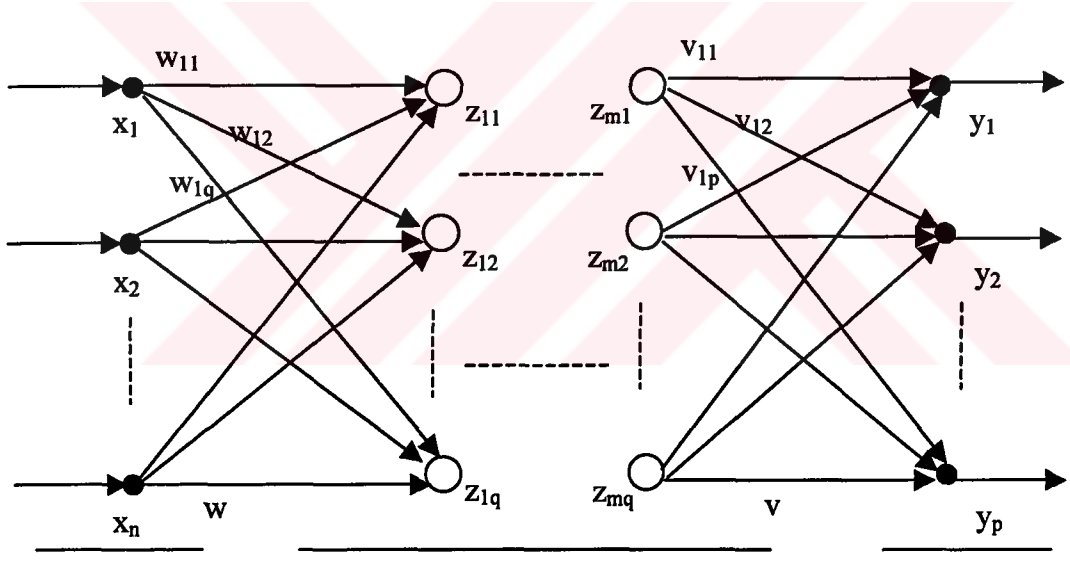
Yakın zamanda yapılan çalışmada (Torpi, 1997), yayınlanan modellere göre ölçme sonuçları ile daha iyi bir uyum sergileyen, mikrodalga transistorunun hem işaret hem de gürültü özellikleri bir YSA modelinde birleştirilmiştir. Fizik-temelli FET modeller gibi hassas modeller kullanma çalışmaları optimizasyon sonuçları ile ilişkilendirilmesi talep edildiğinde göz korkutucu işlemlerle karşı karşıya gelinmektedir. Bu tip modellerin standart optimizasyon ve istatistiksel yaklaşımlarda kullanımı hesaplama açısından çok yoğun bir işlevdir. Bundan dolayı devre simülatörünün her adımında fizik-temelli denklemlerin çözülmesi prosedürü hesaba katılmalıdır. Bu tip var olan optimizasyon metotları off-line tipi

hesaplamalardan esinlenerek geliştirilmiştir. Tasarımcıların devre topolojilerinde değişiklikler yapıldıktan sonra devreyi yeniden optimize etme gereksinimi duymaları durumu gibi interaktif tasarıma uygun değildir. Bu problemlerin üstesinden gelebilmek için, iki tip yaklaşımın çok boyutlu polinomlar yaklaşımı ve "look-up" tablosu yaklaşımının uygulandığı çalışmaların yerine (Torpi, 1997)'de önerilen YSA modeli yaklaşımı eleman modellemeye, devre optimizasyonuna ve istatiksel tasarımda henüz uygulanmaya başlanmamıştır.

Bu tezdeki çalışma bir transistorun YSA modeli kurulmasına yoğunlaşmış ve (Torpi, 1997)'de verilen YSA modeline benzer bir YSA modeli kullanılarak transistorun istenilen frekanstaki gürültü parametreleri belirlenmiştir.

2.2.1 YSA Modelinin Matematiksel Olarak Tanımlanması

YSA biyolojik sinir ağının matematiksel modelidir. Birbirine bağlantılı nöronlar topluluğundan ibarettir. m-gizli katmanlı bir YSA'yı aşağıdaki gibi gösterebiliriz.



Şekil 2.8 m-gizli katmanlı YSA

Model yapısı olarak Şekil 2.8'de görülen YSA'nın tek gizli katmanlı (m=1) olanı kullanılmaktır. Giriş katmanında n düğüm, çıkış katmanında p düğüm ve gizli katmanda q işlem elemanı bulunmaktadır. y çıkış ve x giriş katmanları eleman veya devre parametrelerine ve cevaplarına karşılık düşmektedir. Bizim modelimizde x giriş vektörü [f, V_{CE}, I_C, CT] şeklinde frekans, kutuplama akımı ve gerilimi ve konfigürasyon tipini

içermektedir. ve y çıkış vektörü [|S11|, <S11, |S12|, <S12, |S21|, <S21, |S22|, <S22, F_{min}, $|\Gamma_{opt}|$, $<\Gamma_{opt}$, $R_n/50$] şeklinde transistorun S-parametrelerini ve gürültü parametrelerini içermektedir. Gizli katman q elemanlı bir z vektörüyle gösterilmektedir. Ayrıca,

$$a_k = [a_{k1} \quad a_{k2} \quad \dots \quad a_{kn}]^T \tag{2.11}$$

ve

$$b_k = [b_{k1} \quad b_{k2} \quad \dots \quad b_{kp}]^T \tag{2.12}$$

k. giriş ve çıkış örneğini göstermektedir (k= 1,2,...,N; N; toplam örnek sayısı). Giriş ile gizli katman arasındaki ağırlık değerleri w_{ih}, gizli katmanla çıkış arasında ise v_{hj} ağırlık değerleri bulunmaktadır. Burada i = 1,2,...,n; h = 1,2,...,q ve j = 1,2,...,p olmaktadır. Bu durumda yapay sinir ağının çıkışı,

$$y_j = \sum_{h=1}^{q} z_h v_{hj} \tag{2.13}$$

şeklindedir. Burada,

$$z_h = f(\gamma_h) = \frac{1}{1+e^{-\gamma_h}} \tag{2.14}$$

$$\gamma_h = \left(\sum_{i=1}^{n} a_{ki} w_{ih}\right) + \theta_h = \left(\sum_{i=1}^{n} x_i w_{ih}\right) + \theta_h \tag{2.15}$$

şeklinde tanımlanmaktadırlar (Zaabab A.Hafid, Zhang Qi-Jun, Nakhla Michel, 1995). θ_h; h. gizli işlem elemanı için eşik değeridir. Model parametreleri w_{ih}, v_{hj} ve θ_h değerleridir. Bunların toplam sayısı nxq+pxq+q olmaktadır. Model büyüklüğü veya model parametre sayısı problemin lineer olmama derecesine bağlıdır. Giriş sayısı n artıkça model büyüklüğü exponansiyel olarak artmamaktadır. Dolayısıyla bu model büyük boyutlarda da kullanılabilir. Şekil 2.8' deki YSA için model büyüklüğü;

$$S=q(n+1)p(q+1) \tag{2.16}$$

şeklinde tanımlanmaktadır.

2.2.2 YSA Eğitme Hatası

Yapay sinir ağları belirli sayıda örnek giriş-çıkış verileriyle öğrenmektedirler. Burada a_k ve

b_k k=1,2,...,N (N; toplam örnek sayısı) değerleri eğitme için kullanılacaktır. a_k, n-giriş parametresi, bir devrenin fiziksel veya geometriksel parametreleri olabilir. Bizim durumumuzda giriş parametreleri, [f, V_{CE}, I_C, CT] şeklinde frekans, kutuplama akımı ve gerilimi ve konfigürasyon tipi olmaktadır. b_k, p-çıkış parametresi ise yine, fiziksel devre parametrelerini veya elektriksel devre parametrelerini temsil edebilmektedir. Bizim uygulamamızda ise çıkış parametreleri, [|S11|, <S11, |S12|, <S12, |S21|, <S21, |S22|, <S22, F_{min}, $|\Gamma_{opt}|$, $<\Gamma_{opt}$, $R_n/50$] transistorun S-parametrelerini ve gürültü parametreleri şeklindedir. Yani toplam 12 çıkış parametresi mevcuttur.

Geriye yayılımlı (backpropagation) süresince ağırlıklar ve eşik otomatik olarak ayarlanır ve hata E;

$$E = \sum_{k=1}^{N} E^k = \sum_{k=1}^{N} \left[\frac{1}{2} \sum_{j=1}^{p} (y_j - b_{kj})^2 \right] \tag{2.17}$$

minimum edilir.

On-line ve off-line olmak üzere iki tür eğitim algoritması söz konusu. On-line durumunda yapay sinir ağı parametreleri her bir örnek değerden sonra güncelleniyorlar, off-line durumunda ise tüm örnek değerlerinden sonra güncelleme yapılıyor. Burada on-line durumu kullanılacaktır. Güncelleme denklemleri;

$$v_{hj}^{k+1} = v_{hj}^{k} - \eta \frac{\partial E^k}{\partial v_{hj}} + \alpha (v_{hj}^{k} - v_{hj}^{k-1}) \tag{2.18}$$

$$w_{ih}^{k+1} = w_{ih}^{k} - \eta \frac{\partial E^k}{\partial w_{ih}} + \alpha (w_{ih}^{k} - w_{ih}^{k-1}) \tag{2.19}$$

$$\theta_{h}^{k+1} = \theta_{h}^{k} - \eta \frac{\partial E^k}{\partial \theta_{h}} + \alpha (\theta_{h}^{k} - \theta_{h}^{k-1}) \tag{2.20}$$

bunda η ve α sırasıyla öğrenme oranı (pozitif) ve momenttir. Devreye ilişkin duyarlık,

$$\frac{\partial E^k}{\partial v_{hj}} = \frac{\partial}{\partial v_{hj}} \left[\frac{1}{2} \sum_{j=1}^{p} (y_j - b_{kj})^2 \right] = (y_j - b_{kj}) z_h = \delta_j^{(3)} z_h \tag{2.21}$$

$$\frac{\partial E^k}{\partial w_{ih}} = \sum_{j=1}^{p} \frac{\partial E^k}{\partial y_j} \frac{\partial y_j}{\partial z_h} \frac{\partial z_h}{\partial \gamma_h} \frac{\partial \gamma_h}{\partial w_{ih}}$$

$$= \sum_{j=1}^{p} (y_j - b_{kj}) v_{hj} z_h (1 - z_h) a_{ki}$$

$$= z_h (1 - z_h) \sum_{j=1}^{p} \delta_j^{(3)} v_{hj} a_{ki} = \delta_h^{(2)} a_{ki} \qquad (2.22)$$

ve

$$\frac{\partial E^k}{\partial \theta_h} = \sum_{j=1}^{p} \frac{\partial E^k}{\partial y_j} \frac{\partial y_j}{\partial z_h} \frac{\partial z_h}{\partial \gamma_h} \frac{\partial \gamma_h}{\partial \theta_h}$$

$$= \sum_{j=1}^{p} (y_j - b_{kj}) v_{hj} z_h (1 - z_h)$$

$$= z_h (1 - z_h) \sum_{j=1}^{p} \delta_j^{(3)} v_{hj} = \delta_h^{(2)} \qquad (2.23)$$

şeklindedir. $\delta_h^{(2)}$ ve $\delta_j^{(3)}$ sırasıyla 2. ve 3. katmandaki h. ve j. nöronları ilişkin yerel gradiyentlerdir (Zaabab A.Hafid, Zhang Qi-Jun, Nakhla Michel, 1995).

2.2.3 Eğitme Algoritması

Kullanılan eğitme algoritması geriye-yayılım tekniğine dayanmaktadır. Bu teknik aşağıdaki adımlardan oluşmaktadır (Zaabab A.Hafid, Zhang Qi-Jun, Nakhla Michel, 1995).

1. Adım: Gizli düğüm sayısı q'yu belirle ve w_{ih}, v_{hj} ve θ_h'yı küçük rasgele sayı alarak belirle. Ayrıca η (öğrenme oranı) ve α (momenti) için başlangıç değeri ver.

2. Adım: k=1

3. Adım: (a_k, b_k) belirle ve $x=a_k$ al.

4. Adım: İleri yayılım: (2.13)-(2.15)'den y çıkışını hesapla.

5. Adım: Hatanın geriye yayılımı: (2.17)'dan E^k hatasını hesapla ve (2.21)-(2.23)'den gradyentleri hesapla. (2.18)-(2.20)'dan ağırlık değerlerini belirle.

6. Adım: k= k+1, eğer $k \leq N$ ise 3. Adım'a git.

7. Adım: Toplam hata E'yi hesapla.

8. Adım: Eğer E verilen ε eğitme toleransından daha küçükse dur.

9. Adım: Eğer E, ε'dan daha büyükse, öğrenme oranı ve momenti düşür.

$\eta=\eta.\gamma$ ve $\alpha=\alpha.\gamma$ ve 2.Adım'a git.

10. Adım: Eğer E artıyorsa öğrenme oranı ve momenti arttır.

$\eta=\eta.(1/\gamma)$ ve $\alpha=\alpha.(1/\gamma)$ ve 2. Adıma git.

2.2.4 Eğitme Parametreleri

Kullanılan eğitme parametreleri aşağıda verilmiştir. Ayrıca Çizelge 2.5 bu konuda bir tabloyu içermektedir.

- Gizli katman sayısı; bu problemde bir alınması yeterli
- Gizli düğüm sayısı q; karmaşık problemlerde büyük olmalı, şu anda 12 olması yeterli
- Öğrenme oranı η; adım aralığını ayarlar ve dinamik değişiyor
- Moment α; genelde $0<\alpha<1$ ve yakınsamayı hızlandırır ve parametrik
- Eğitme toleransı ε; çok küçük alınırsa uzun eğitme süresi gerekir, 0.5 alındı
- Öğrenme oranı uydurma γ; öğrenme süresini kısaltır, bu değer şimdilik 1 alınıyor.

Buradaki $q,\eta,\alpha,\varepsilon$ ve γ değerleri genelde probleme bağımlıdırlar ve deneysel sonuçlarla elde edilirler.

2.2.5 Mikrodalga Transistorun Küçük-İşaret ve Gürültü Davranışının Belirlenmesi

Belli bir konfigürasyon tipinde bir kutuplama noktası civarındaki mikrodalga transistorunun küçük-işaret ve gürültü performansı ω-domeninde saçılma **S** ve gürültü **N** parametre vektörleri sıklıkla verilir. Ölçülmüş performans parametre verisi tablo-formunda aşağıdaki gibi verilebilir:

$$\begin{bmatrix} f_1 & : & \mathbf{S}^{(1)} & \mathbf{N}^{(1)} \\ f_2 & : & \mathbf{S}^{(2)} & \mathbf{N}^{(2)} \\ & & & \\ f_N & : & \mathbf{S}^{(N)} & \mathbf{N}^{(N)} \end{bmatrix} \tag{2.24}$$

Burada $\mathbf{S}^{(1)}$, $\mathbf{N}^{(1)}$; ... ;$\mathbf{S}^{(N)}$,$\mathbf{N}^{(N)}$ sırasıyla, saçılma ve gürültü vektörleridir ve f_1, ,...,f_N örnek çalışma frekansları ve $\mathbf{S}^{(N)}$ ve $\mathbf{N}^{(N)}$ performans vektörleri aşağıdaki gibi verilebilir.

$$\left[S^{(N)}\right]^T = \left[|S_{11}|^{(N)} \angle S_{11}^{(N)} \quad |S_{12}|^{(N)} \angle S_{12}^{(N)} \quad |S_{21}|^{(N)} \angle S_{21}^{(N)} \quad |S_{22}|^{(N)} \angle S_{22}^{(N)}\right] \tag{2.25}$$

$$\left[N^{(N)}\right]^T = \left[F_{\min}^{(N)} \quad |\Gamma_{opt}|^{(N)} \angle \Gamma_{\text{opt}}^{(N)} \quad R_N^{(N)}\right]$$

(2.24) ve (2.25)'de verilen veriler YSA modelini eğitmek için kullanılır ve sonra, performans-parametre vektörleri $\mathbf{S}^{(k)}$, $\mathbf{N}^{(k)}$ arzu edilen bir f_k frekansında devrenin çıkışı, f_k frekansı girilerek elde edilebilir.

İki kapılı aktif elemanın bir çalışma frekansında **S** ve **N** vektörlerini belirledikten sonra, (Z_S, Z_L) sonlandırma çifti belirlenecektir.

2.2.6 Eşdeğer YSA Modeli

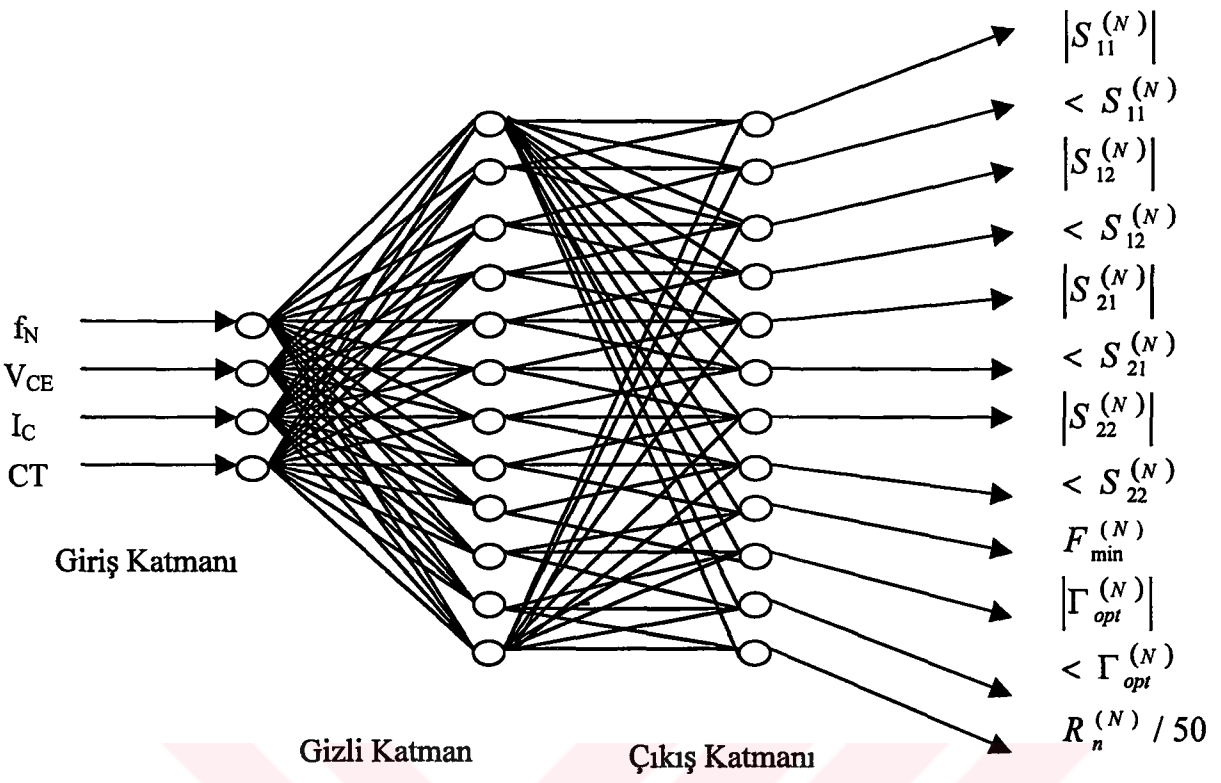
Yukarıda kısaca mikrodalga mühendisliğinde aktif mikrodalga elemanlarının analizinin temeli verildikten, S ve gürültü parametrelerinin rolü anlaşıldıktan sonra bir kara kutu işlevi görecek olan eşdeğer YSA modellerini tanıtmaya başlayalım. Çalışmalarımızın başlangıcında kullandığımız eşdeğer YSA modellerimiz hem gürültü hem de S-Parametrelerini belirli bir frekans bandında belirlemeye yöneliktir ve Şekil 2.7'de gösterilmiştir.

Bu çalışmada sürekli değerli girişe sahip, ileri beslemeli bir yapay sinir ağı olan çok katmanlı algılayıcı (ÇKA) kullanılmıştır. Eğitme algoritması olarak da geriye yayılım algoritması (GYA) kullanılmıştır. Geriye yayılım eğitim algoritması çok katmanlı algılayıcının aktif çıkışıyla arzu edilen çıkış (Hedef uzayla belirlenen) arasındaki karesel ortalama hatanın minimize edilmesine dayanan bir gradyant (Türev temelli arama) algoritmasıdır (Torpi, 1997). Dolayısıyla bu teknik sürekli türevi alınabilir lineer olmayan transfer fonksiyonuna ihtiyaç gösterir. Öğrenme işlemi GYA ile ağırlıklar ayarlanarak gerçekleştirilir.

Bu çalışma için geliştirilen YSA için giriş katmanı, bir gizli katman ve çıkış katmanı olmak üzere üç katmanlı algılayıcı olarak adlandırılan yapının yeterli olduğu görülmüş ve ayrıca ağırlıkların başlangıç değerleri seçiminin, sonucu etkilemediği gözlenmiştir, dolayısıyla rasgele alınmışlardır (Torpi, 1997).

Bu aşamada geliştirdiğimiz eşdeğer YSA modeli Şekil 2.7'de verilmiştir. Her bir modelde YSA olarak Çok Katmanlı Algılayıcı (MLP:Multi Layer Perceptron) kullanılmıştır.

Zaman-hata optimum ilişkisini sağlamak için her bir yapıda çıkış katmanı düğüm sayısı gizli katman düğüm sayısına eşit alınmıştır (Torpi, 1997).



Şekil 2.7 Çok boyutlu eşdeğer işaret-gürültü parametreleri YSA modeli

Bu çok boyutlu modellemeyi temin eden YSA'nın eğitilmesi üretici firmanın verdiği muhtelif kutuplama ve konfigürasyon tipindeki performans verisini kullanılarak hedef uzayının teşkil edilmesi ile sağlanmıştır. Eğitme algoritması olarak geriye yayınım algoritması kullanılmıştır. Hedef uzayını teşkil ederken kullanılacak verinin adedini modelden optimum performansı elde edilecek şekilde belirlenmiştir. Eğitme işlemi tamamlandıktan sonra, istenilen f_k frekansında, V_{CE}, I_C (veya V_{DS}, I_{DS}) ile belirli kutuplama noktasında, her hangi bir konfigürasyon tipiyle belirli giriş vektörü için aktif mikrodalga elemanının çıkışı olarak işaret ve gürültü parametre vektörleri $\mathbf{S}^{(k)}$ ve $\mathbf{N}^{(k)}$ kestirilebilir. Eşdeğer YSA'da konfigürasyon tipi belirli sayılarla temsil edilmiştir. Eğer $\mathbf{S}^{(k)}$, $\mathbf{N}^{(k)}$ hedef uzayı içinde değilse YSA'nın kestirim özelliği ile bu değerle tabii olarak hesaplanabilecektir. Bu YSA'nın eğitilmemiş girişler içinde güzel çıkış verme özelliğinin bir yansımasıdır. Bizim uygulamamızda YSA işaret-gürültü eşdeğeri sadece verilen uzayın kapsayan frekans ve kutuplama kademesinde değil (interpolasyon) bu frekans ve kutuplama kademesinin dışında da kestirim (extrapolasyon) yapabilme yeteneği vardır. Sekil 2.7`deki YSA`nin program akış diyagramı 5. Bölümde ve programın kendisi Ek-1`de

verilmiştir. YSA`nin giriş/çıkış ve diğer gerekli parametreleri aşağıdaki çizelgelerde özetlenmiştir.

Çizelge 2.3 YSA giriş parametreleri

Parametre	**Notasyon**	**Birim**	**Yorum**
Frekans	f	Hz	Transistorun çalışma bandı
Kutuplama Gerilimi	V_{CE}	V	Veya V_{DS}
Kutuplama Akımı	I_C	A	Veya I_D
Konfigürasyon Tipi	CT	-	CE→ CT=0.1, CC→ CT=0.9 CE; Common Emiter, CC; Common Collector

Çizelge 2.4 YSA çıkış parametreleri

Parametre	**Notasyon**	**Birim**	**Yorum**
S-parametreleri Genliği	$\|S_{ij}\|$	-	i,j=1,2
S-parametreleri Açısı	$<S_{ij}$	deg.	i,j=1,2
Minimum Gürültü	F_{min}	dB	
Optimum Kaynak Yansıma Katsayısı Genliği	$\|\Gamma_{opt}\|$	-	
Optimum Kaynak Yansıma Katsayısı Açısı	$<\Gamma_{opt}$	deg.	
Eşdeğer Gürültü Direnci	R_n	Ω	

Çizelge 2.5 YSA eğitme parametreleri

Parametre	**Notasyon**	**Değer**	**Yorum**
Giriş Düğüm Sayısı	n	4	Parametrik
Çıkış Düğüm Sayısı	P	12	Parametrik
Gizli Düğüm Sayısı	q	12	Parametrik
Gizli Katman Sayısı	Q	1	Parametrik
Örnek Sayısı	N	-	İsteğe bağlı
Momentum	α	0.1	Parametrik
Öğrenme Oranı	η	0.5	Dinamik Değişiyor

Yukarıda verilen veriler kullanılarak geliştirilen ve Ek-1`de verilen programın akış diyagramı 5. Bölümde verilmiştir. Tanıtımı yapılan genel YSA modeline yönelik detaylı performans analiz sonuçları (Torpi, 1997)'da verilmiştir.

3. MİKRODALGA TRANSİSTORUN PERFORMANS ANALİZİ ve EMPEDANS VERİLERİ

3.1 Transistorun Performans Ölçü Fonksiyonları

Bu bölümde gerekli temel formülasyonları verdikten sonra ileriki bölümlerde analiz işlemlerine geçilecektir.

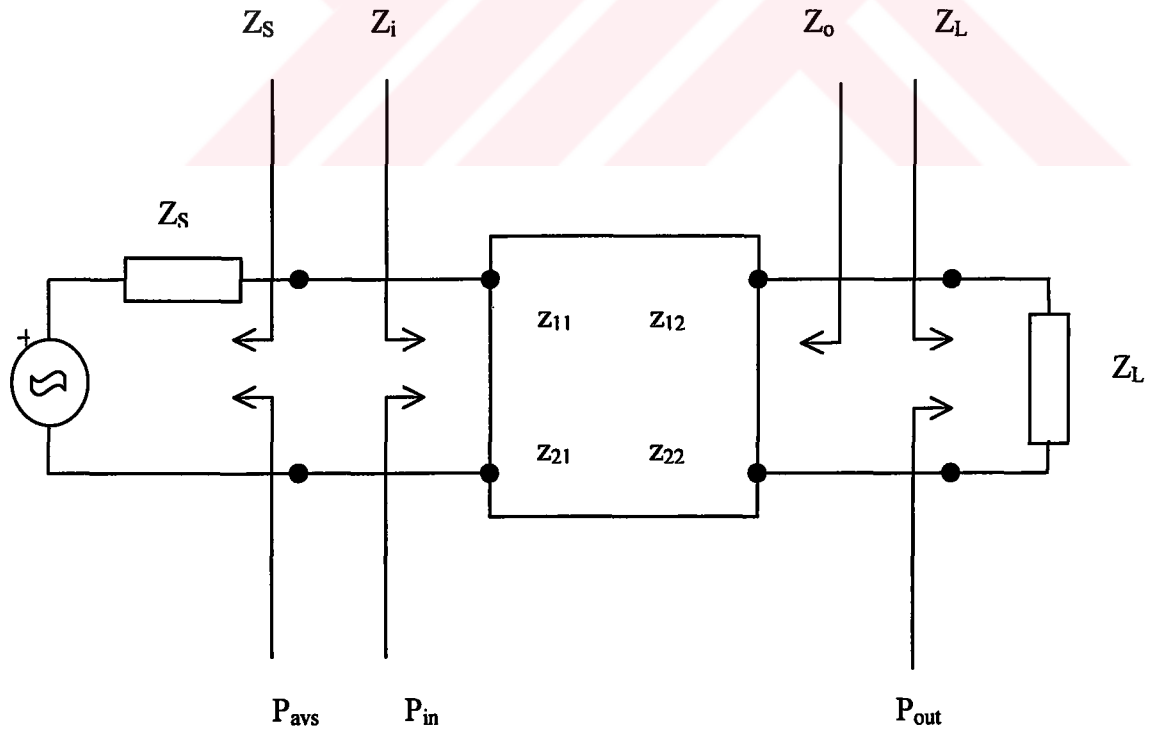
3.1.1 Gürültü Faktörü

Şekil 3.1'de verilen bir $Z_S = R_S + jX_S$ kaynak empedansına ilişkin gürültü faktörü:

$$F = F_{\min} + \frac{R_n \left| Z_S - Z_{opt} \right|^2}{\left| Z_{opt} \right|^2 R_S} \tag{3.1}$$

şeklinde ifade elde edilebilir (Güneş Filiz, Güneş M., Fidan M., 1994).

Burada R_n; eşdeğer gürültü direnci, $F_{\min}$; minimum gürültü faktörü ve $Z_{opt} = R_{opt} + jX_{opt}$; optimum kaynak empedansı olmakta ve bu değerler transistor parametreleriyle birlikte verilmektedirler.



Şekil 3.1 İki Kapılı Devre ve Kapı Empedansları

3.1.2 Giriş VSWR

$Z_S = R_S + jX_S$ kaynak empedansı ve $Z_L = R_L + jX_L$ yük empedansına sahip Şekil 3.1 deki gibi bir iki kapılının giriş VSWR (Vi):

$$V_i = \frac{1+|\rho_i|^2}{1-|\rho_i|^2} \tag{3.2}$$

şeklinde ifade edilir. Burada ρ_i:

$$|\rho_i| = \left|\frac{Z_S - Z_i^*}{Z_S + Z_i}\right| \tag{3.3}$$

şeklinde ifade edilen giriş yansıma katsayısı olmaktadır. Buradaki giriş empedansı z_{ij} iki kapılının küçük işaret açık devre z-parametreleri olmak üzere $(Z_i = R_i + jX_i)$:

$$Z_i = z_{11} - \frac{z_{12}z_{21}}{z_{22} + Z_L} \tag{3.4}$$

şeklinde ifade edilmektedir (Güneş Filiz, Güneş M., Fidan M., 1994).

3.1.3 Transduser Güç Kazancı

Şekil 3.1 deki iki kapılının Z_S, Z_L ve z-parametrelerine bağlı transduser güç kazancı:

$$G_T = \frac{4R_S R_L |z_{21}|^2}{\left|(z_{11} + Z_S)(z_{22} + Z_L) - z_{12}z_{21}\right|^2} \tag{3.5}$$

şeklinde ifade edilir (Güneş Filiz, Güneş M., Fidan M., 1994).

3.2 Optimizasyon Problemi

Bu bölümün konusu olan ve GÜNEŞ metoduna dayanan (Güneş Filiz, Güneş M., Fidan M., 1994) optimizasyon problemi, $F_{req} - F(R_S, X_S) = 0$ ve $V_{ireq} - V_i(R_S, X_S, R_L, X_L) = 0$ olmak üzere ($G_{T\max}$, maksimum kazanç ve G_{Treq} istenilen kazanç), $G_{T\max} - G_T(R_S, X_S, R_L, X_L) = 0$ ve ayrıca $G_{Treq} - G_T(R_S, X_S, R_L, X_L) = 0$ olacak şekilde iki ayrı durum için kararlı bölgede kalmak koşuluyla mümkün tüm $Z_S = R_S + jX_S$ ve $Z_L = R_L + jX_L$ kompleks sonlandırmalarını geometriksel ve matematiksel olarak belirlemek

şeklinde ifade edilebilir. Bu durumda bu işlemler sonucunda; $\{F_{req}, V_{ireq}, G_{T\max}[f, V_{CE}, I_C]\} \Leftrightarrow Z_{L\max}[f, V_{CE}, I_C]$, $Z_{S\max}[f, V_{CE}, I_C]$ ve $\{F_{req}, V_{ireq}, G_{T\min} \le G_{Treq} \le G_{T\max}[f, V_{CE}, I_C]\} \Leftrightarrow Z_{Lreq}[f, V_{CE}, I_C]$, $Z_{Sreq}[f, V_{CE}, I_C]$ fonksiyonlarına ulaşılmış olacaktır. Burada f; frekansı ve V_{CE} ile I_C (veya V_{DS} ile I_{DS}) kutuplama noktasını dolayısıyla transistorun gürültü ve S-parametrelerini ifade etmektedir.

Bu tür yüksek dereceli polinomların içerildiği optimizasyon problemlerin çözümüne yönelik bir takım yöntemler literatürde sıkça yer almaktadır. Ancak tanımladığımız problemin hem reel hem de imajiner kısma sahip olması ve ayrıca çok hesaplama zamanına ihtiyaç duymaları önerilen geometriksel yöntemi daha avantajlı duruma getirmektedir. Bunun yanında bu tezde tüm çözümler hem analitik hem de geometriksel olarak ifade edilmişlerdir.

Kullanılan yöntem aşağıdaki adımlardan oluşmaktadır:

a) Verilen bir transistor ilişkin F=sabit, V_i=sabit ve G_T=sabit dairelerine yönelik Zs-düzleminde analiz yapılır. Bu adımda gürültü ve VSWR dairelerinin Zs-düzlemindeki konumları belirlenmiş olur. Bu dairelerin durumları $G_{T\max}, G_{Treq}, Z_S$ ile Z_L değerlerini belirlenmesine yardımı olacaktır.

b) Zi-düzleminde yapılacak analizlerde kararlılık bölgeleri belirlenir. Ayrıca burada kararlı bölgede belirlenecek Z_i değeri bir Z_L'e ve Zs-düzleminde F ve V_i dairelerinin durumuna göre bir veya iki Z_S değerine karşılık düşürülecektir.

c) Maksimum ve istenilen kazanç için tüm mümkün Z_S ve Z_L değerleri kararlı bölgede kalacak şekilde belirlenir.

d) Bu işlemler tüm çalışma bandı frekanslarında yapılarak analiz tanımlanmış olur .

3.3 Zs-düzleminde Sabit Gürültü, Giriş VSWR ve Kazanç Daireleri

Zs-düzleminde merkezi $Z_c = R_c + jX_c$ ve yarıçapı r_c olan bir daire denklemi:

$$|Z_S - Z_c| = r_c \tag{3.6}$$

veya

$$|Z_S|^2 - 2R_cR_S - 2X_cX_S + |Z_c|^2 - r_c^2 = 0 \tag{3.7}$$

şeklinde ifade edilebilir (Güneş Macit, 1980).

3.3.1 Sabit Gürültü Daireleri

(3.1) eşitliğini de kullanarak Zs-düzlemindeki sabit gürültü dairesi aşağıdaki eşitlikteki gibi ifade edilebilir. Bunun geometriksel ifadesi Şekil 3.2'de verilmiştir (Güneş Macit, 1980).

$$\left|Z_S - Z_{opt}\right|^2 = 2NR_S \tag{3.8}$$

$$\left|Z_S\right|^2 - 2(R_{opt} + N)R_S - 2X_{opt}X_S + \left|Z_{opt}\right|^2 = 0 \tag{3.9}$$

Burada merkez $Z_{cn} = R_{cn} + jX_{cn}$ ve yarıçapı r_n (3.6)'nın yardımıyla şu şekilde bulunabilir:

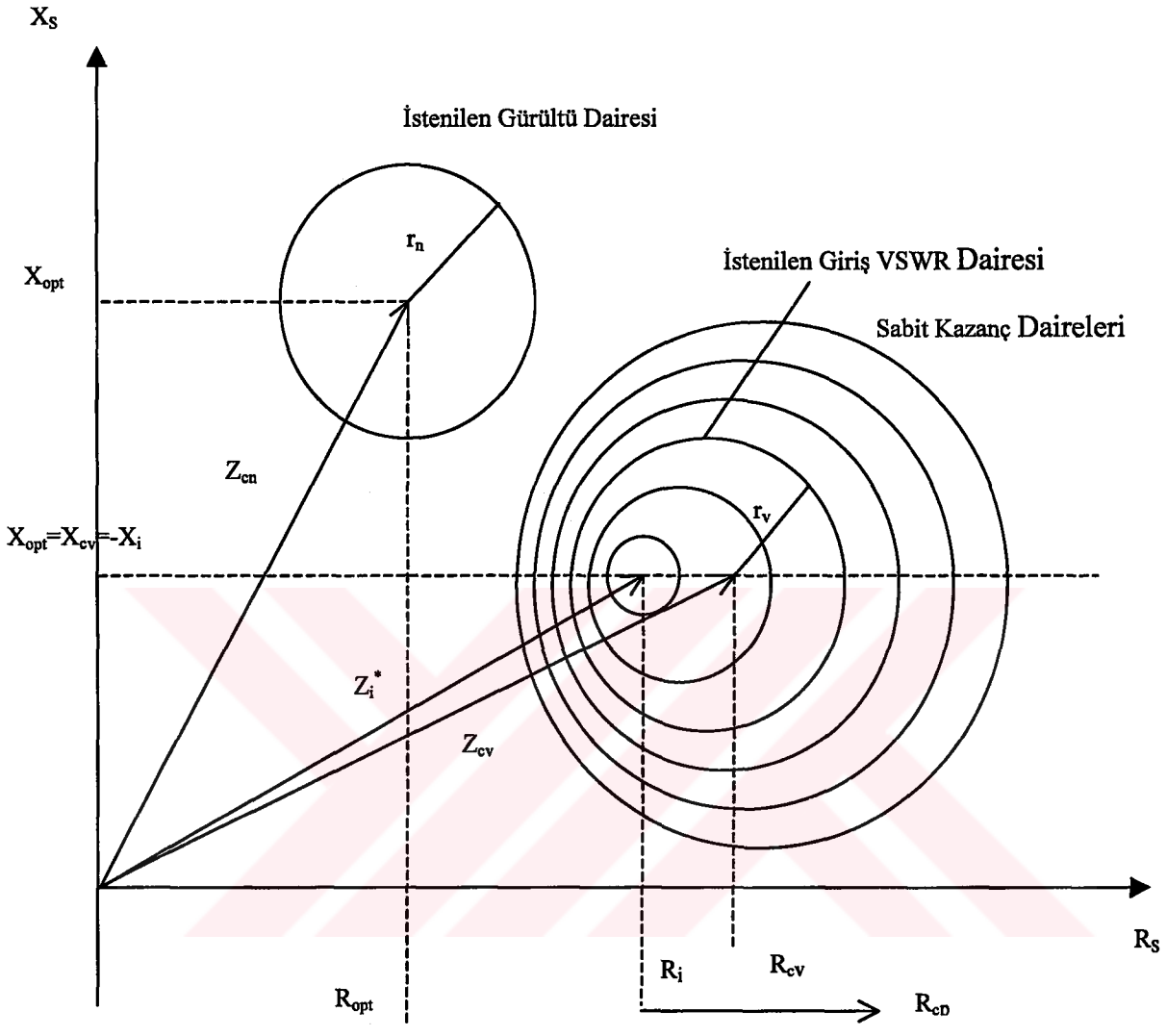
$$R_{cn} = R_{opt} + N \tag{3.10}$$

$$X_{cn} = X_{opt} \tag{3.11}$$

$$r_n = \sqrt{N(N + 2R_{opt})} \tag{3.12}$$

$$N = \frac{F_{req} - F_{\min}}{2R_N}\left|Z_{opt}\right|^2 \tag{3.13}$$

$F_{\min}, R_N, R_{opt}, X_{opt}$ Transistorun S-Parametreleri dosyasından gelmektedir. F_{req} istenilen gürültüdür.



Şekil 3.2 Sabit Gürültü Faktörü, Giriş VSWR ve Kazanç Daireleri

3.3.2 Sabit Giriş VSWR Daireleri

Sabit giriş VSWR (ρ_i) dairelerine yönelik işlemleri basitleştirmek için giriş VSWR (V_i) yerine giriş yansıma katsayısı, (3.3) formülasyonundan elde edilerek kullanılmaktadır.

(3.3) eşitliği kullanılarak sabit giriş VSWR daireleri Zs-düzleminde aşağıdaki şekilde elde edilir:

$$|Z_S|^2 - 2R_i \frac{1+|\rho_i|^2}{1-|\rho_i|^2} R_S + 2X_i X_S + |Z_i|^2 = 0 \tag{3.14}$$

Burada (3.7) eşitliği kullanılarak merkez $Z_{cv} = R_{cv} + jX_{cv}$ ve yarıçap r_v aşağıdaki gibi bulunabilir (Şekil 3.2) (Güneş Macit, 1980);

$$R_{cv} = \frac{1+|\rho_i|^2}{1-|\rho_i|^2} R_i \tag{3.15}$$

$$X_v = -X \tag{3.16}$$

$$r_v = 2\frac{|\rho_i|}{1-|\rho_i|^2} R_i \tag{3.17}$$

Burada R_i ve X_i giriş empedansının reel ve imajiner kısımlarıdır $(Z_i = R_i + jX_i)$. (3.15), (3.16), (3.17) eşitliklerinden açıkça görüldüğü gibi $|\rho_i| = 0$ olduğunda $Z_{cv} = Z_i, r_v = 0$ ve giriş VSWR = 1 (giriş kapısında kompleks eşlenik uydurma) olur.

3.3.3 Sabit Transduser Kazanç Daireleri

(3.5) formülünü Z_S`e göre düzenlersek:

$$|Z_S|^2 - 2(\frac{C}{G_T} - R_i)R_S + 2X_i X_S + |Z_i|^2 = 0 \tag{3.18}$$

ifadesi elde edilir. Burada $C = (2R_L|z_{21}|^2)/|z_{22} + Z_L|^2$ şeklinde bir sabittir.

Zs-düzleminde sabit transduser güç kazanç dairelerinin merkezi $Z_{cp} = R_{cp} + jX_{cp}$ ve yarıçapı r_p aşağıdaki şekilde ifade edilebilir (Şekil 3.2) (Güneş Macit, 1980).

$$R_{cp} = \frac{C}{G_T} - R_i \tag{3.19}$$

$$X_{cp} = -X_i \tag{3.20}$$

$$r_p = \sqrt{\frac{C}{G_T}\left(\frac{C}{G_T} - 2R_i\right)} \tag{3.21}$$

Maksimum kazancın, yarıçapı sıfıra eşit olan limit dairesi için elde edilebileceğini biliyoruz. Buradan yola çıkarak $r_p = 0$ ve C>0 ise $(C/G_T) = 2R_i$` yi (3.19) de yerine koyarsak $Z_{cp} = Z_S = Z_i^*$ bulunur. Çıkan sonuçtan da görüldüğü gibi maksimum kazanç kaynak empedansı, iki kapılının giriş empedansının kompleks eşleniğine uydurulduğunda elde edilir. (3.16) ve (3.20)'den görüldüğü gibi giriş VSWR transduser kazanç dairelerinin merkezleri $(-X_i)$ imajiner eksenin üzerinde olduğu hemen anlaşılır. Ayrıca geometriksel durumda da (Şekil 3.2) bunu göstermektedir. $R_{cn} = R_{cv}$ olduğunda $r_n = r_c$ olduğu kolayca gösterilebilir. (3.15) ve (3.19) eşitlenerek;

$$\frac{C}{G_T} = \frac{2R_i}{1-|\rho_i|^2} \tag{3.22}$$

elde edilir ve (3.22), (3.21)'de yerine konularak;

$$r_p = 2\frac{|\rho_i|}{1-|\rho_i|^2} R_i \tag{3.23}$$

bulunur. r_p (3.17)'de bulunan r_v'ye eşit çıkmıştır. C ifadesi (3.22)'de yerine konularak Zs-düzleminde verilen giriş VSWR'ı karşılayan maksimum kazanç bulunur:

$$G_T = \frac{|z_{21}|^2}{|z_{22} + Z_L|^2}\frac{R_L}{R_i}(1-|\rho_i|^2) \tag{3.24}$$

(3.24) ifadesi Şekil 3.1'den de belirlenebilir.

$$G_T = \frac{P_{out}}{P_{avs}} = \frac{P_{in}}{P_{avs}} \cdot \frac{P_{out}}{P_{in}} \tag{3.25}$$

şeklindedir. Burada çalışma güç kazancı G_{out};

$$G_{out} = \frac{P_{out}}{P_{in}} \tag{3.26}$$

şeklinde tanımlanır. Ayrıca;

$$G_{out} = \frac{|z_{21}|^2 R_L}{|z_{22} + Z_L|^2 R_i} \tag{3.27}$$

ve

$$\frac{P_{in}}{P_{avs}} = \frac{4R_S R_i}{|Z_S + Z_i|^2} = \frac{|Z_S + Z_i|^2 - |Z_S - Z_i^*|^2}{|Z_S + Z_i|^2} = 1 - |\rho_i|^2 \tag{3.28}$$

olduğundan (3.25) ifadesinden (3.24) ifadesine ulaşılabilir (Güneş Macit, 1980).

Bu tartışmaların sonucu olarak yalnız verilen gürültü ve giriş VSWR daireleri Zs-düzleminde göz önüne alınabileceği ortaya çıkar. Çünkü bu iki dairenin ortak iki noktası yalnız gürültü ve giriş VSWR gereksinimlerini karşılamakla kalmaz aynı zamanda Zs-düzleminde transduser güç kazancını da maksimum yapar. İstenilen bir G_{Treq}`de bu dairelerin kesişimleri sonucu elde edilebilecektir.

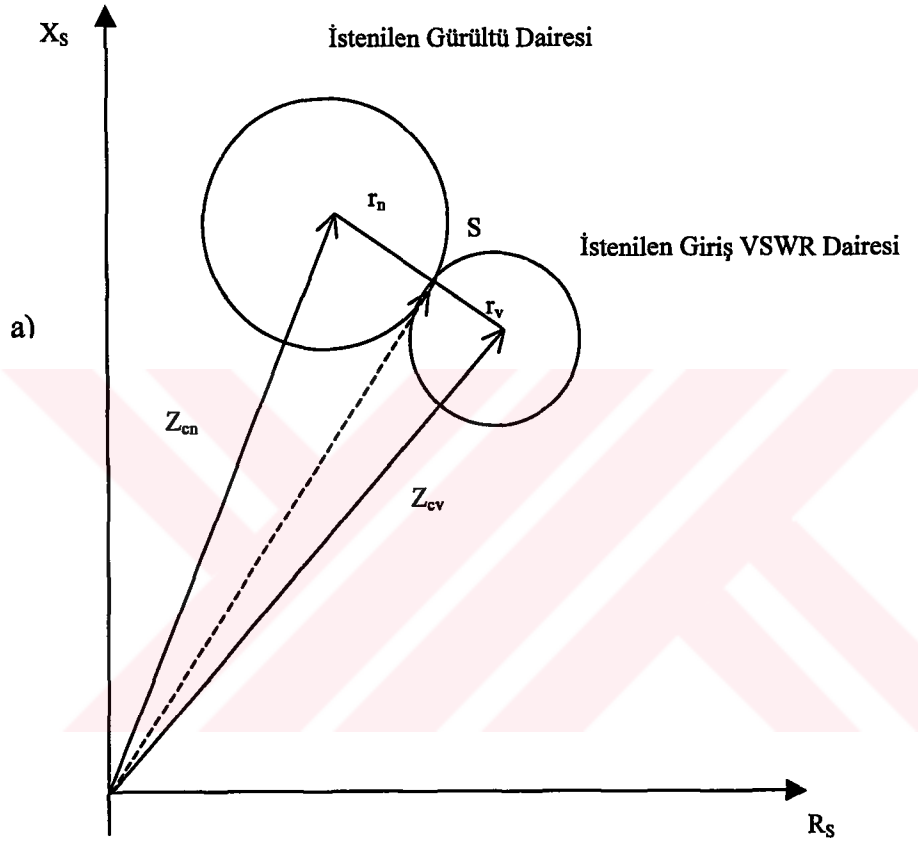
Çalışmalar neticesinde verilen gürültü dairesinin Zs-düzleminde sabit olduğunu ve verilen giriş VSWR dairesinin giriş empedansı (Z_i) yoluyla yük empedansına bağlı olarak yer değiştirildiğini gördük. Bu nedenle aşağıdaki durumlar mümkündür (Güneş Macit, 1980):

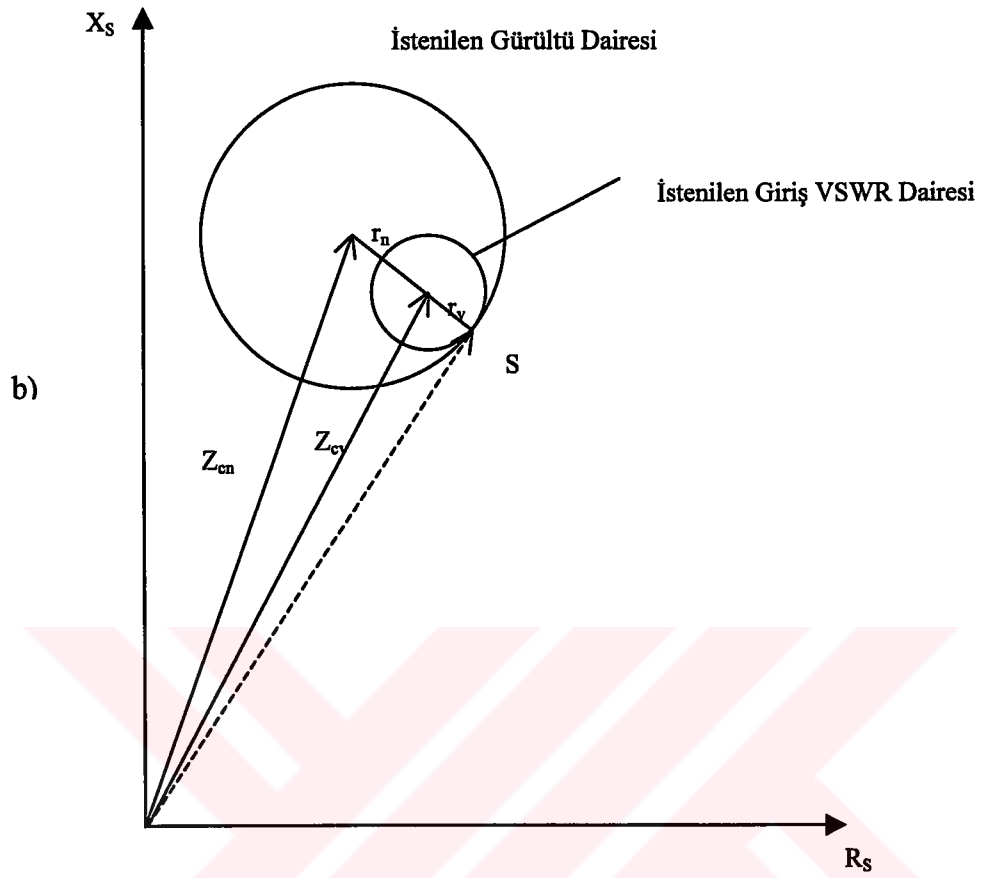
a)-Bu daireler birbirlerine dokunmayabilirler

b)-Birbirlerine iç veya dış teğet olabilirler (Şekil 3.3)

c)-Birbirlerini kesebilirler

İleriki bölümde bu olasılıkların her biri ayrı ayrı incelenecektir.


X_S
İstenilen Gürültü Dairesi
r_n
S
İstenilen Giriş VSWR Dairesi
r_v
a)
Z_{cn}
Z_{cv}
R_S



Şekil 3.3 Zs-düzleminde İstenilen Gürültü Faktörü ile Giriş VSWR'ın Teğet Durumları

a) İç Teğet

b) Dış Teğet

3.3.4 Zs-düzleminde Sabit Gürültü ve Giriş VSWR Dairelerinin Konumları

Zs-düzleminde birbirine değmeyen ve kesişen pozisyonlar arasında geçiş safhaları olan gürültü ve giriş VSWR dairelerinin iki teğet pozisyonu, Şekil 3.3'de dış ve iç teğet pozisyonları olarak gösterilmiştir. Zs-düzleminde her iki pozisyonu kapsayacak teğet durumu için genel bir eşitlik aşağıdaki şekilde verilebilir:

$$\left|Z_{cn} - Z_{cv}\right|^2 = (r_n \pm r_v)^2 \tag{3.29}$$

(3.10)-(3.12) ve (3.15)-(3.17)'deki Z_{cn}, r_n, Z_{cv}, r_v, ifadeleri (3.29)'da yerine konulursa Zs-düzlemindeki teğet gürültü ve giriş VSWR daire çifti, Zi-düzleminde aşağıdaki çıkarılan

eşitlikteki gibi ifade edilebilir:

$$|Z_i|^2 - 2\left[(R_{opt}+N)\frac{1+|\rho_i|^2}{1-|\rho_i|^2} \pm 2\sqrt{N(N+R_{opt})}\frac{|\rho_i|}{1-|\rho_i|^2}\right]R_i + 2X_{opt}X_i + |Z_{opt}|^2 = 0 \qquad (3.30)$$

Yukarıdaki ifade Zs-düzlemine iki teğet durumunun olması nedeniyle iki farklı daireyi ifade etmektedir (Güneş Filiz, Güneş M., Fidan M., 1994).

3.3.4.1 Dış Teğet Pozisyonu

(3.30)'deki ifade de işaret pozitif alındığı durumlardaki T1 dairesinin merkezi ($Z_{ct1} = R_{ct1} + jX_{ct1}$) ve yarıçapı ($r_{t1}$) için aşağıdaki ifadeleri elde edebiliriz (Güneş Filiz, Güneş M., Fidan M., 1994):

$$R_{ct1} = R_{cn}U + r_nV \qquad (3.31)$$

$$X_{ct1} = -X_{opt} \qquad (3.32)$$

$$r_{t1} = \sqrt{|Z_{ct1}|^2 - |Z_{opt}|^2} \qquad (3.33)$$

$$U = \frac{1+|\rho_i|^2}{1-|\rho_i|^2} \qquad (3.34)$$

$$V = \frac{2|\rho_i|}{1-|\rho_i|^2} \qquad (3.35)$$

R_{cn} ve r_n daha önce (3.10) ve (3.12)'de verilmişti.

3.3.4.2 İç Teğet Pozisyonu

(3.30)'deki ifadede işaret negatif alındığı durumda T2 dairesinin merkezi ($Z_{ct2} = R_{ct2} + jX_{ct2}$) ve yarıçapı ($r_{t2}$) için aşağıdaki ifadeleri elde edebiliriz (Güneş Filiz, Güneş M., Fidan M., 1994):

$$R_{ct2} = R_{cn}U - r_nV \qquad (3.36)$$

$$X_{ct2} = -X_{opt} \qquad (3.37)$$

$$r_{t2} = \sqrt{|Z_{ct2}|^2 - |Z_{opt}|^2} \tag{3.38}$$

T1 ve T2 dairelerinin merkezleri aynı imajiner eksen üzerindedir ve T2 dairesi her zaman T1 dairesinin içinde ve T1'e değmeksizin bulunduğu tüm durumlarda aşağıdaki eşitsizliğin var olduğu gösterilerek ispat edilebilir:

$$r_{t1} > R_{ct1} - R_{ct2} + r_{t2} \tag{3.39}$$

eşitlik sadece $|\rho_i| = 0$ (yani VSWR =1) olduğunda sağlanabilir. (3.31) eşitliğindeki V'nin çözümü sıfır olur ve R_{ct1} ve r_{t1} sırasıyla R_{ct2} ve r_{t2}'ye eşit olur. T1 ve T2 dairesi çakışık çıkar. Daireler ayni daire olurlar. Bunun anlamı Zi-düzleminde sadece bir daire vardır ve giriş VSWR dairesi Zs-düzleminde bir daire değil ama bir noktadır.

Şekil 3.4'de gösterilen T1 ve T2 daireleriyle sınırlanan giriş empedans düzleminde 5 farklı bölge, Zs-düzleminde gürültü dairesiyle verilen giriş VSWR dairesinin aşağıda açıklanan farklı etkileşimlerine neden olur (Şekil 3.5):

1.Bölge: T1 dairesinin dışında seçilen herhangi bir Z_i (giriş empedansı) için Zs-düzleminde istenen giriş VSWR dairesiyle gürültü dairesi birbirine değmeme pozisyonundadır (Şekil 3.5a).

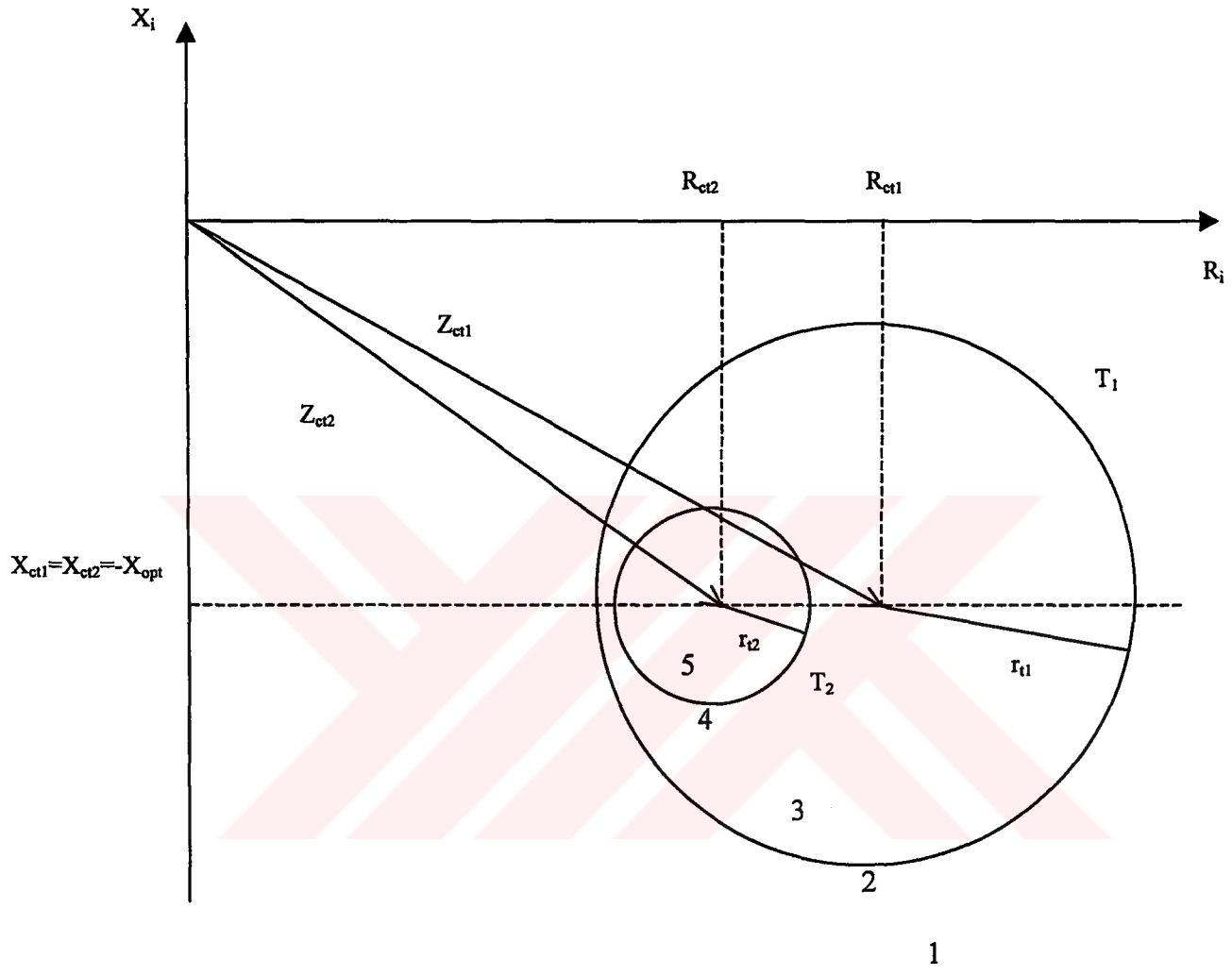
2.Bölge: T1 dairesi üzerinde alınan bir Z_i empedansı sonucu verilen giriş VSWR ve gürültü daireleri Zs-düzleminde dış teğet pozisyonuna karşı düşer (Şekil 3.5b).

3.Bölge: T1 ve T2 dairelerinin arasındaki bölgede seçilen herhangi bir Zi-empedansı sonucunda Zs-düzleminde verilen giriş VSWR dairesiyle gürültü dairesi birbirlerini iki ayrı noktada keserler (Şekil 3.5c).

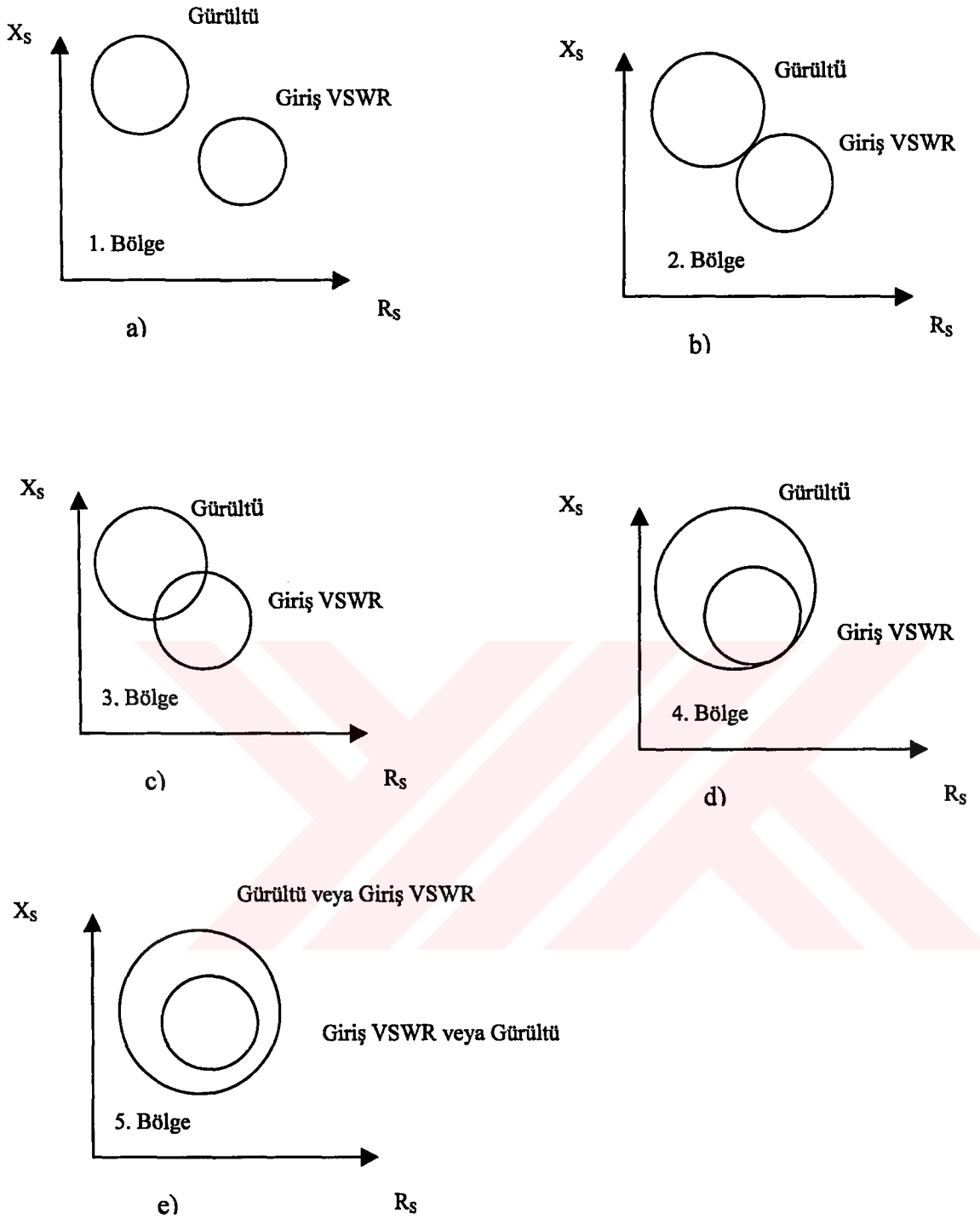
4.Bölge: Z_i'nin T2 dairesinin üzerinde seçilmesi durumunda verilen giriş VSWR ve gürültü daireleri, Zs-düzleminde iç teğet pozisyonundadırlar (Şekil 3.5d).

5.Bölge: Z_i'nin T2 dairesinin içinde seçilmesi durumunda gürültü ve gerekli giriş VSWR daireleri Zs-düzleminde birbirine değmeden iç içe bulunurlar (Şekil 3.5e).

Şekil 3.5a ve Şekil 3.5e'den görüldüğü gibi Zi-düzlemindeki 1.bölge ve 5.bölgelerde, Zs-düzlemindeki verilen giriş VSWR ve gürültü dairelerinin hiçbir ortak noktası yoktur. Bu yüzden bu durumlar verilen giriş VSWR ve verilen gürültü için istenilen ve maksimum kazanç çözümünü vermezler (Güneş Macit, 1980).



Şekil 3.4 Zs-düzleminde Giriş VSWR ve Gürültü Dairelerinin Durumlarını belirleyen Zi-düzlemindeki Beş Farklı Bölge



Şekil 3.5 Zi-düzlemindeki Beş Bölgenin Belirlediği Zs-düzlemindeki Giriş VSWR ve Gürültü Dairelerinin Durumları

a) Zs-düzleminde 1.bölge olması durumu

b) Zs-düzleminde 2.bölge olması durumu

c) Zs-düzleminde 3.bölge olması durumu

d) Zs-düzleminde 4.bölge olması durumu

e) Zs-düzleminde 5.bölge olması durumu

3.4 Koşulsuz Kararlı Mikrodalga Transistoru

3.4.1 Verilen Giriş VSWR İçin Zi-Düzlemi Sabit Kazanç Daireleri

Koşullu optimizasyon probleminin geometrik ve analitik çözümlerini bulabilmek amacıyla verilen giriş VSWR'ı gerçekleyen sabit kazanç daireleri, Zi-düzleminde oluşturulacak ve bu kazanç dairelerinin çözüm bölgeleri içindeki değerleri araştırılıp bütün koşulları sağlayanlar çözüm kümesini oluşturacaktır.

(3.24) ifadesinde yük empedansı (3.4)'den yararlanarak giriş empedansı şeklinde ifade edilirse, Zi-düzleminde bir daire denklemi halinde aşağıdaki gibi yazabilir:

$$|Z_i|^2 - \frac{1}{r_{22}}(2r_{11}r_{22} - r - \frac{G_T|z_{12}|^2}{1-|\rho_i|^2})R_i - \frac{1}{r_{22}}(2x_{11}r_{22} - x)x_i - \frac{1}{r_{22}}(rr_{11} + xx_{11}) + |z_{11}|^2 = 0 \qquad (3.40)$$

Burada;

$$r_{ij} = \mathrm{Re}\{z_{ij}\}, \qquad x_{ij} = \mathrm{Im}\{z_{ij}\}, \qquad z = z_{12}z_{21} = r + jx \qquad (3.41)$$

Bu daire ailesinin merkezi ve yarıçapı sırasıyla $Z_{cg} = R_{cg} + jX_{cg}$ ve r_g olsun.

$$R_{cg} = \frac{1}{2r_{22}}(Q - P) \qquad (3.42)$$

$$X_{cg} = \frac{1}{2r_{22}}(2x_{11}r_{22} - x) \qquad (3.43)$$

$$r_g = \frac{1}{2r_{22}}\sqrt{P^2 - 2QP + |z|^2} \qquad (3.44)$$

Burada;

$$P = \frac{|z_{12}|^2}{1-|\rho_i|^2}G_T \qquad (3.45)$$

$$Q = 2r_{11}r_{22} - r \qquad (3.46)$$

(3.42), (3.44)-(3.46)'dan görülebileceği gibi X_{cg} sabit kaldığında G_T'deki artma ile R_{cg} azalacaktır (Şekil 3.6) ve maksimum kazanca sadece yarıçapı r_g sıfıra eşit olduğu noktada ulaşılır. Bu nedenle (3.44) ifadesi sıfıra eşitlenirse;

$$P = Q \mp \sqrt{(Q^2 - |z|^2)} \tag{3.47}$$

elde edilir ve bu ifadeye iki kapılının koşulsuz kararlı olması durumunda ulaşılacaktır.

$$Z_{i\max} = Z_{cg\max} = R_{cg\max} + jX_{cg} \tag{3.48}$$

Burada;

$$R_{cg\max} = \frac{1}{2r_{22}}\sqrt{(Q^2 - |z|^2)} \tag{3.49}$$

ve X_{cg} (3.43)'de daha önceden belirlenmiştir.

Bu sayede verilen giriş VSWR`ı karşılayan maksimum transduser güç kazancı (3.45) ve (3.47)'ye eşitleyerek aşağıdaki şekilde çıkarabiliriz (Güneş Filiz, Güneş M., Fidan M., 1994):

$$G_{T\max} = \{Q - \sqrt{(Q^2 - |z|^2)}\}\frac{1 - |\rho_i|^2}{|z_{12}|^2} \tag{3.50}$$

Bir iki kapılının her iki kapısında da kompleks eşlenik uydurma söz konusu ise $(|\rho_i| = 0)$ maksimum kullanılabilir güç kazancına (MAG: maxsimum available gain) ulaşılmış olur.

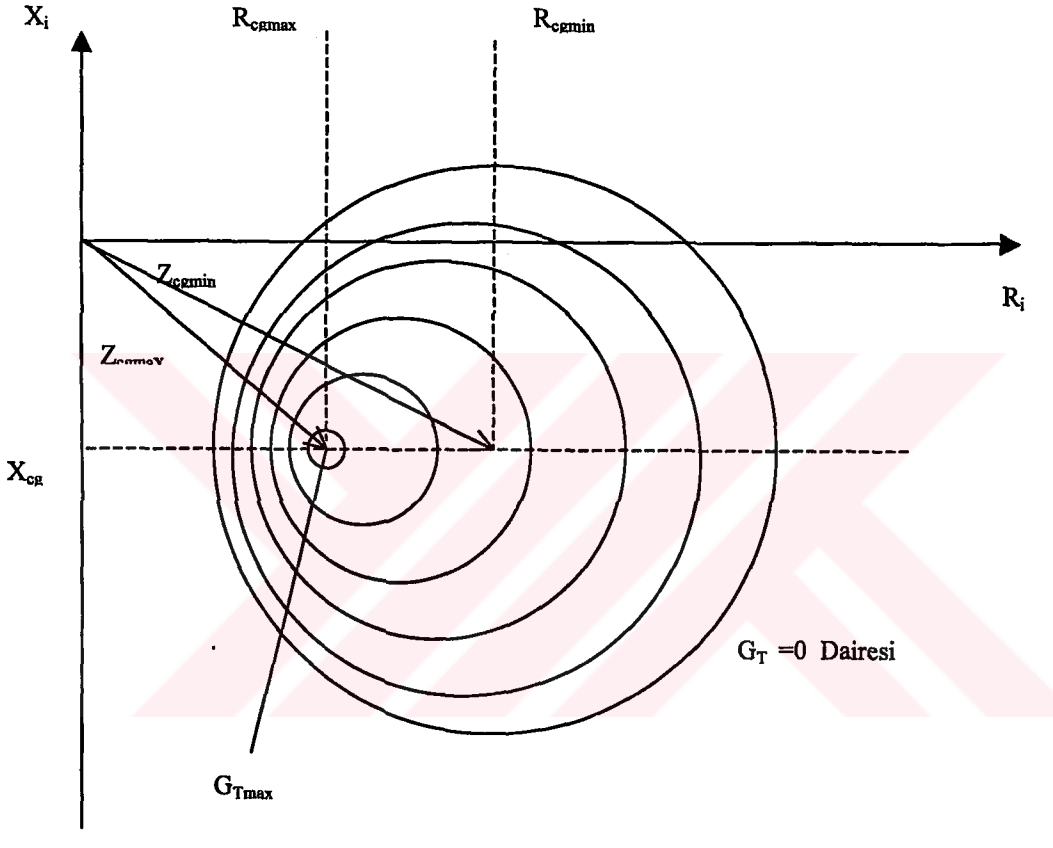
G_T=0 dairesinin verilen daire ailesinin bir limit dairesi olduğunu kolayca bulabiliriz. (3.42) ve (3.44)'de G_T=0'i yerine koyarak aşağıdaki gibi G_T=0 dairesinin merkezini $(Z_{cg\min})$ ve yarıçapını $(r_{g\min})$ bulabiliriz.

$$Z_{cg\min} = R_{cg\min} + jX_{cg} \tag{3.51}$$

$$R_{cg\min} = \frac{Q}{2r_{22}} \tag{3.52}$$

$$r_{g\min} = \frac{|z|}{2r_{22}} \tag{3.53}$$

Devrenin mutlak kararlılığı için $Q > |z|$ olduğunda $R_{cg\min} > Z_{cg\max}$ dur. Sonuç olarak limit daire bütünüyle pozitif reel düzlemdedir ve $G_T > 0$ olan tüm daireler, limit $G_T = 0$ dairesinin içindedir (Şekil 3.6) (Güneş Macit, 1980).

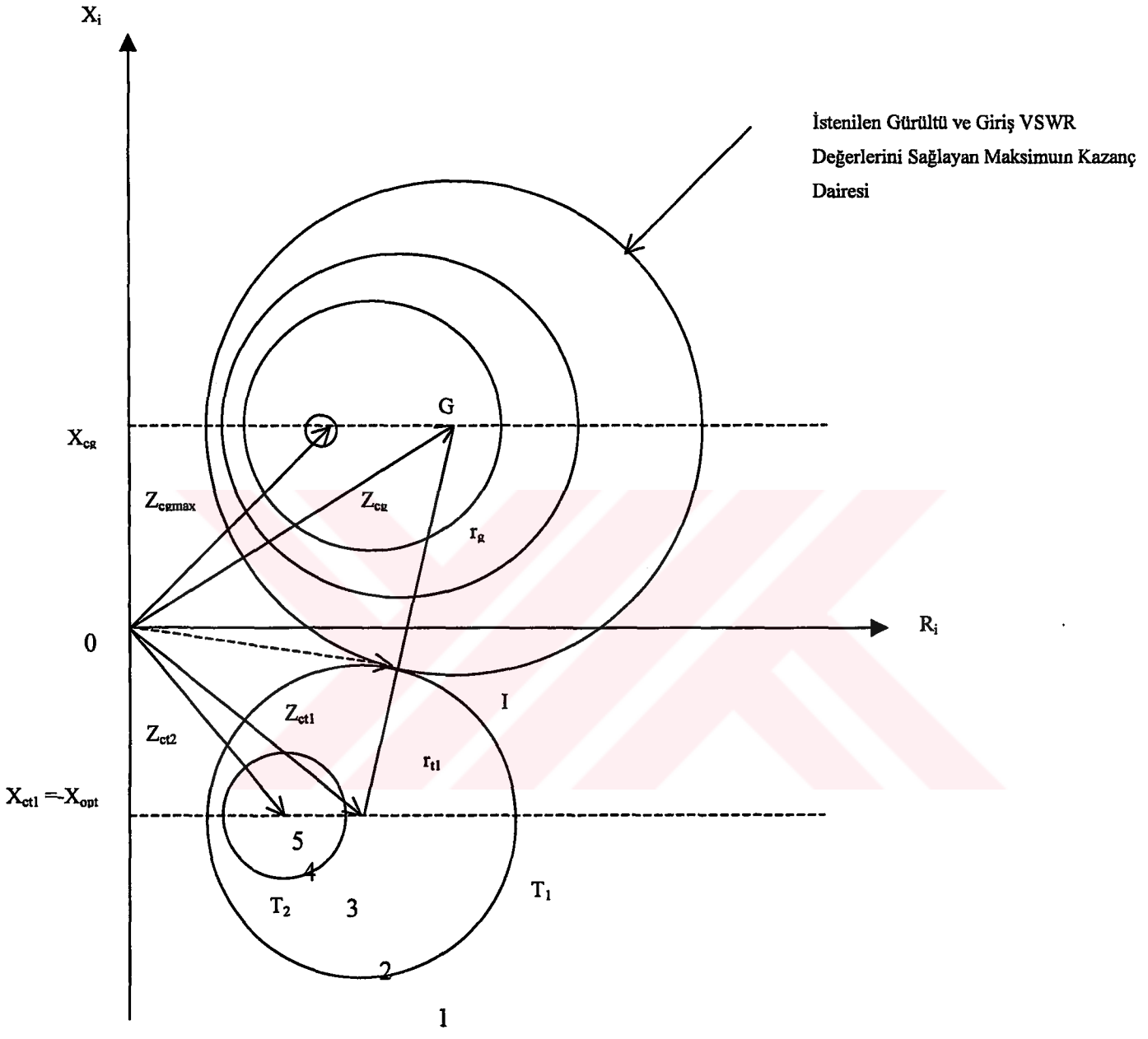


Şekil 3.6 İstenilen Giriş VSWR'ı Sağlayan Sabit kazanç Daireleri

3.4.2 Verilen Giriş VSWR ve Gürültü Faktörünü Sağlayan Koşulsuz Kararlı Transistorun Kazanç Daireleri ve Empedans Verileri

$r_{11} > 0$, $r_{22} > 0$ ve $(Q/|z|) > 1$ olması durumunda koşulsuz kararlı durum (Güneş Macit, 1980) söz konusudur ve koşulsuz kararlılık şartı altında geometrik çözüm $Z_{cg\max}$'ın pozisyonuna bağlı olarak çıkartılabilir (Şekil 3.7). Örneğin Zcgmax 1.bölgede ise maksimum kazanç T1 dairesine teğet sabit kazanç dairesinin değeri olacaktır. Benzer biçimde Zcgmax 5.bölgede

olduğu zaman maksimum kazanç T2 dairesi ile teğet sabit kazanç dairesinin değerini alır.



Şekil 3.7 Verilen Giriş VSWR ve Gürültü Faktörünü Sağlayan İstenilen ve Maksimum Kazanç Daireleri

Ayrıca fiziksel olmayan çözümler için iki olasılık daha akla gelir. Bunlar, $G_T = 0$ limit dairesi, T1 dairesinin tamamen dışında veya T2 dairesinin tamamen içinde olmasının sonucudur. $Z_{cg\,max}$`in 3.bölgede olması durumunda gürültü ve giriş VSWR daireleri Zs-

düzleminde birbirlerini iki noktada keserler. 2. ve 4.bölgelerdeki $Z_{cg\max}$, Zs-düzleminde gürültü ve giriş VSWR dairelerinin teğet olmalarını gerektirecektir. Bütün bu olasılıklar çıkarılacak analitik çözümler için ayrı ayrı incelenecektir.

3.4.2.1 Zcgmax 1.bölgede

Kazancı maksimum yapan giriş empedansı $Z_{i\max} = Z_{cg\max}$ değeri Şekil 3.7'de gösterilen bölgelerden 1.bölgede bulunduğu durum için, kazanç dairelerinden biri ve T1 dairesinin teğet durumlarını yazarsak:

$$\left|Z_{cg} - Z_{ct1}\right|^2 = (r_{t1} + r_g)^2 \tag{3.54}$$

yada;

$$\left|Z_{cg}\right|^2 + \left|Z_{ct1}\right|^2 - 2R_{cg}R_{ct1} - 2X_{cg}X_{ct1} = r_{t1}{}^2 + r_g{}^2 + 2r_{t1}r_g \tag{3.55}$$

yazılabilir. R_{cg}, X_{cg}, r_g ve R_{ct1}, X_{ct1}, r_{t1} için daha önce Bölüm 3.4'de belirlenen ifadeler kullanılırsa aşağıdaki ifade elde edilir (Güneş Filiz, Güneş M., Fidan M., 1994).

$$P^2(1-F^2) - 2(Q+EF)P + \left|z\right|^2 - E^2 = 0 \tag{3.56}$$

Burada;

$$E = \frac{Dr_{22} - R_{ct1}(2r_{11}r_{22} - r) - 2X_{ct1}X_{cg}r_{22}}{r_{t1}} \tag{3.57}$$

$$F = \frac{R_{ct1}}{r_{t1}} \tag{3.58}$$

$$D = \frac{r_{22}\left|z_{11}\right|^2 - rr_{11} - xx_{11}}{r_{22}} + \left|Z_{opt}\right|^2 \tag{3.59}$$

şeklinde ifade edilirler. Q, (3.46) ifadesinde tanımlanmıştır.

(3.56) çözülür ve (3.45) de yerine konursa, sonuçta G_T:

$$G_{T1,2} = \frac{1-\left|\rho_i\right|^2}{\left|z_{12}\right|^2}\frac{1}{1-F^2}\{Q + EF \pm \sqrt{(Q+EF)^2 - (1-F^2)(\left|z\right|^2 - E^2)}\} \tag{3.60}$$

bulunur (Güneş Filiz, Güneş M., Fidan M., 1994).

$F>0$ ve $1-F^2<0$ olduğunda (3.60)'deki negatif işaret daha çok kazanç sağlayacaktır. (3.60)'den görülebileceği gibi maksimum kazanç kaynak veya sonlandırma yükünün direkt fonksiyonu değildir. Bununla beraber istenilen giriş VSWR, gürültü faktörü ve z-parametrelerinin direkt fonksiyonudur. (3.60)'da maksimum kazanca ve dolayısıyla istenilen kazanca ait sonlandırmalar (iki- kapılanın kaynak ve yük empedansları) aşağıda bulunmuştur.

Şekil 3.27'de görüldüğü gibi, sabit kazanç dairesinin T1`e teğet olması durumunda Z_i giriş empedansı aşağıdaki şekilde ifade edilebilir.

$$Z_i = \frac{r_g}{r_{t1}+r_g} Z_{ct1} + \frac{r_{t1}}{r_{t1}+r_g} Z_{cg} \tag{3.61}$$

Burada Z_{ct1}, r_{t1} ve Z_{cg}, r_g sırasıyla T1 ve G_T dairelerin merkez ve yarıçaplarıdır

Yük empedansı giriş empedansı ile bağlantılı olarak (3.4) ifadesinden kolayca bulunabilir:

$$Z_L = \frac{z_{12}.z_{21}}{z_{11}-Z_i} - z_{22} \tag{3.62}$$

Uygun kaynak empedansı Z_S, Zs-düzleminde hem gürültü hem de giriş VSWR dairesinin teğet noktasındaki empedansıdır (Şekil. 3.3a) ve formülasyon olarak aşağıdaki şekilde ifade edilir.

$$Z_S = \frac{r_n}{r_n+r_v} Z_{cv} + \frac{r_v}{r_v+r_n} Z_{cn} \tag{3.63}$$

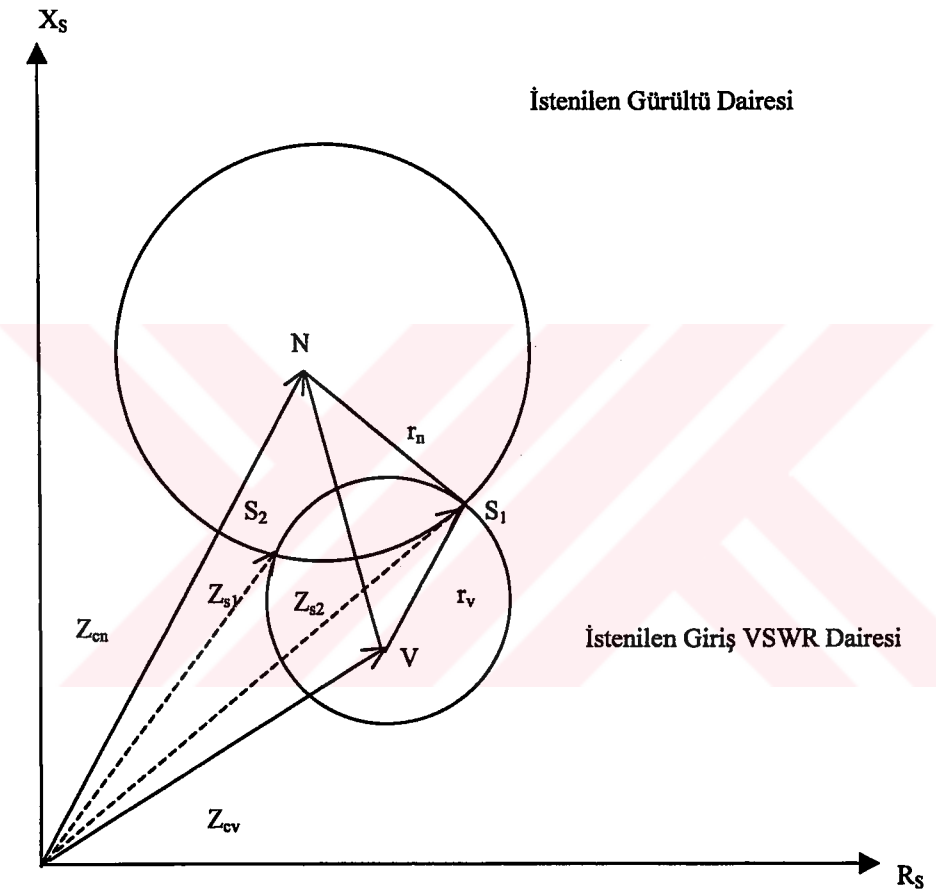
Burada Z_{cn}, r_n ve Z_{cv}, r_v sırasıyla gürültü ve giriş VSWR dairelerinin merkezleri ve yarıçaplarıdır.

3.4.2.2 Zcgmax 2.bölgede

Zcgmax 2.bölgede olduğu zaman Zs-düzleminde giriş VSWR dairesini gürültü dairesine teğet yapan bütün giriş empedanslarının geometrik yeri T1 dairesinin kendisi olur ve maksimum kazanç (3.50)`da verilen $G_{T\max}$`dır ve bu kazancın elde edildiği giriş empedansı (3.48)`de verilen $Z_{cg\max}$ olacaktır. Kaynak empedansı da (3.63) formülünden hesaplanabilir.

3.4.2.3 Zcgmax 3.bölgede

$Z_{cg\max}$ 3.bölgede olduğu zaman gerekli giriş VSWR dairesi Zs-düzleminde gürültü dairesini kesecektir. Bu durumda hem verilen giriş VSWR'ı hem de gürültüyü gerçekleyen 2 kaynak empedansı vardır (Şekil 3.8). $Z_{cg\max}$ 3.bölgede olduğu sürece maksimum kazanç (3.50)'de verilen $G_{T\max}$ olacaktır ve Z_i (3.48)'de verilen $Z_{cg\max}$ olacaktır.



Şekil 3.8 Zcgmax 3.bölgedeyken Kaynak Empedansları

Kaynak empedansları Z_{S1}, Z_{S2} için Şekil 3.8'de iki dairenin kesişme noktalarından aşağıdaki şekilde çıkarılabilir:

$$Z_{S1,2} = R_{S1,2} + jX_{S1,2} \tag{3.64}$$

$$R_{S1,2} = C \pm \sqrt{C^2 - D} \tag{3.65}$$

$$X_{S1,2} = AR_{S1,2} + B \tag{3.66}$$

$$A = \frac{R_{cn} - R_{cv}}{X_{cv} - X_{cn}} \tag{3.67}$$

$$B = \frac{|Z_i|^2 - |Z_{opt}|^2}{2(X_{cv} - X_{cn})} \tag{3.68}$$

$$C = \frac{R_{cn} + AX_{cn} - AB}{1 + A^2} \tag{3.69}$$

$$D = \frac{B^2 + |Z_{opt}|^2 - 2X_{cn}B}{1 + A^2} \tag{3.70}$$

3.4.2.4 Zcgmax 5.bölgede

Bu durumda maksimum kazanç, sabit kazanç dairesinin ve T2 dairesinin birbirine iç içe teğet oldukları noktada elde edilebilir. Bu çözüm ayrıca gürültü ve giriş VSWR dairelerinin, Zs-düzleminde iç teğet çözümlerine karşılık düşer. Şekil 3.9'daki iki dairenin teğet pozisyon koşulu için şu eşitlik yazılabilir:

$$|Z_{cg} - Z_{ct2}|^2 = (r_g - r_{t2})^2 \tag{3.71}$$

Bu (3.54)'e çok benzeyen bir eşitliktir. Yani Zcgmax 1.bölgede iken elde edilen sonuçlar da, $Z_{cg\max}$ 5.bölgede iken Z_{ct1}, r_{t1} yerine $Z_{ct2}, -r_{t2}$ konularak yeniden çıkartılabilir.

İki daire teğet olduğu noktadaki giriş empedansı Z_i yine aşağıdaki formül yardımıyla bulunabilir.

$$Z_i = \frac{r_g}{r_g - r_{t2}} Z_{ct2} - \frac{r_{t2}}{r_g - r_{t2}} Z_{cg} \tag{3.72}$$

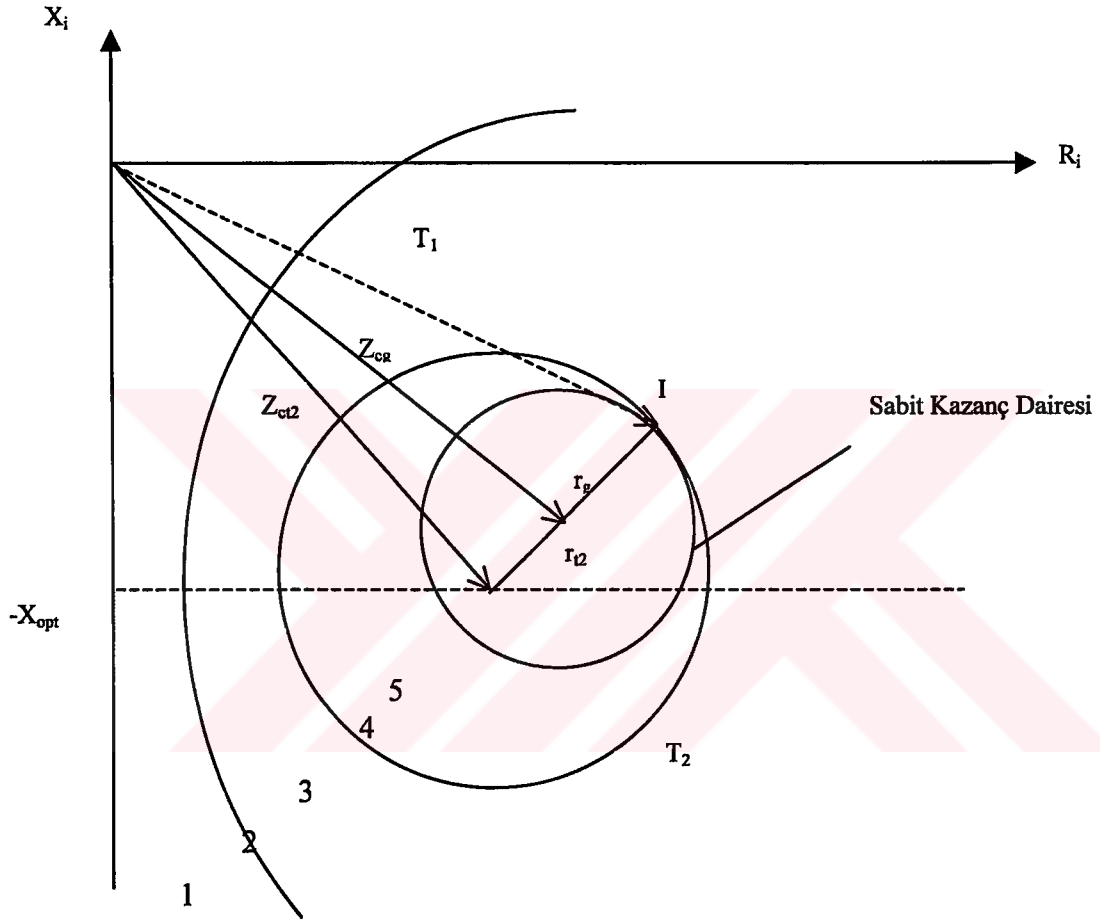
Verilen gürültü ve giriş VSWR dairelerinin Şekil 3.3b'deki iç teğet noktasındaki kaynak empedansı hangi dairenin daha geniş olduğuna bağlı olarak (3.63) ifadesine benzer olarak aşağıdaki şekilde bulunur:

$r_n > r_v$ olduğunda;

$$Z_S = \frac{r_n}{r_n - r_v} Z_{cv} - \frac{r_v}{r_n - r_v} Z_{cn} \tag{3.73}$$

$r_n <= r_v$ olduğu durumda ise,

$$Z_S = \frac{r_v}{r_v - r_n} Z_{cn} - \frac{r_n}{r_v - r_n} Z_{cv} \tag{3.74}$$



Şekil 3.9 Zcgmax 5.bölgedeyken Kazanç Daireleri ve Sonlandırmalar

3.4.2.5 Zcgmax 4.bölgede

Bu durumda maksimum kazanç ve uygun giriş empedansı daha önce tanımlanan $G_{T\max}$ ve $R_{cg\max}$ olur. Kaynak empedansı gürültü ve giriş VSWR dairelerinin yarıçaplarına bağlı olarak (3.73) veya (3.74)'den birisi ile bulunur.

3.5 Koşullu Kararlı Mikrodalga Transistoru

3.5.1 Kaynak Kararlılık Dairesi

$r_{11} > 0$, $r_{22} > 0$ ve $0 < (Q/|z|) < 1$ olması durumunda koşullu kararlı durum (Güneş Macit, 1980) söz konusudur. (3.40) ifadesi kaynak kararlılık dairesinin merkezi ve yarıçapı göz önüne alınarak yeniden düzenlenirse;

$$|Z_i|^2 + 2(R_{cs} + S)R_i + 2X_{cs}X_i + |Z_{cs}| - r_s^2 = 0 \tag{3.75}$$

bulunur ve burada;

$$S = \frac{|z_{12}|^2 G_T}{2r_{22}(1-|\rho_i|^2)} \tag{3.76}$$

şeklindedir . Ayrıca $Z_{cs} = R_{cs} + jX_{cs}$ ve r_s sırasıyla kaynak kararlılık dairesinin merkezi ve yarıçapıdır ve aşağıdaki şekilde ifade edilirler.

$$R_{cs} = -\frac{2r_{11}r_{22} - r}{2r_{22}} \tag{3.77}$$

$$X_{cs} = -\frac{2x_{11}r_{22} - x}{2r_{22}} \tag{3.78}$$

$$r_s = \frac{|z|}{2r_{22}} \tag{3.79}$$

Burada kazanç dairesine ilişkin merkez $Z_{cg} = R_{cg} + jX_{cg}$ ve yarıçap r_g;

$$R_{cg} = -(R_{cs} + S) \tag{3.80}$$

$$X_{cg} = -X_{cs} \tag{3.81}$$

$$r_g = (S^2 + 2SR_{cs} + r_s^2) \tag{3.82}$$

olarak ifade edilirler (Güneş Filiz, Güneş M., Fidan M., 1994).

(3.75)-(3.80) ifadelerinde G_T dairelerinin özellikleri aşağıdaki şekilde ifade edilebilir (Şekil 3.10) (Güneş Macit, 1980).

a) Tüm G_T daireleri imajiner ekseni aynı noktada keser. Bu nokta kompleks eşlenik kaynak kararlılık dairelerinin imajiner ekseni kestiği noktadır.

b) $G_{T\min}$ dairesi ($G_T = 0$), $Z_{cg\min} = -R_{cs} - jX_{cs}$ merkezine ve $r_{g\min} = r_s$ yarıçapına sahiptir ve imajiner eksene göre kompleks eşlenik kaynak kararlılık dairesinin simetriğidir.

c) $G_{T\max} > G_{Treq} > 0$ daireleri $G_T = 0$ ve kompleks eşlenik kaynak karalılık dairesinin Zi-düzleminin pozitif reel kısmında kalan arkı ile sınırlandırılmış olacaklardır. Bu dairelerin merkez fazörü her zaman $X_i = -X_{cs}$ çizgisi üzerinde olacaklardır. Koşulsuz kararlı durumda tüm G_T daireleri Zi-düzleminde sağ yarıda kalmaktaydılar. Koşullu kararlı durumda ise maksimum kazanç kompleks eşlenik kaynak kararlılık dairesinin sağ yarıda kalan ark üzerinde olacaktır.

(3.89) ifadesinde $R_{cg} = R_{cs}$ alınıp maksimum kazanç;

$$G_T = 2\frac{1-|\rho_i|^2}{|z_{12}|^2}(2r_{11}r_{22}-r) \tag{3.83}$$

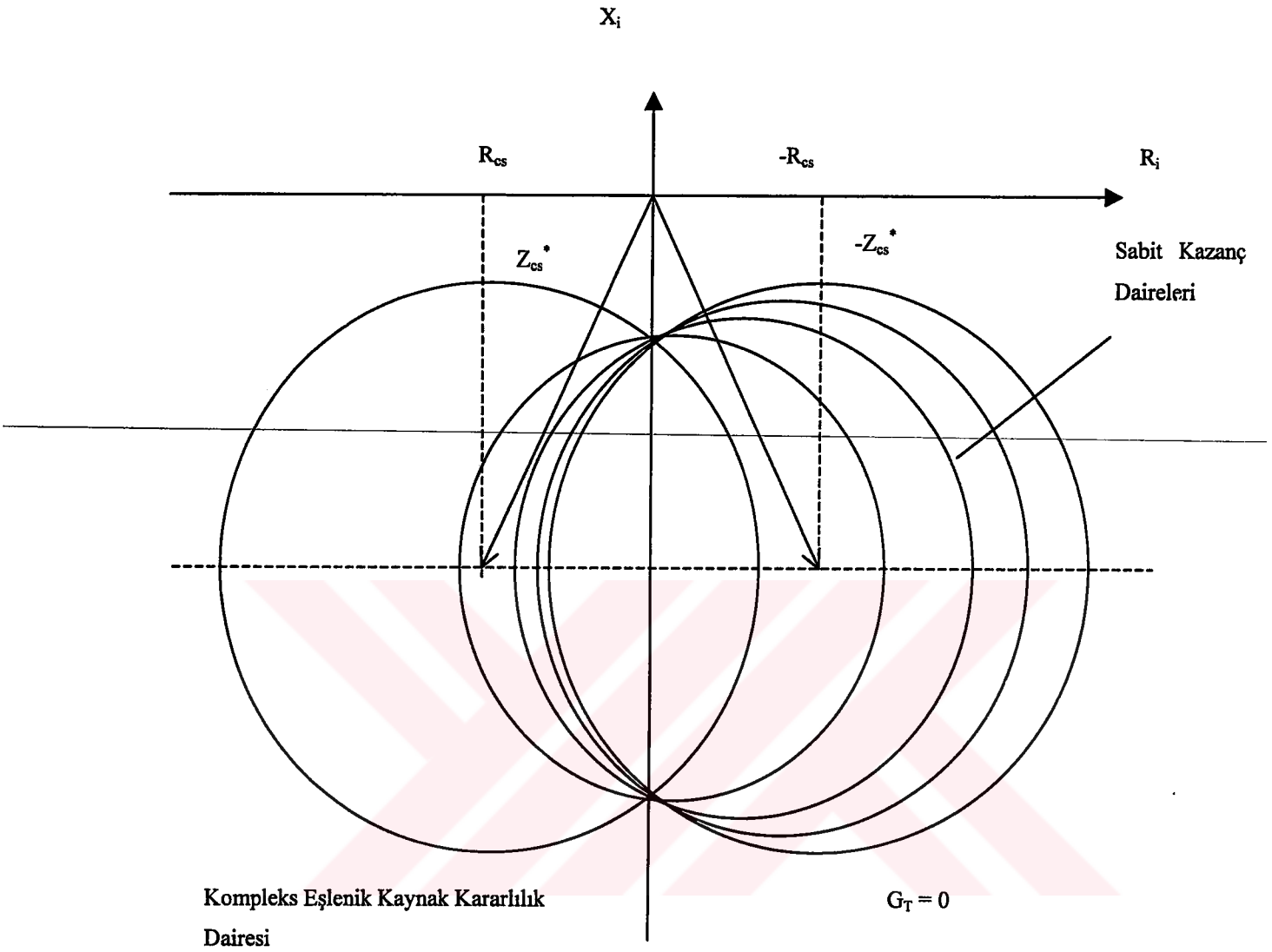
olanak belirlenir Burada $|\rho_i| = 0$ alınırsa maksimum kararlı kazanç (MSG; Maximum Stable Gain);

$$MSG = 2\left|\frac{z_{21}}{z_{12}}\right|\eta \tag{3.84}$$

olanak bulunur. Burada η kararlılık faktörüdür ve;

$$\eta = \frac{2r_{11}r_{12}-r}{|z|} \tag{3.85}$$

olarak ifade edilir. Koşullu kararlı durumda $0< \eta <1$ şeklindedir (Güneş Macit, 1980).



Şekil 3.10 Koşullu Kararlı Transistor İçin Zi-düzleminde Sabit Kazanç Daireleri

3.5.2 Verilen Giriş VSWR ve Gürültü Faktörünü Sağlayan Koşullu Kararlı Transistorun Kazanç Daireleri ve Empedans Verileri

$r_{11} > 0$, $r_{22} > 0$ ve $0 < (Q/|z|) < 1$ olması durumunda koşullu kararlı durumun (Güneş Macit, 1980) söz konusu olduğundan daha önceki bölümlerde bahsedilmişti. Koşulsuz kararlı durumda çözüme beş farklı bölge üzerinden ulaşılmıştı. Koşulsuz kararlı durum için yapılan bölgelendirme Şekil 3.11'de gösterilen biçimde koşullu kararlı durum için de yapılabilir.

Koşullu kararlı transistor durumunda dört durum söz konusudur (Güneş Macit, 1980):

3.5.2.1 a) Durumu

$G_T = 0$ dairesi tamamen 1.bölgedeyse fiziksel çözüm yoktur. Bu durum dairenin merkez ve yarıçapı şöyle tanımlanabilir :

$$\left|Z_{cg\min} - Z_{ct1}\right| > r_s + r_{t1} \tag{3.86}$$

3.5.2.2 b) Durumu

2.bölge (T1 dairesi) sabit kazanç dairesi alanını kestiğinde bir maksimum kazanç olasılığı mevcuttur. Maksimum kazanç ve uygun giriş empedansı bir sabit kazanç dairesiyle Şekil 3.11'de gösterilen T1 dairesinin teğet bağıntısından hesaplanabilir:

$$\left|Z_{cg} - Z_{ct1}\right|^2 = (r_g + r_{t1})^2 \tag{3.87}$$

Maksimum kazanç Bölüm 3.6'da yapıldığı gibi (3.80)-(3.82) ve (3.77)-(3.79) ifadelerinin (3.87)'de yerine konmasıyla çıkarılabilir. Uygun giriş empedansı aşağıdaki şekilde yazılır:

$$Z_i = \frac{r_{t1}}{r_{t1} + r_g} Z_{cg} + \frac{r_g}{r_{t1} + r_g} Z_{ct1} \tag{3.88}$$

Z_i bulunduktan sonra kaynak empedansı (Zs) (3.63)'den hesaplanabilir ki burada gürültü ve giriş VSWR daireleri Zs-düzleminde dış teğet konumundadırlar. Bu durumun varlığı aşağıdaki bir çift eşitsizlikle gösterilebilir.

$$\left|Z_{cg\max} - Z_{ct1}\right| > r_s + r_{t1} > \left|Z_{cg\min} - Z_{ct1}\right| \tag{3.89}$$

3.5.2.3 c) Durumu

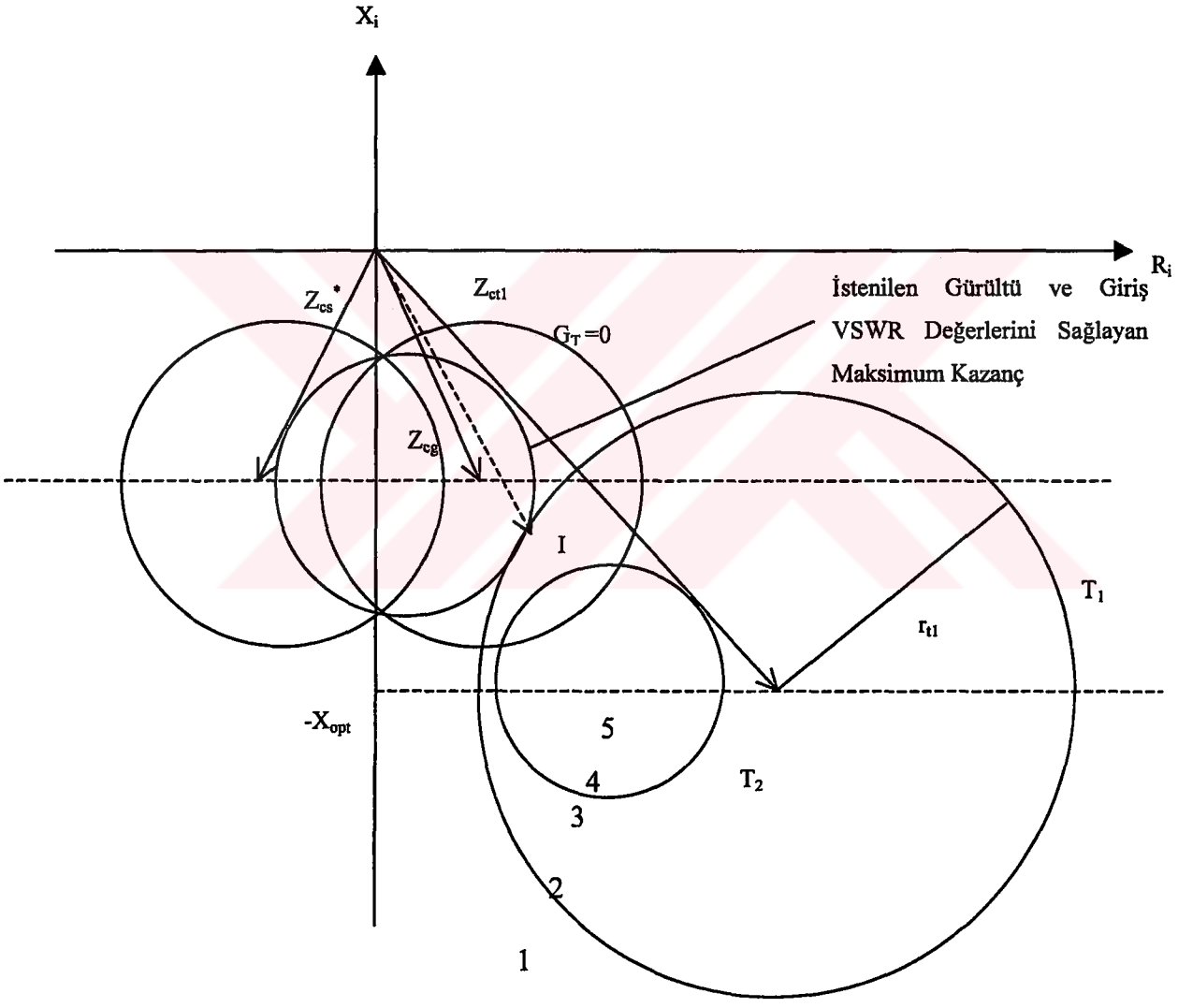
T1 dairesi, kaynak kararlılık dairesinin eşleniğinin kestiği zaman; bu kararlılık dairesinin eşleniğinin bazı bölümleri 3.bölğede kalır. Bu yüzden bu bölgedeki giriş empedansı (3.83)'de verilen maksimum kazançla aynı kazancı sağlar (Şekil 3.12). Bu durumda gürültü ve giriş VSWR dairelerinin şartlarını sağlayan iki kaynak empedansı olacaktır. Bunlar (3.64)-(3.70) arası hesaplanmışlardır. Bu durum aşağıdaki bir çift eşitsizlikle ifade edilebilir:

$$\left|Z_{cg\max} - Z_{ct2}\right| > r_s + r_{t2} \tag{3.90}$$

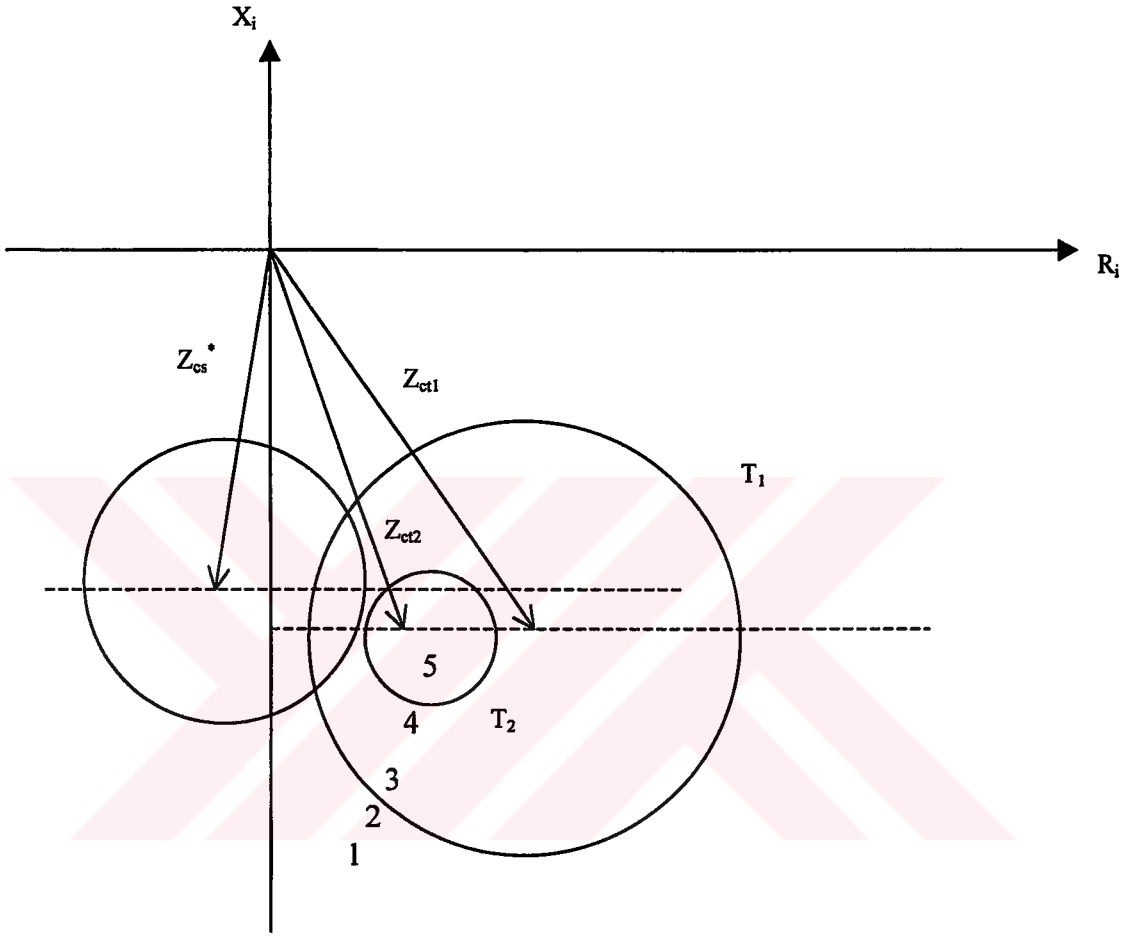
$$\left|Z_{cg\max} - Z_{ct1}\right| < r_s + r_{t1} \tag{3.91}$$

3.5.2.4 d) Durumu

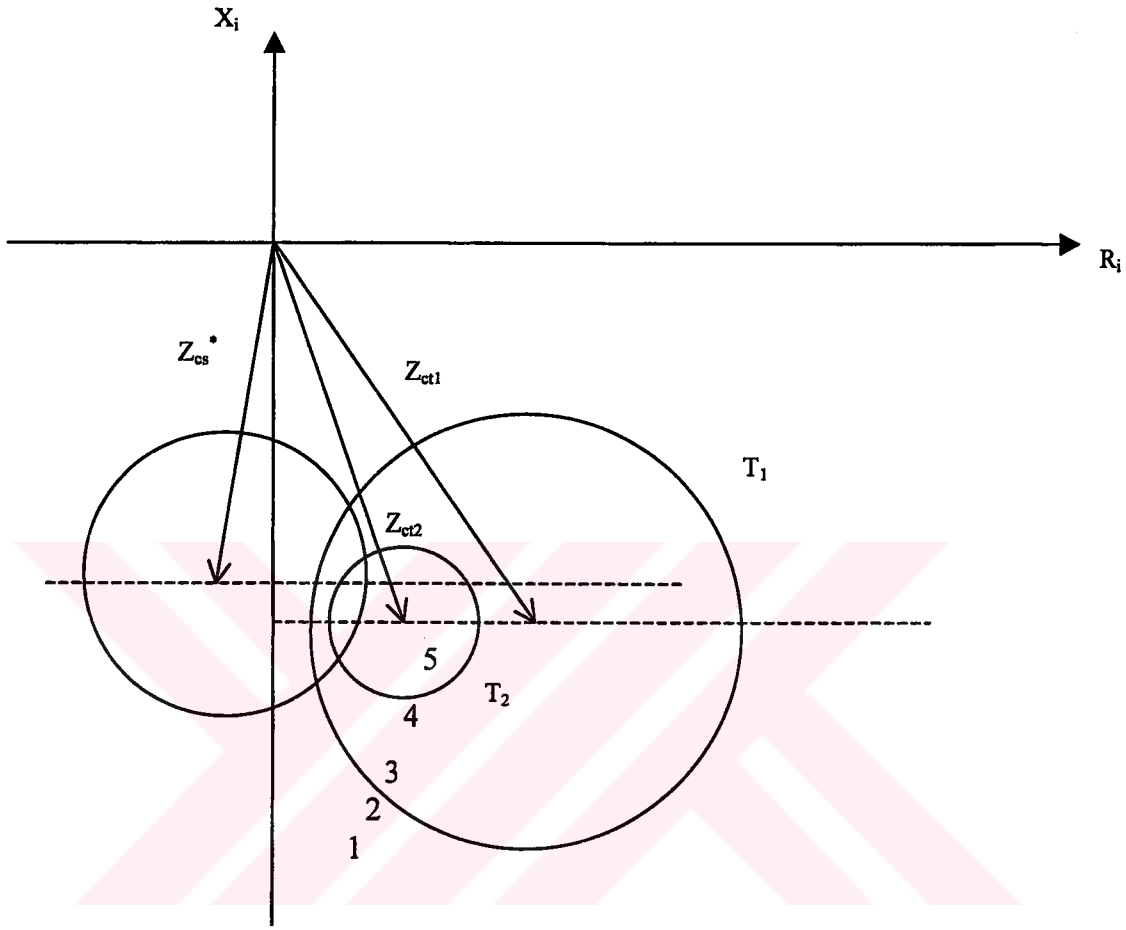
T2 dairesi eşlenik kaynak dairesini kestiğinde bu dairenin 5.bölgedeki bölümü çözüm vermez. 3.bölgedeki bölümü maksimum kazanç ve sonlandırma sağlar (2. durumda açıklandığı gibi) (Şekil 3.13).



Şekil 3.11 Koşullu Kararlı Transistor İçin İstenilen Giriş VSWR ve Gürültüyü Sağlayan Kazanç Daireleri ve Sonlandırmalar



Şekil 3.12 T1 Dairesinin Kaynak Kararlılık Dairesinin Eşleniğini Kestiği Durum



Şekil 3.13 T2 Dairesinin Kaynak Kararlılık Dairesinin Eşleniğini Kestiği Durum

4. EMPEDANS VERİLERİNİN MODELLENMESİ

4.1 Empedans Yaklaşımı İle Modelleme

Fiziksel devreden ölçülen veya herhangi bir analiz yöntemiyle bulunan veriler, pozitif reel bir $z(s)$ empedans fonksiyonu ile modellendiği zaman bu $z(s)$ empedans fonksiyonu sentez edilebilir. Ölçülen verilerdeki reel ve sanal kısımları ayrı ayrı modellemek, Empedans Yaklaşımı ile Modelleme yöntemimizin esasını oluşturur. Reel ve sanal kısımları ayrı ayrı modellemek için, (Kılınç Ali, 1995)'de konu edilen aşağıdaki yöntem kullanılmıştır.

Pozitif reel bir $z(s)$ empedans fonksiyonu, bir Foster fonksiyonu $z_F(s)$ ile bir minimum reaktans fonksiyonunun $z_{MR}(s)$ toplamı olarak yazılabilir (Kılınç Ali, 1995).

Verilen bir $z(s)$ fonksiyonu; reel s değerleri için $z(s)$ reel ve $\mathrm{Re}\{s\} \geq 0$ için $\mathrm{Re}\{z(s)\} \geq 0$ koşullarını sağladığında pozitif reeldir denir (Kılınç Ali, 1995).

Yukarıda tanımı verilen pozitif reel bir immitans fonksiyonunun kutupları, yalnızca jω ekseni üzerinde yada sol tarafında olabilir, sağ yarı düzlemde bulunamazlar. Bu kutuplardan jω üzerinde olanlar Foster fonksiyonu olarak, sol yarı düzlemde olanlar ise minimum reaktans/süseptans fonksiyonu olarak yazılabilir. Bunun için bir pozitif reel empedans fonksiyonu göz önüne alındığında;

$$z(s) = \frac{N(s)}{D(s)} = \frac{a_0 + a_1 s + ... + a_m s^m}{b_0 + b_1 s + ... + b_m s^m} \tag{4.1}$$

(4.1) ile verilen bir pozitif reel fonksiyon olduğuna göre, $N(s)$ ve $D(s)$ polinomlarının sağ yarı düzlemde kökleri yoktur. $D(s)$ 'in jω ekseni üzerindeki köklerinden $D_1(s)$ oluşturulursa;

$$D_1(s) = (s^2 + \omega_1^2)(s^2 + \omega_2^2)...(s^2 + \omega_m^2) \tag{4.2}$$

geri kalan köklerinden de $D_2(s)$ oluşturulursa;

$$D_2(s) = \frac{D(s)}{D_1(s)} \tag{4.3}$$

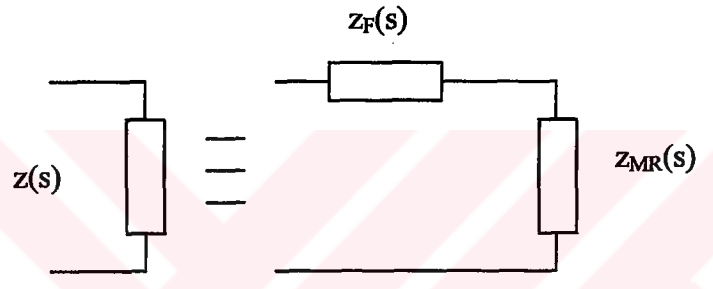
Bu durumda $D_2(s)$ polinomunun jω ekseni üzerinde kutbu yoktur, yani Kesin Hurtwitz'dir. Böylece $z(s)$ empedans fonksiyonu;

$$z(s) = \frac{N(s)}{D_1(s).D_2(s)} \tag{4.4}$$

haline gelir. Bu ifade, jω eksenindeki kutuplarına göre kısmı çarpanlarına ayrılırsa;

$$z(s) = \underbrace{\left[k_\infty s + \frac{k_0}{s} + \sum_{i=1}^{m}\left(\frac{k_i.s}{s^2+\omega_i^2}\right)\right]}_{\text{Foster fonksiyonu}} + \underbrace{\frac{N_2(s)}{D_2(s)}}_{\text{minimum reaktans fonksiyonu}} \tag{4.5}$$

gibi iki fonksiyonu toplamı olarak yazılabilir. Bu fonksiyonlardan ilki, jω ekseni üzerindeki kutuplardan oluşmuştur, yani Foster fonksiyonudur, ikincisi ise sol yarı düzlemdeki kutuplardan oluşmuştur, yani minimum reaktans/süseptans fonksiyonudur.



Şekil 4.1. Verilen empedansın iki bölüme ayrılması.

Buradan,

$$z(s) = z_{MR}(s) + z_F(s) \tag{4.6}$$

şeklinde yazılır ki bu pozitif reel bir $z(s)$ empedans fonksiyonu, bir Foster fonksiyon $z_F(s)$ ile bir minimum reaktans fonksiyonu $z_{MR}(s)$ 'in toplamı olarak yazılabileceğinin ifadesidir (Şekil 4.1).

4.2 Empedans Yaklaşımı İle Modelleme Algoritması

1. ω_i, i=1,2,3,...,k frekanslarında yapılmış k adet ölçüm veya analiz sonucu elde edilen veriler empedansın reel ve sanal bileşenleri $Z_D(\omega_i) = R_D(\omega_i) + jX_D(\omega_i)$ şeklinde yazılır.

2. $z_{MR}(\omega_i) = r_{MR}(\omega_i) + jx_{MR}(\omega_i)$ olmak üzere reel kısım verileri $r_{MR}(\omega_i) = R_D(\omega_i)$ 'ya bir eğri uydurulur. Eğrinin fonksiyonu (4.13) 'de verildiği gibi bir çift fonksiyon olmalıdır.

$$r_{MR}(\omega) = \frac{A_0 + A_1\omega^2 + ... + A_m\omega^{2m}}{B_0 + B_1\omega^2 + ... + B_n\omega^{2n}} \tag{4.7}$$

Bu denklemde n≥m olması koşulu ile n minimum reaktans fonksiyonunun kaç eleman ile modelleneceğini, m ise iletim sıfırlarının sayısını belirlemektir. Bu gösteriminde, n ve m giriş empedansının minimum reaktans kısmının kompleksliğini belirlemektedir. Bir çok uygulamada minimum reaktans fonksiyonunun iletim sıfırları sıfırda (DC), sonsuzda, sonlu bir frekansda veya bunların birleşiminden oluşur. Sonuçta elde edilecek $r_{MR}(\omega)$ hiçbir zaman negatif olmamalı, bu amaçla A_i ve B_i katsayıları uyumlu olarak belirlenmelidir.

3. $r_{MR}(\omega)$ elde edildikten sonra minimum reaktans fonksiyonu $z_{MR}(\omega)$ analitik olarak bulunmalıdır. Bu amaçla Bode veya Gewertz prosedürleri kullanılabilir. Sentezin kolaylığı açısından Gewertz prosedürünün tercih edilmesi daha uygundur. Bunun sonucunda $z_{MR}(s)$, (4.14)'deki gibi elde edilir. Bu durumda $x_{MR}(s)$ 'da ulaşılmış olunur.

$$z_{MR}(s) = \frac{a_0 + a_1 s + ... + a_m s^m}{b_0 + b_1 s + ... + b_n \omega^n} \tag{4.8}$$

4. $z_F(\omega_i) = jx_F(\omega_i)$ şeklinde sanal olduğundan, (4.6)'dan $z(j\omega_i) = r_{MR}(\omega_i) + jx_{MR}(\omega_i) + jx_F(\omega_i)$ şeklinde yazılabilir. Bu verilen empedans $Z_D(\omega_i) = R_D(\omega_i) + jX_D(\omega_i)$ 'ye eşitlenirse $x_F(\omega_i) = X_D(\omega_i) - x_{MR}(\omega_i)$ şeklinde minimum reaktans fonksiyonu tarafından modellenmemiş sanal veriler elde edilir (Foster fonksiyonu verileri).

5. Elde edilen Foster verilerine bir eğri uydurulur. Uydurulacak eğrinin fonksiyonu (4.9) ile verilmektedir.

$$x_F(\omega) = -\frac{k_0}{\omega} + k_\infty \omega + \sum_{i=1}^{p} \frac{k_i \omega}{\omega_i^2 - \omega^2} \tag{4.9}$$

Bu denklemde k_0 k_∞ ve k_i katsayılarının pozitif olması sağlanmalıdır. Bu fonksiyondaki kutupların sayısı (p+2) tanedir (sıfırda, sonsuzda ve p tane de ω'da kutup var). $x_F(\omega)$ belirlendikten sonra, $\omega = s/j$ dönüşümü yapılarak $z_F(\omega)$ Foster empedans fonksiyonu bulunur.

$$z_F(s) = \frac{k_0}{s} + k_\infty s + \sum_{i=1}^{p} \frac{k_i s}{s^2 + \varpi_i^2} \tag{4.10}$$

6. Aranılan model fonksiyon $z(s) = z_{MR}(s) + x_F(s)$ şeklinde elde edilir. Daha sonra bulunan fonksiyon, Darlington sentezi yöntemi ile sentezlenerek 1Ω ile sonlandırılmış kayıpsız iki kapılı devre elde edilir (Kılınç Ali, 1995).

4.3 Gewertz Prosedürü

Bu prosedürü kullanarak, minimum reaktans fonksiyonunun reel kısmı bilindiği zaman empedansın tamamı elde edilmektedir. Minimum reaktans fonksiyonunun pay ve paydası s'in kuvvetlerine göre çift ve tek kısımlarına ayrılarak yazılırsa:

$$Z_{MR}(s) = Odd(Z_{MR}(s)) + Even(Z_{MR}(s)) = \frac{N(s)}{D(s)} \tag{4.11}$$

Şimdi $r_{MR}(w)$ fonksiyonunun bilindiği farz edilsin. $s = jw$ dönüşümü yapılarak $r_{MR}(s^2)$ fonksiyonu bulunur.

$$Z_{MR}(s) = \frac{a_0 + a_1 s^1 + \ldots + a_m s^m}{b_0 + b_1 s^1 + \ldots + b_n s^n} \tag{4.12}$$

Minimum reaktans fonksiyonunun yapısını (4.8)'deki gibi vermiştik (n=m+1).

$$r_{MR}(s^2) = \frac{A_0 + A_1 s^2 + \ldots + A_m s^{2m}}{B_0 + B_1 s^2 + \ldots + B_n s^{2n}} \tag{4.13}$$

Minimum reaktans fonksiyonunun reel kısmını ise (4.13)'deki gibi elde edilir.

$$\frac{A_0 + A_1 s^2 + \ldots + A_m s^{2m}}{B_0 + B_1 s^2 + \ldots + B_n s^{2n}} = \frac{m_1 m_2 - n_1 n_2}{m_2^{\,2} - n_2^{\,2}} = \frac{m_1 m_2 - n_1 n_2}{D(s)D(-s)} \tag{4.14}$$

Genel olarak (4.14) eşitliğini elde edilebilir. İlk önce $r_{MR}(s^2)$'nin payda polinomunu ele alınırsa;

$$D(s)D(-s) = (B_0 + B_1 s^2 + \ldots + B_n s^{2n}) \tag{4.15}$$

$D(s)$ polinomunu elde etmek için $r_{MR}(s^2)$'nin payda polinomunu bütün kökleri bulunur, sol yarı düzlemde olan kökler $p_1, p_2 \ldots p_n$ ise $D(s)$ şu şekilde yazılır;

$$D(s) = \sqrt{B_n}(s - p_1)(s - p_2)\ldots(s - p_n) \tag{4.16}$$

$D(s)$'i bulmanın bir başka yolu da aşağıdaki gibi açıklanabilir:

1.Önce $D(s)$ polinomu $D(s) = b_n s^n + b_{n-1}s^{n-1} + \ldots + b_0$ şeklinde tanımlanır.

2.Daha sonra $D(s)\,D(-s)$ çarpımı bulunup, bulunan katsayılar, B_i i=1...n katsayılarına eşitlenir ve böylece n+1 tane denklem elde edilir.

3.Bulunan denklemler çözülürse aranan katsayılar bulunur. Ancak dikkat edilecek husus bu denklemler çözülürken birden fazla cevap bulunabilir. Pozitif katsayılar alınır ve daha sonra bu denklemin kesin Hurwitz olmasına dikkat edilir. İlk yöntemde ise yine aynı şekilde sadece sol yarı düzlemdeki kökler alınarak $D(s)$'in kesin Hurwitz olması sağlanmıştı. İlk yöntemdeki kök bulma işlemi ve ikinci yöntemdeki denklem çözümü Mathcad programı ile rahatlıkla yapılabilmektedir.

Payda polinomu dikkate alınırsa;

$$A_0 + A_1 s^2 + \ldots + A_m s^{2m} = m_1 m_2 - n_1 n_2 =$$

$$(a_0 + a_2 s^2 + \ldots)(b_0 + b_2 s^2 + \ldots) - (a_1 s + a_3 s^3 + \ldots)(b_1 s + b_3 s^3 + \ldots) \tag{4.17}$$

Burada katsayılar düzenlenirse;

$$A_0 = b_0 a_0$$

$$A_1 = b_2 a_0 - b_1 a_0 + b_0 a_2$$

$$A_2 = b_4 a_0 - b_3 a_1 + b_2 a_2 - b_1 a_3 + b_0 a_4 \tag{4.18}$$

$$\vdots$$

$$A_c = \sum_{j=-c}^{c} (-1)^{c+j} b_{c-j} a_{c+j}$$

Genelde eğri uydurma işinde $r_{MR}(w)$'nin pay polinomu aşağıdaki gibidir.

$$N(w) = w^{2k} \prod_{p=1}^{t} (w^2 - w_p{}^2)^2 \tag{4.19}$$

Fonksiyon Pozitif reel olduğu için, fonksiyonun değeri tüm w'lar için asla negatif olamaz yani 0'ın altına inemez, bir fonksiyonun 0 olması için negatif değerler de alabilmesi gerekmektedir, w sonsuza giderken 0'da asimptotu (y=0 ekseni) olmasını fonksiyonun 0 olması olarak düşünemeyiz, böyle kabul edilse de bu ifademize yansıtılamaz. Bu fonksiyonun belli w değerleri için 0'a teğet gelmesi ve negatife inmeden tekrar pozitif bir değer alması ise zor bir ihtimal, bu olayı eğri uydururken kullanmak çok daha zor bir olaydır. Bütün bu varsayımlar doğrultusunda fonksiyonun gidişine göre verebileceğimiz k değerleri ile beraber ($k = 0,1,2...$) yeni pay polinomu şöyledir:

$$N(w) = A_k w^{2k} \tag{4.20}$$

Bizim uygulamalarımızda ise k=0 için en iyi sonuçlar alınmıştır. $N(w) = A_0$ ifadesi kullanılmıştır, öyleyse (4.18) denklemleri daha basit bir hal alıp A_c katsayıları ($c = 1,2,3...m$) 0 olacaktır.

Giriş ve çıkış uydurma devre tasarımına yönelik yöntemler açısından daha detaylı bilgiler (Kılınç Ali, 1995), (Carlin H.J. and Yarman S.B., 1983), (Yarman S.B., Feltuvies A., 1990) ve (Yarman S.B., Akşen A., 1992) kaynaklarında verilmektedir.

4.4 Normalizasyon ve Denormalizasyon

Devre analizinde ve sentezinde Ohm, Farad, Henry, Hertz gibi standart birimlerin kullanılması hesaplamalar sırasında çok büyük ve çok küçük sayılarla çalışmayı gerektirir. Bu tür farklı büyüklüklerdeki sayılarla çalışmak işlemleri zorlaştırdığı gibi yuvarlatma ve kesmeden dolayı hataları da arttırmaktadır. Bu problemlerden kurulmak için normalizasyon ve denormalizasyon yöntemlerine başvurulur.

Empedans yaklaşımı ile modelleme algoritmasında aşağıdaki normalizasyon ve denormalizasyon yöntemine başvurulmuştur (Kılınç Ali, 1995).

Verilen bir devrede bütün empedanslar R_0 ile ve bütün açısal frekanslar w_0 ile normalize edilirse;

$$R = R' / R_0 \tag{4.21}$$

$$w = w' / w_0 \tag{4.22}$$

olur ve bu gösterilimde "'" normalize edilmemiş gerçek eleman değeridir. Normalize edilmiş devrede fiziksel ilişkilerin ve karakteristik özelliklerin değişmemesi için endüktans ve kapasite normalizasyon değerleri;

$$L_0 = R_0 / w_0 \tag{4.23}$$

$$C_0 = 1/(R_0 w_0) \tag{4.24}$$

şeklinde olmalıdır. Bu durumda endüktans ve kapasite normalizasyonu;

$$L = L' / L_0 \tag{4.25}$$

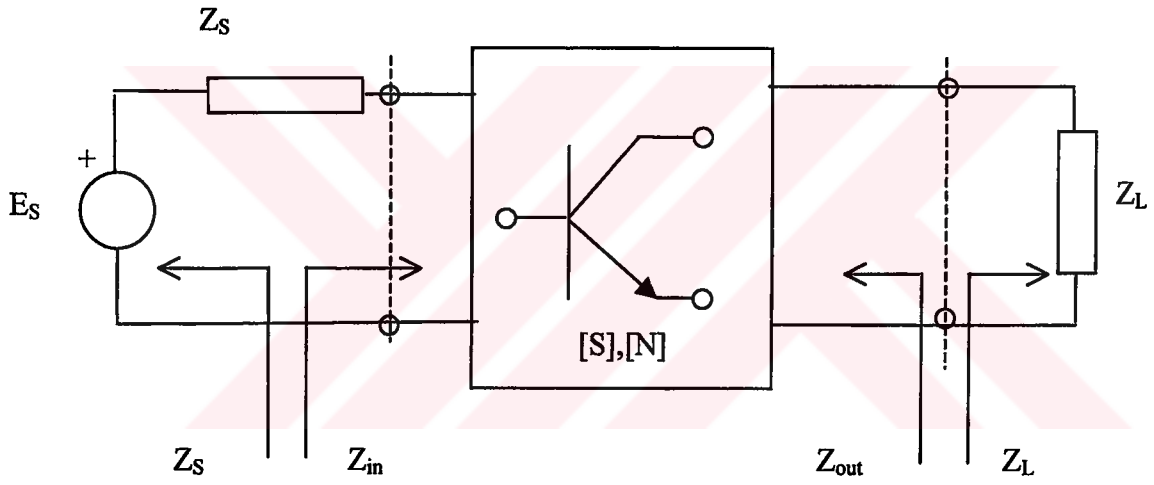
$$C = C' / C_0 \tag{4.26}$$

şeklide olur ve denormalizasyon işlemi yukarıdaki ifadelerde "'" değerlerini çekip gerçek değerlere geçmekle yapılabilir.

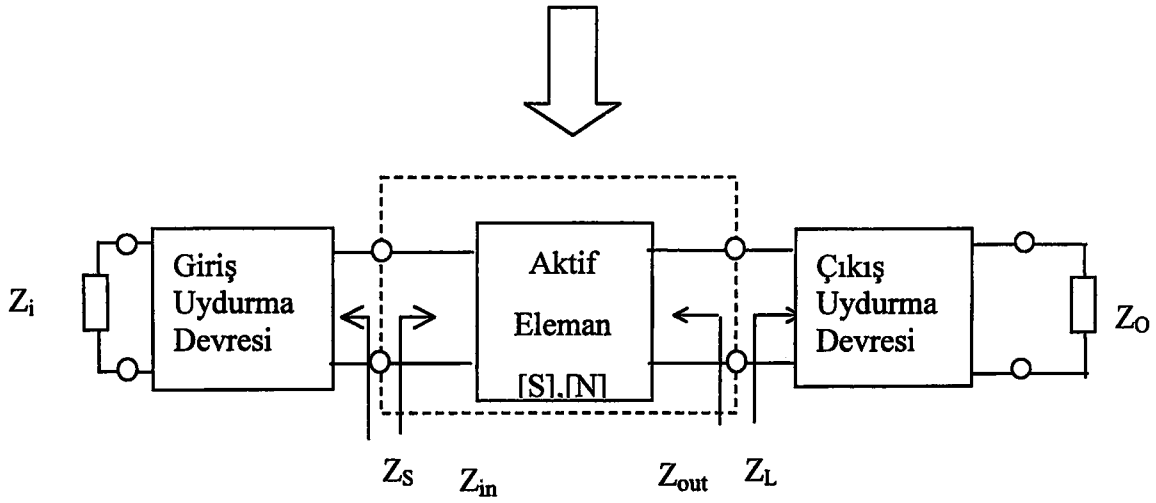
5. MİKRODALGA TRANSİSTORUN YSA İLE PERFORMANS ANALİZİ, MODELLENME BLOKLARI ve PROGRAMLAMA

5.1 Blok Yapıları ve Akış Diyagramları

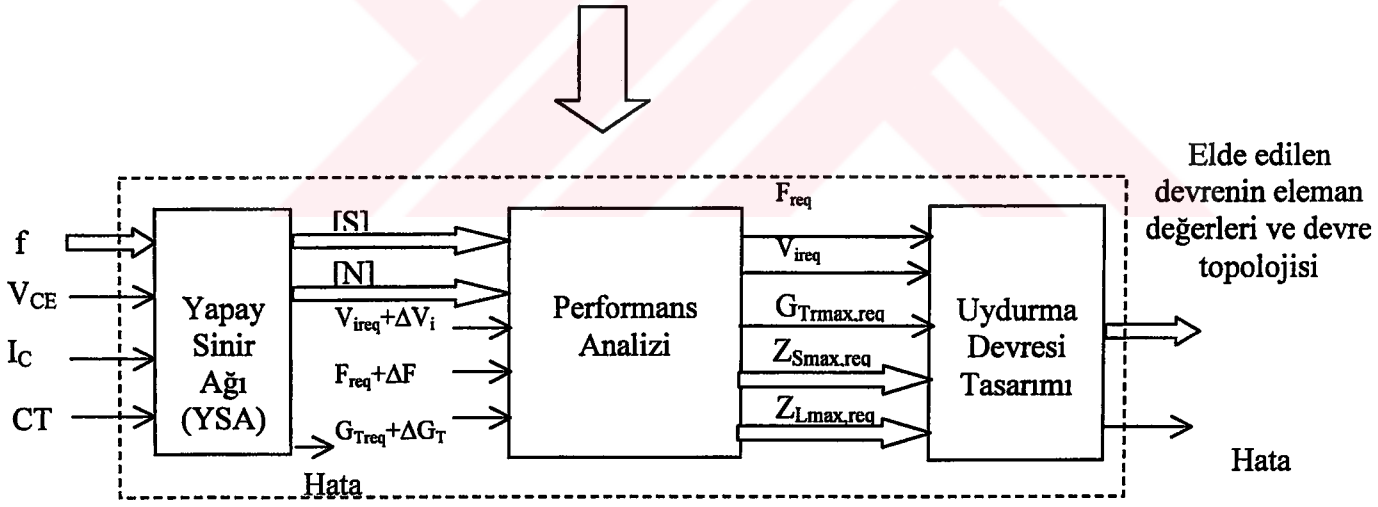
YSA kullanılarak mikrodalga transistorun performans analizi ve modellenmesine yönelik sistemin blok yapıları ve programlara yönelik akış diyagramları aşağıda verilmiştir.



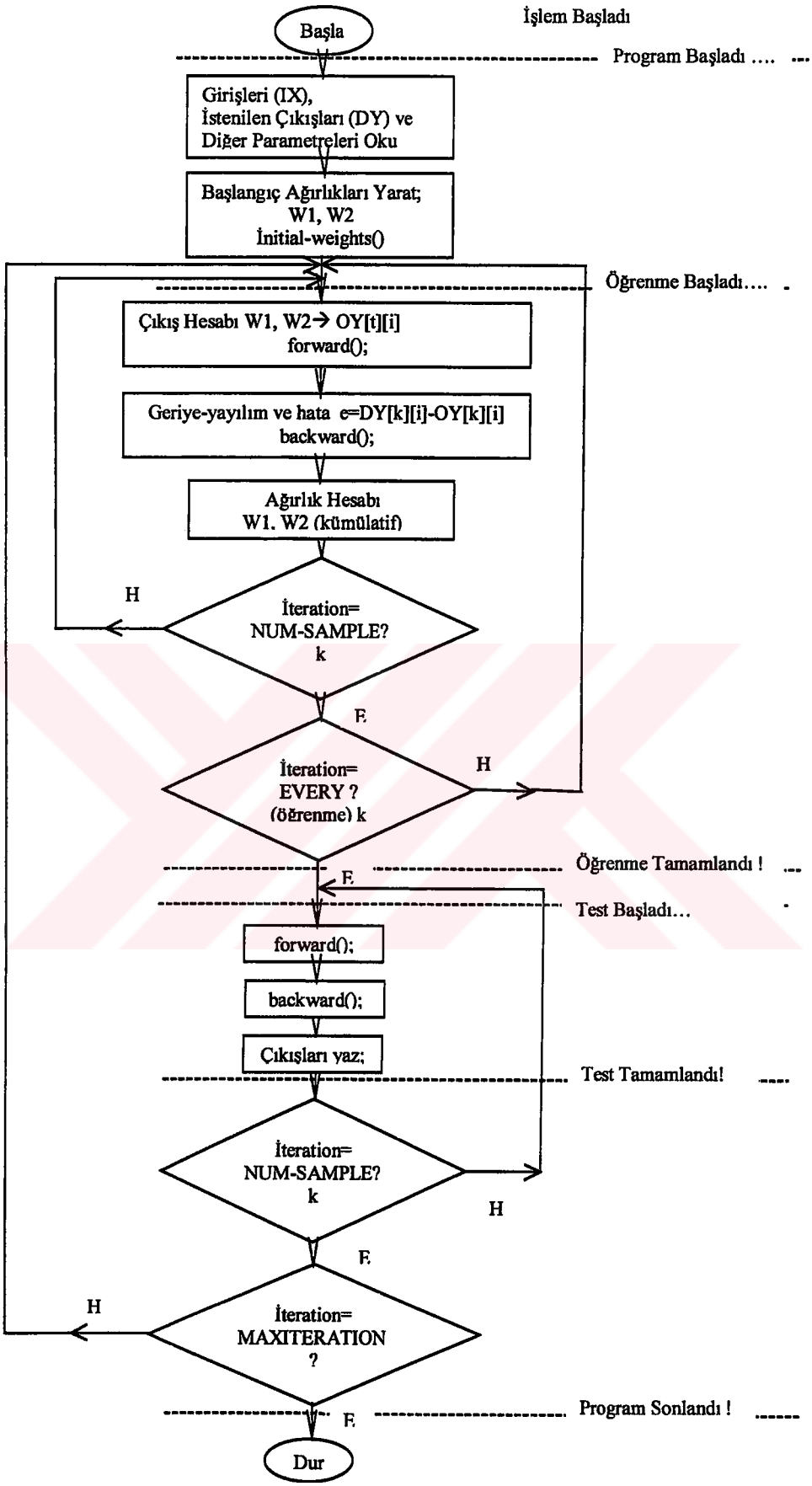
Şekil 5.1 Bir Küçük-İşaret Mikrodalga Transistoru Modeli



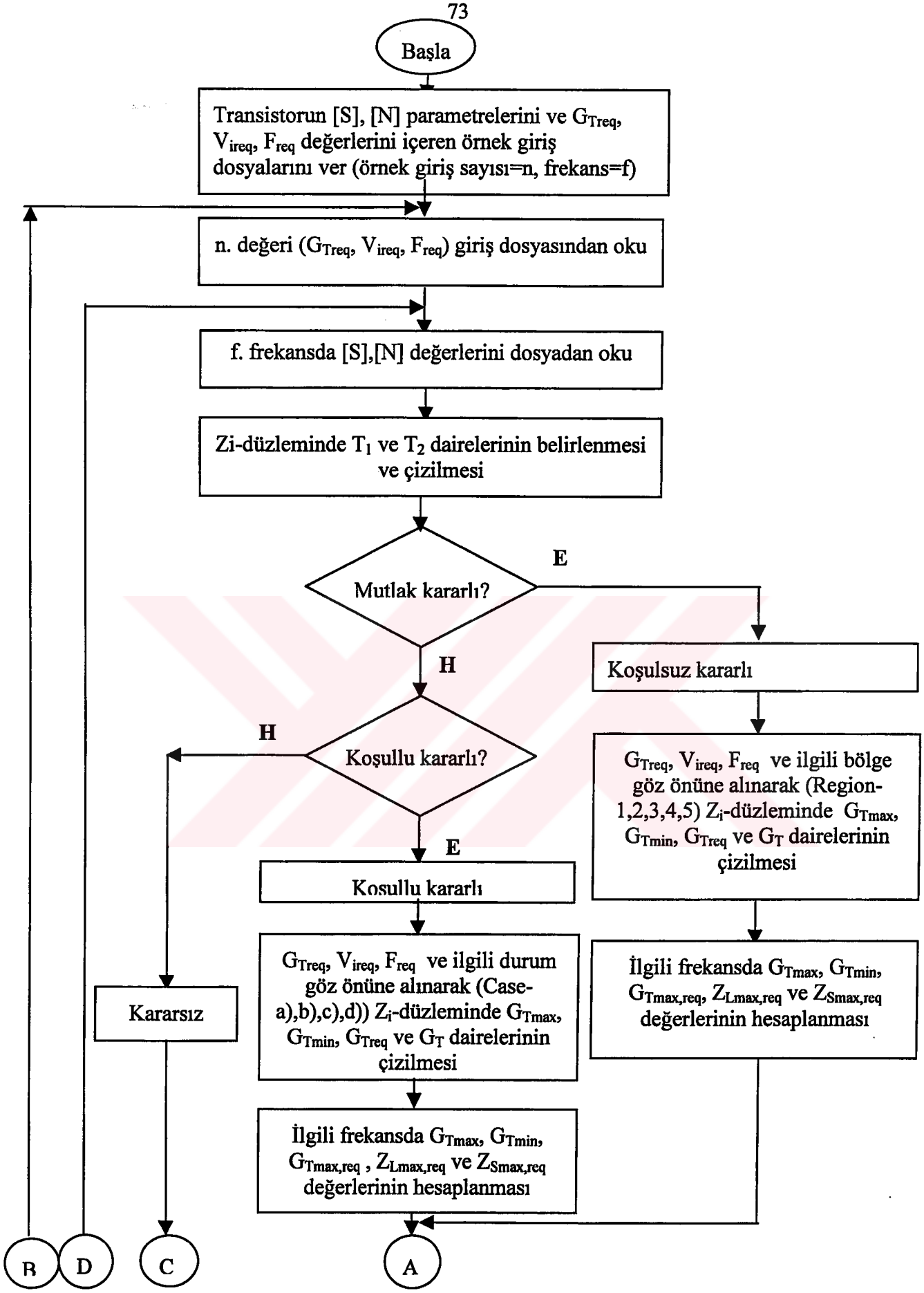
Şekil 5.2 Bir Uydurulmuş Transistorlu Kuvvetlendirici Blok Yapısı

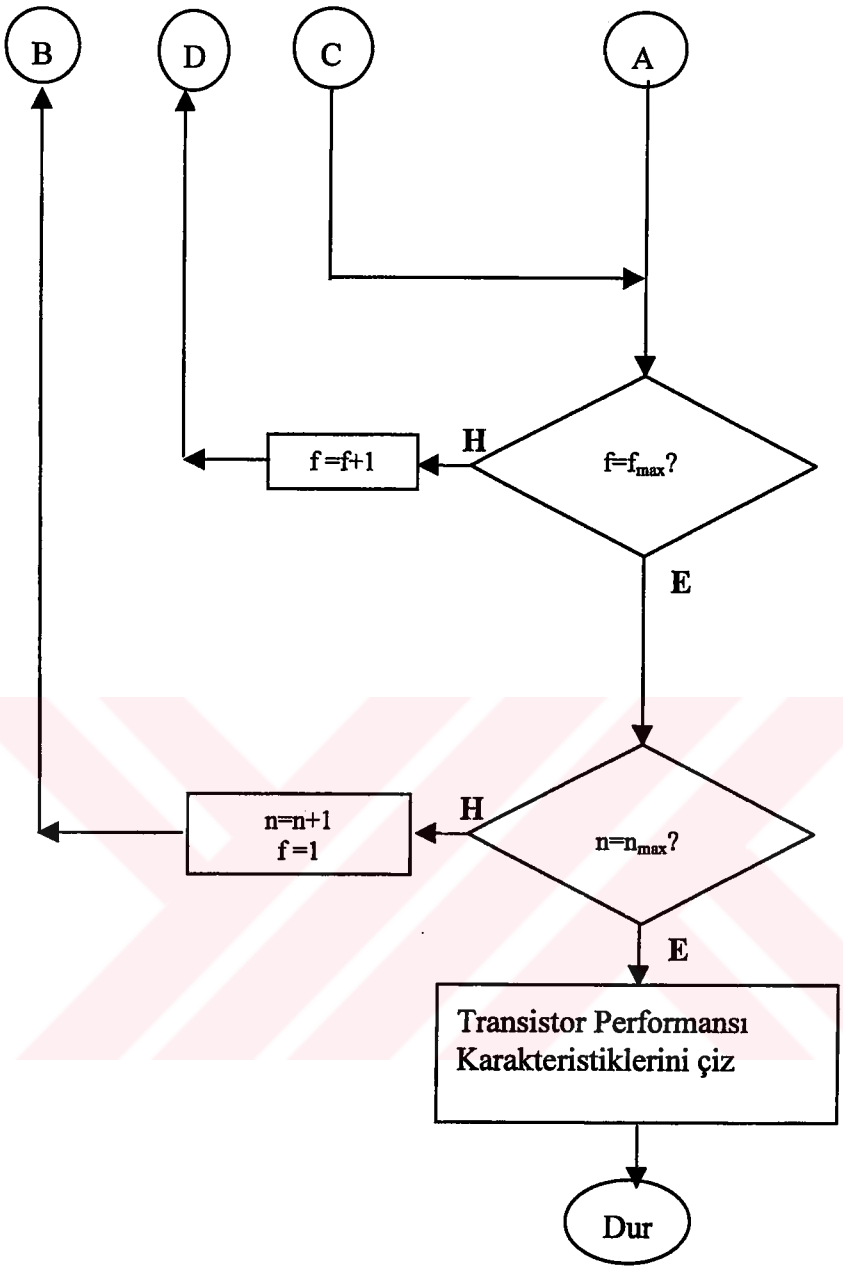


Şekil 5.3 Aktif Elemanın YSA ile Performans Analizini Yapan ve Uydurma Devresi Tasarlayan Sistemin Blok Yapısı

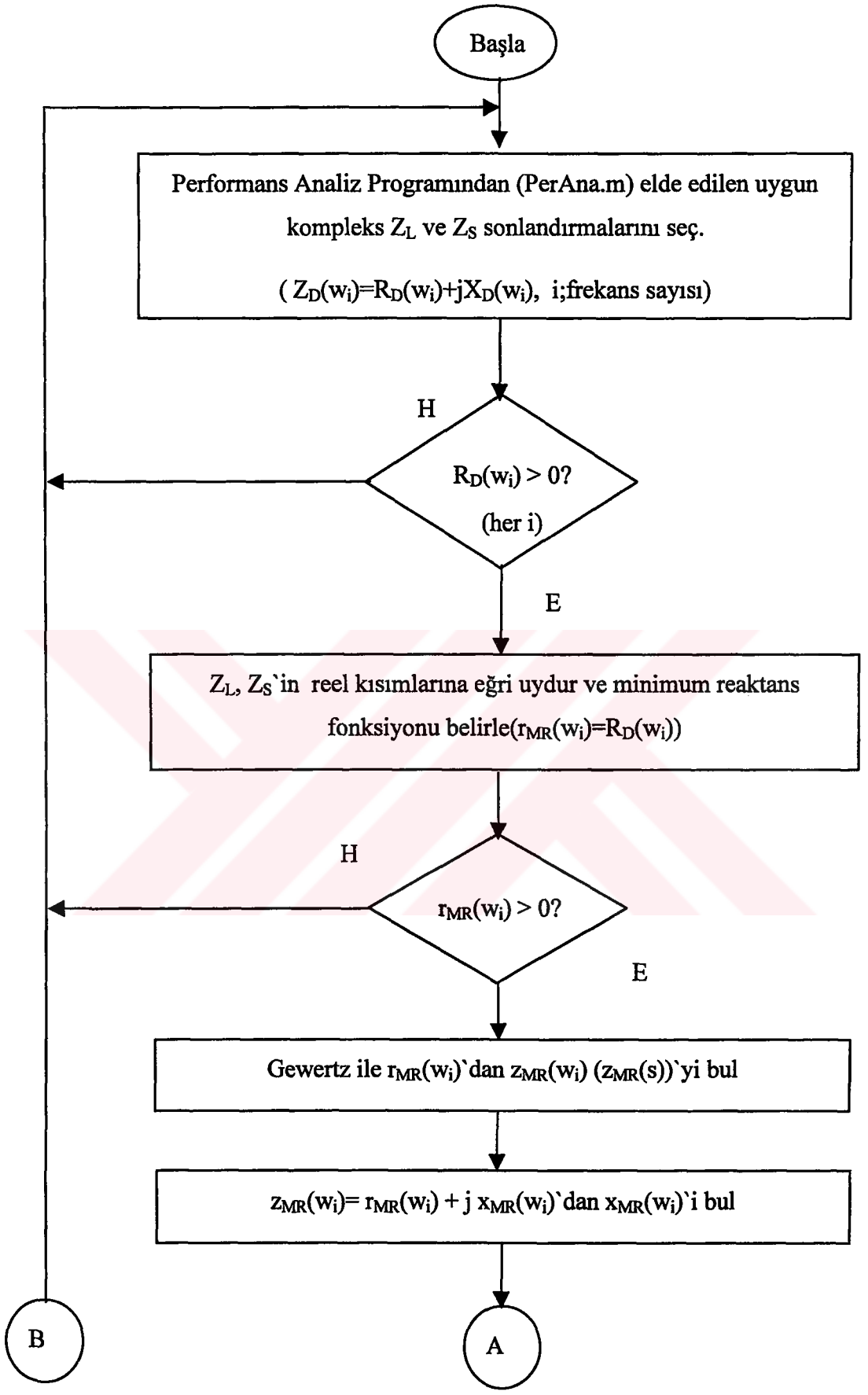


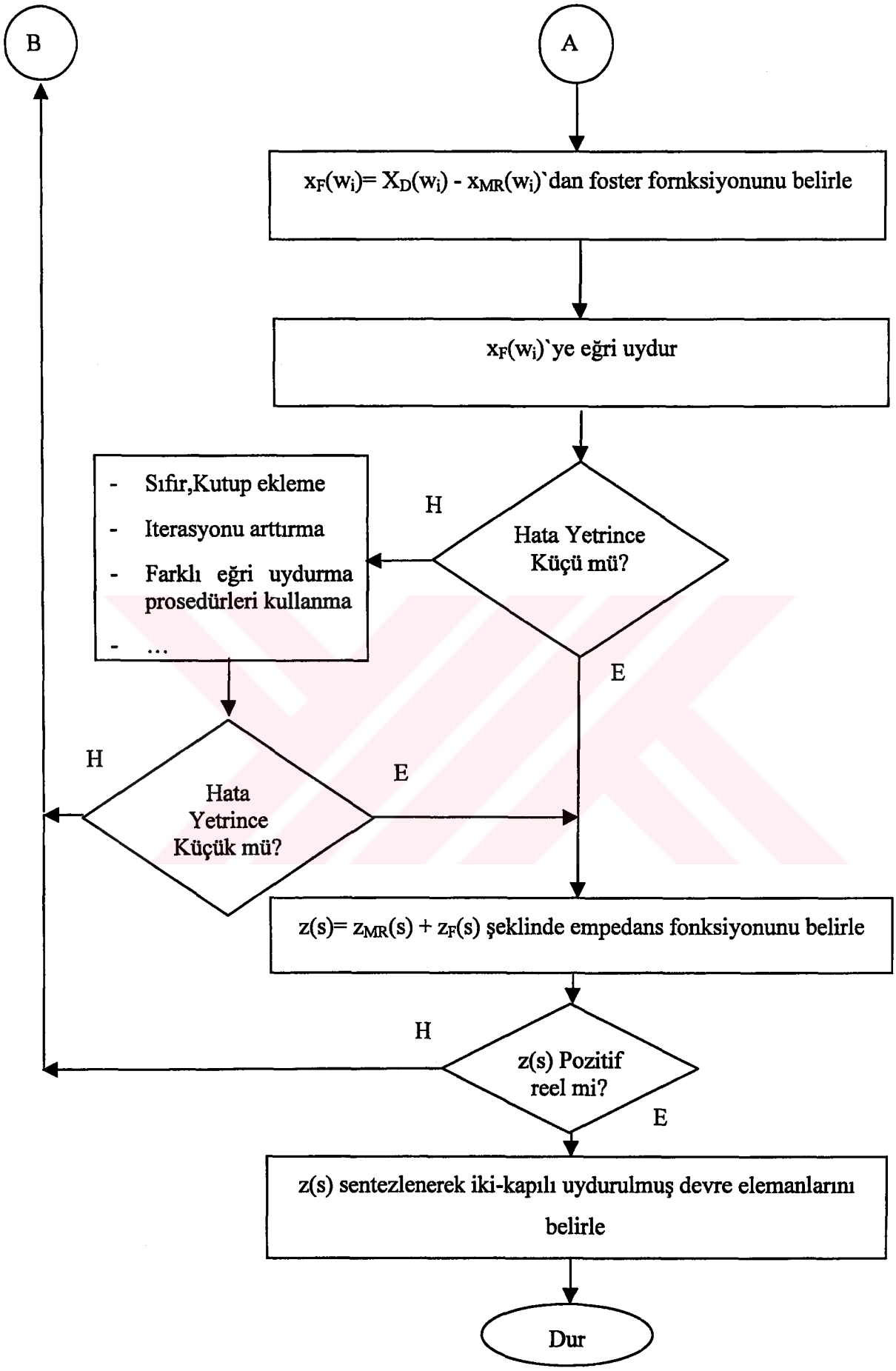
Şekil 5.4 Yapay Sinir Ağı Programı (nnpsn.pas) Akış Diyagramı


Başla
Transistorun [S], [N] parametrelerini ve G_{Treq}, V_{ireq}, F_{req} değerlerini içeren örnek giriş dosyalarını ver (örnek giriş sayısı=n, frekans=f)
n. değeri (G_{Treq}, V_{ireq}, F_{req}) giriş dosyasından oku
f. frekansda [S],[N] değerlerini dosyadan oku
Zi-düzleminde T_1 ve T_2 dairelerinin belirlenmesi ve çizilmesi
Mutlak kararlı?
E
H
Koşullu kararlı?
H
E
Koşulsuz kararlı
Kosullu kararlı
Kararsız
G_{Treq}, V_{ireq}, F_{req} ve ilgili durum göz önüne alınarak (Case-a),b),c),d)) Z_i-düzleminde G_{Tmax}, G_{Tmin}, G_{Treq} ve G_T dairelerinin çizilmesi
G_{Treq}, V_{ireq}, F_{req} ve ilgili bölge göz önüne alınarak (Region-1,2,3,4,5) Z_i-düzleminde G_{Tmax}, G_{Tmin}, G_{Treq} ve G_T dairelerinin çizilmesi
İlgili frekansda G_{Tmax}, G_{Tmin}, $G_{Tmax,req}$, $Z_{Lmax,req}$ ve $Z_{Smax,req}$ değerlerinin hesaplanması
İlgili frekansda G_{Tmax}, G_{Tmin}, $G_{Tmax,req}$, $Z_{Lmax,req}$ ve $Z_{Smax,req}$ değerlerinin hesaplanması
B
D
C
A



Şekil 5.5 Performans Analizi Programı (PerAna.m) Akış Diyagramı


Başla
Performans Analiz Programından (PerAna.m) elde edilen uygun kompleks Z_L ve Z_S sonlandırmalarını seç.
($Z_D(w_i)=R_D(w_i)+jX_D(w_i)$, i;frekans sayısı)
H
$R_D(w_i) > 0$?
(her i)
E
Z_L, Z_S`in reel kısımlarına eğri uydur ve minimum reaktans fonksiyonu belirle($r_{MR}(w_i)=R_D(w_i)$)
H
$r_{MR}(w_i) > 0$?
E
Gewertz ile $r_{MR}(w_i)$`dan $z_{MR}(w_i)$ ($z_{MR}(s)$)`yi bul
$z_{MR}(w_i)= r_{MR}(w_i) + j\, x_{MR}(w_i)$`dan $x_{MR}(w_i)$`i bul
B
A



Şekil 5.6 Uydurma Devresi Tasarım Programı Akış Diyagramı

5.2 Programlar

5.2.1 Yapay Sinir Ağı Programı (nnpsn.pas)

Yapay sinir ağı simülatörü olarak kullanılan Pascal programı (nnpsn.pas) Ek-1`de verilmiştir.

Program kullanılırken aşağıdaki konulara dikkat edilmesi gerekmektedir.

1. Normalizasyon dosyası: norm.dat satır bazında sıra ile kaç frekans bilgisi olduğu (Number_of_input_line), S21`in maksimum değeri (S21max), Fmin`nin maksimum değeri (Fminmax) ve Transistorun ad bilgilerini içeriyor olması gerekmektedir. Eğer bu değerler 1 den küçük ise norm.dat'a 1 yazılır, böylelikle o değer üzerinde normalizasyon yapılmamış olur.
2. Giriş dosyası: input.dat; format: f(GHz) VCE(V) IC(mA) CT değerlerini satır bazlı içeriyor olmalı. Burada konfigürasyon tipi CT=0.1 --> CE (common emiter), CT=0.9 --> CC (common collector) şeklinde olmalıdır.
3. Test giriş dosyası: input_t.dat; giriş dosyası formatında test verilerini içeriyor olmalı.
4. İstenilen değerler: desired.dat, format: |S11| <S11(R) |S21| <S21(R) |S12| <S12(R) |S22| <S22(R) Fmin(dB) |Gopt| <Gopt(R) Rn/50 şeklindedir ve bu değerler üretici verileri olmaktadır.
5. Yukarıdaki değerler girildikten sonra "nnpsn.exe" çalıştırılır.
6. Çıkış dosyası: output.txt; eğitme verileriyle yapılan test sonucunu ve hatayı içermektedir.
7. Test çıkış dosyası: output_t.txt; test verileriyle elde edilen sonuçlar içermektedir.

5.2.2 Performans Analizi Programı (PerAna.m)

Transistorun performans analizinde kullanılan matlab programı (PerAna.m) Ek-2`de verilmiştir.

Program kullanılırken aşağıdaki konulara dikkat edilmesi gerekmektedir.

1. Her bir transistor için aşağıdaki gibi bir ifade PerAna.m içine yazılması gerekmektedir.

```
fid_r_s=fopen('ne02135c.ins','r');
fid_r_n=fopen('ne02135c.inn','r');
fid_r_p=fopen('ne02135c.inp','r');
```

Buradaki *.ins dosyası transistorun frekansla birlikte S-parametrelerini satır bazlı içeren bir dosyadır. *.inn ise frekansla birlikte YSA`dan elde edilen gürültü parametrelilerini satır bazlı içermektedir. *.inp dosyası istenilen GTreq, Vireq, Freq değerlerini içeren giriş dosyasıdır. *.inp dosyasında verilen Freq değeri Fmin`den küçük olması durumunda mesaj verilerek sonraki giriş değerine geçilmektedir.

2. Transistorun analizine başlayabilmek için programda aşağıdaki parametreler uygun değerlere çekilmelidir.

```
Name_of_transistor='ne02135c(VCE=10V, IC=20mA) ';  % transistor adı
N_Num=7;      % hesaplamaya giren frekans sayısı
P_Num=14;     % *.inp`deki satır sayısı
```

3. Programda ihtiyaca göre değiştirilebilecek iki parametre mevcuttu.

```
step_for_sample_on_circle=0.01;    % Bölge 3`de hesaplama adımı
switch_for_plot=0;                 % 1 --> Daireleri çiz, 0 --> Daireleri çizme
```

4. Program matlab`den "run perana" şeklinde koşturulduktan sonra üç tane dosya oluşturmaktadır.

out_int.txt; Gtmax'a karşılık gelen ZLmax, ZSmax ile GTreq`a karşılık gelen ve T1 ile T2 daireleriyle kesişim noktalarındaki ZLreq, ZSreq değerlerini içeren dosya.

out_oth.txt; Bölge 3`de GTreq`a karşılık gelen ZLreq, ZSreq değerleri içeren dosya.

out_gra.txt; out_int.txt`daki bilgilerden GTmax, ZLmax, ZSmax değerleri ve GTreq`a karşılık gelen bazı ZLreq, ZSreq değerlerinin, değişim eğrileri elde edebilmek amacıyla sütun bazında gösterilimini içeren dosyadır.

6. UYGULAMALAR

6.1 Transistor: ne02135c(VCE=10V, IC=20mA)

6.1.1 YSA Sonuçları

6.1.1.1 Transistorun Gürültü ve S-parametrelerini İçeren Üretici Verileri

Aşağıdaki dosya transistorun, üreticiden alınan gürültü ve S-parametrelerini içermektedir.

Ne02135c.s2p:

```
! FILENAME:      NE02135C.S2P  VERSION: 1.0
! NEC PART NUMBER: NE02135      DATE:   4/85
! BIAS CONDITIONS: VCE=10V, IC=20mA
# ang in deg                          NOISE PARAMETERS (2/81)
#       S11        S21        S12        S22     Fmin.  âopt   RN/50
#GHZ  mag.  ang.   mag.  ang.   mag. ang. mag.   ang.  (dB)   mag.  ang.
Z----------------------------------------------------------------
 0.1  0.545  -74  32.448  140  0.002  67 0.763   -38
 0.2  0.584 -118  22.507  119  0.006  46 0.549   -53
 0.3  0.586 -140  16.328  106  0.013  47 0.395   -66
 0.4  0.597 -152  12.754   99  0.019  46 0.324   -70
 0.5  0.593 -163  10.200   94  0.020  46 0.270   -80  1.8 0.16  149  0.15
 0.6  0.604 -168   8.718   90  0.024  47 0.239   -75
 0.7  0.616 -175   7.505   85  0.024  49 0.224   -86
 0.8  0.604 -179   6.535   80  0.027  50 0.219   -86
 0.9  0.619  176   5.877   77  0.031  49 0.186   -92
 1.0  0.602  173   5.276   75  0.040  49 0.217   -89  1.9 0.33  169  0.13
 1.2  0.616  165   4.440   68  0.055  51 0.207   -90
 1.4  0.603  160   3.729   63  0.066  50 0.203   -91
 1.6  0.639  157   3.357   58  0.076  50 0.189   -90  2.4 0.46 -179  0.09
 1.8  0.626  151   2.990   52  0.091  49 0.172   -92
 2.0  0.616  148   2.718   46  0.108  44 0.161  -110  2.9 0.53 -167  0.08
 2.5  0.623  135   2.159   39  0.133  48 0.176  -120  3.2 0.57 -154  0.14
 3.0  0.639  123   1.841   29  0.156  43 0.188  -128  3.9 0.62 -139  0.27
```

```
3.5  0.644  111   1.549  13  0.180 32 0.210 -135  4.3 0.67 -134  0.42
4.0  0.649  102   1.411   6  0.205 32 0.232 -142
4.5  0.656   93   1.238   1  0.228 26 0.245 -151
5.0  0.674   85   1.135 -14  0.251 17 0.268 -161
```

Yukarıdaki dosyadan da görülebileceği gibi frekans sabit bir adımla artmakta ve bazı frekanslarda gürültü parametreleri verilmemektedir. Bölüm 2'de verilen YSA modeli kullanılarak verilmeyen frekanslardaki S ve gürültü parametreleri elde edilebilir. Aşağıda sonuçları verilen uygulama, YSA'nın üretici tarafından verilen S ve gürültü parametreleriyle eğitilmesi ve gürültü parametreleri verilmemiş frekanslardaki S ve gürültü parametrelerini kestirimine dayanmaktadır. Elde edilen sonuçlar aşağıda verilmiştir.

Test verileri eğitme verileriyle aynı olması durumunda YSA çıktısı (output.txt):

```
Input Values:
Num f(GHz)  VCE(V) IC(mA)  CT
--- ------  ------ ------  ---
 0   0.50  10.00  20.00   0.10
 1   1.00  10.00  20.00   0.10
 2   1.60  10.00  20.00   0.10
 3   2.00  10.00  20.00   0.10
 4   2.50  10.00  20.00   0.10
 5   3.00  10.00  20.00   0.10
 6   3.50  10.00  20.00   0.10
Calculated Values:
Num |S11|  <S11(D) |S21|  <S21(D) |S12|  <S12(D) |S22|  <S22(D) Fmin(dB) |Gopt| <Gopt(D) Rn/50
--- ------------------------------------------------------------------------------
 0 0.59165 -162.55 9.62732 94.7525 0.03004 45.9844 0.27261 -79.687 1.77817 0.15357 148.897 0.15521
 1 0.61103 172.498 5.49202 73.0486 0.05447 49.1096 0.21241 -86.078 1.96534 0.33750 167.697 0.11304
 2 0.62007 156.334 3.20668 57.8691 0.08081 49.0484 0.18233 -98.334 2.38981 0.46290 -175.99 0.09822
 3 0.62412 147.560 2.67159 49.0717 0.09897 47.8553 0.17279 -107.19 2.76895 0.51891 -166.33 0.10206
 4 0.62940 136.479 2.21957 37.6315 0.12307 44.8980 0.17363 -117.10 3.35300 0.57405 -155.10 0.13886
```

5 0.63627 123.090 1.83948 25.4733 0.15377 40.0674 0.18846 -127.35 3.90542 0.62492 -142.64 0.25155

6 0.64351 110.452 1.55541 16.9577 0.18583 35.2489 0.20887 -136.35 4.15310 0.66688 -131.36 0.42810

Desired Values:

Num |S11| <S11(D) |S21| <S21(D) |S12| <S12(D) |S22| <S22(D) Fmin(dB) |Gopt| <Gopt(D) Rn/50

--- ----------

0 0.59300 -163.00 10.2000 94.0000 0.02000 46.0000 0.27000 -80.000 1.80000 0.16000 149.000 0.15000

1 0.60200 173.000 5.27600 75.0000 0.04000 49.0000 0.21700 -89.000 1.90000 0.33000 169.000 0.13000

2 0.63900 157.000 3.35700 58.0000 0.07600 50.0000 0.18900 -90.000 2.40000 0.46000 -179.00 0.09000

3 0.61600 148.000 2.71800 46.0000 0.10800 44.0000 0.16100 -110.00 2.90000 0.53000 -167.00 0.08000

4 0.62300 135.000 2.15900 39.0000 0.13300 48.0000 0.17600 -120.00 3.20000 0.57000 -154.00 0.14000

5 0.63900 123.000 1.84100 29.0000 0.15600 43.0000 0.18800 -128.00 3.90000 0.62000 -139.00 0.27000

6 0.64400 111.000 1.54900 13.0000 0.18000 32.0000 0.21000 -135.00 4.30000 0.67000 -134.00 0.42000

iteration= 50000 nu = 0.011318 error_learning= 0.000512 error_test= 0.000504

Burada error_learning eğitim hatası, error_test ise test hatası olmaktadır. Bu hatalarla elde edilen YSA aşağıdaki durumdaki gibi farklı test verileriyle test edilirse yine aşağıda verilecek sonuçlar elde edilir.

Test verilerin eğitme verilerinden farklı olması durumunda YSA çıktısı (output_t.txt):

Input Values:

Num f(GHz) VCE(V) IC(mA) CT

--- ------ ------ ------ ---

0 0.60 10.00 20.00 0.10

1 0.70 10.00 20.00 0.10

2 0.80 10.00 20.00 0.10

3 0.90 10.00 20.00 0.10

4 1.20 10.00 20.00 0.10

5 1.40 10.00 20.00 0.10

6 1.80 10.00 20.00 0.10

Calculated Values:

Num |S11| <S11(D) |S21| <S21(D) |S12| <S12(D) |S22| <S22(D) Fmin(dB) |Gopt| <Gopt(D) Rn/50

--- ----------

0 0.59577 -167.61 9.22333 90.1744 0.03392 46.6852 0.25942 -80.654 1.80471 0.18381 152.521 0.14546

1 0.60021 -173.15 8.52093 85.3184 0.03871 47.4412 0.24554 -81.789 1.83638 0.22158 156.535 0.13546

2 0.60446 -178.57 7.52741 80.6728 0.04402 48.1467 0.23253 -83.077 1.87347 0.26293 160.568 0.12637

3 0.60811 176.588 6.44059 76.5605 0.04939 48.7148 0.22141 -84.507 1.91634 0.30273 164.325 0.11887

4 0.61509 166.130 4.25937 67.3415 0.06356 49.4394 0.19930 -89.670 2.08280 0.39054 173.539 0.10524

5 0.61783 161.016 3.61069 62.4670 0.07208 49.3627 0.18985 -93.827 2.22562 0.42973 178.848 0.10061

6 0.62214 151.865 2.90893 53.4241 0.08984 48.5511 0.17655 -102.87 2.57103 0.49256 -171.02 0.09837

Yukarıdaki veriler içinde S-parametreleri, üretici tarafından verilen orijinal değerlerle karşılaştırıldığında belli bir hata ile değerlerin doğru olarak bulunduğu görülecektir. Aynı şekilde gürültü parametrelerin de aykırı değerler olmadığı görülmektedir. Hata seviyesini düşürmek için daha çok eğitme verileri bulunmalı veya YSA'nın eğitme parametreleriyle oynanmalı. Bizim durumumuzda bu hata seviyesi kabul edilebilir olmaktadır.

6.1.1.2 Transistorun Gürültü ve S-parametrelerini İçeren YSA Verileri

Daha öncede belirtildiği gibi, şu andaki uygulama YSA'nın üretici tarafından verilen S ve gürültü parametreleriyle eğitilmesi ve gürültü parametreleri verilmemiş frekanslardaki S ve gürültü parametrelerini kestirimine dayanmaktadır. Aslında bu frekanslarda S-parametreleri üretici tarafından zaten verilmiş olduğundan, YSA'dan elde edilen ve belli bir hatayı içeren veriler yerine üretici verileri kullanılmıştır. Dolayısıyla bu özel durumda, YSA temelde gürültü parametrelerini kestirmek için kullanılmıştır. Bu durumda YSA tarafından yeni belirlenen gürültü parametreleri ile üretici tarafından verilen S-parametreleri birleştirilirse transistora ilişkin yeni S-parametreleri ve gürültü parametreleri dosyası aşağıdaki şekilde elde edilir.

Ne02135c.sn:

```
! FILENAME:      NE02135C.S2P  VERSION: 1.0
! NEC PART NUMBER: NE02135     DATE:  4/85
! BIAS CONDITIONS: VCE=10V, IC=20mA
# ang in deg                          NOISE PARAMETERS (2/81)
#      S11        S21       S12       S22     Fmin.  âopt   RN/50
#GHZ mag.  ang.   mag.  ang.  mag. ang. mag.  ang. (dB)  mag.  ang.
Z----------------------------------------------------------
 0.1  0.545  -74  32.448 140  0.002 67 0.763  -38
 0.2  0.584 -118  22.507 119  0.006 46 0.549  -53
```

0.3 0.586 -140 16.328 106 0.013 47 0.395 -66

0.4 0.597 -152 12.754 99 0.019 46 0.324 -70

0.5 0.593 -163 10.200 94 0.020 46 0.270 -80 1.8 0.16 149 0.15

0.6 0.604 -168 8.718 90 0.024 47 0.239 -75 1.8 0.18 152.5 0.14

0.7 0.616 -175 7.505 85 0.024 49 0.224 -86 1.83 0.22 156.5 0.13

0.8 0.604 -179 6.535 80 0.027 50 0.219 -86 1.87 0.26 160.5 0.12

0.9 0.619 176 5.877 77 0.031 49 0.186 -92 1.9 0.30 164.3 0.11

1.0 0.602 173 5.276 75 0.040 49 0.217 -89 1.9 0.33 169 0.13

1.2 0.616 165 4.440 68 0.055 51 0.207 -90 2.0 0.39 173.5 0.10

1.4 0.603 160 3.729 63 0.066 50 0.203 -91 2.2 0.42 178.8 0.10

1.6 0.639 157 3.357 58 0.076 50 0.189 -90 2.4 0.46 -179 0.09

1.8 0.626 151 2.990 52 0.091 49 0.172 -92 2.5 0.49 -171.0 0.09

2.0 0.616 148 2.718 46 0.108 44 0.161 -110 2.9 0.53 -167 0.08

2.5 0.623 135 2.159 39 0.133 48 0.176 -120 3.2 0.57 -154 0.14

3.0 0.639 123 1.841 29 0.156 43 0.188 -128 3.9 0.62 -139 0.27

3.5 0.644 111 1.549 13 0.180 32 0.210 -135 4.3 0.67 -134 0.42

4.0 0.649 102 1.411 6 0.205 32 0.232 -142

4.5 0.656 93 1.238 1 0.228 26 0.245 -151

5.0 0.674 85 1.135 -14 0.251 17 0.268 -161

6.1.2 Performans Analizi Sonuçları

Yukarıda belirlene YSA'dan elde edilen transistorun S-parametreleri ve gürültü parametreleri dosyası da göz önüne alınmak üzere, performans analiz programına aşağıdaki giriş dosyasının verilmesi durumunda yine aşağıda belirtilen sonuçlar alınacaktır.

6.1.2.1 Giriş Dosyası (ne02135c.inp):

G_{Treq} V_{ireq} F_{req}

5.0 1.5 4.5

6.1.2.2 Daire Kesişimlerini İçeren Çıktı (out_int.txt):

Transistor: ne02135c(VCE=10V, IC=20mA) from Neural Network

Detailed information,Terminations for GTmax and all of intersection points on T1/T2 for GTreq:

*** 1. Given Values:

Required GT(dB) value: 5.000000

Required Vi value: 1.500000

Required F(dB) value: 4.500000

Circuit Definition:

f(Hz): 500000000.000000

S11 Module and Angle(Rad): 0.593000 -2.844887

S21 Module and Angle(Rad): 10.200000 1.640609

S12 Module and Angle(Rad): 0.020000 0.802851

S22 Module and Angle(Rad): 0.270000 -1.396263

Fmin 1.513561, GamaOpt Module and Angle(Rad) 0.160000 2.600541, Rn 0.150000

Z11 Real and Imaginer parts: 0.187926 -0.002324

Z21 Real and Imaginer parts: 2.483199 11.990902

Z12 Real and Imaginer parts: 0.020731 0.012114

Z22 Real and Imaginer parts: 0.867459 -0.308711

Freq,Fmin,Ropt,Xopt,Rn,Rmod,N,Rcn,Xcn,rn,Rct1,Xct1,rt1,Rct2,Xct2,rt2: 2.818383 1.513561 0.749600 0.126789 0.150000 0.447214 2.513849 3.263449 0.126789 3.176193 8.446265 -0.126789 8.412936 1.344082 -0.126789 1.115642

Unconditional stability: R11>0 & R22>0 & ita>1

R11,R22,ita = 0.187926,0.867459,1.427868

GTDBreq, T1 and T2 circles:

RCGreq= 0.240667, XCGreq= -0.162942, rgreq= 0.167589

Rct1= 8.446265, Xct1= -0.126789, rt1= 8.412936

Rct2= 1.344082, Xct2= -0.126789, rt2= 1.115642

1. Intersection point on T1; RintT1= NaN: XintT1= NaN

2. Intersection point on T1; RintT1= NaN: XintT1= NaN

ZireqT11 Value: real and imaginer part = NaN NaN

ZireqT12 Value: real and imaginer part = NaN NaN

1. Intersection point on T2; RintT2= 0.236203: XintT2= 0.004588

2. Intersection point on T2; RintT2= 0.247178: XintT2= -0.330404

ZireqT21 Value: real and imaginer part = 11.810129 0.229378

ZireqT22 Value: real and imaginer part = 12.358916 -16.520184

GTreq cuts only T2

ZCGmax(0.172727 -0.162942),Zct1(8.446265 -0.126789), Zct2(1.344082 -0.126789)

rt1 8.412936, rt2 1.115642

(abs(ZCGmax-Zct1) < rt1) & (rt2 < abs(ZCGmax-Zct2)): ZCGmax in Region-3

Zi Value: real and imaginer part = 8.636330 -8.147080

ZL Value: real and imaginer part = 39.864893 52.505761

Zs1 Value: real and imaginer part = 4.494323 12.800920

Zs2 Value: real and imaginer part = 4.384786 3.698168

GTMAX Value (dB) = 24.946712

GTMIN Value (dB) = 0.000000

T2 Intersection:

Zireq1 Value: real and imaginer part = 11.810129 0.229378

Zireq2 Value: real and imaginer part = 12.358916 -16.520184

ZLreq1 Value: real and imaginer part = 11.315847 -281.002463

ZLreq2 Value: real and imaginer part = 0.253392 21.848611

ZSreq11 Value: real and imaginer part = 4.524513 -0.825067

ZSreq12 Value: real and imaginer part = 4.524510 -0.825067

ZSreq21 Value: real and imaginer part = 4.754790 17.490405

ZSreq22 Value: real and imaginer part = 4.754790 17.490405

Circuit Definition:

f(Hz): 600000000.000000

Bu sonuçlar her frekans için bu şekilde verilmektedir. Bu bilgilerin özeti aşağıda verilmiş olduğundan burada verilmeyecektir.

==========

6.1.2.3 Bölge-3'de Elde Edilen Çözümlerini İçeren Çıktı (out_oth.txt):

==========

Transistor: ne02135c(VCE=10V, IC=20mA) from Neural Network

Terminations that give GTreq in Region-3:

*** 1. Given Values:

Required GT(dB) value: 5.000000

Required Vi value: 1.500000

Required F(dB) value: 4.500000

f(Hz): 500000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T2

Zireq1 Value: real and imaginer part = 11.303269 0.200487

Zireq2 Value: real and imaginer part = 11.851252 -16.524533

ZLreq1 Value: real and imaginer part = 17.232973 -359.948409

ZLreq2 Value: real and imaginer part = 0.245285 23.197894

ZSreq11 Value: real and imaginer part = 4.639848 -3.036728

ZSreq12 Value: real and imaginer part = 4.435814 1.524986

ZSreq21 Value: real and imaginer part = 4.932436 19.778073

ZSreq22 Value: real and imaginer part = 4.604278 15.093510

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq29 Value: real and imaginer part = 3.922711 -6.041684

Zireq30 Value: real and imaginer part = 4.077921 -10.778869

ZLreq1 Value: real and imaginer part = 0.343434 95.383893

ZLreq2 Value: real and imaginer part = 0.163632 59.138632

ZSreq11 Value: real and imaginer part = 4.424719 1.905879

ZSreq12 Value: real and imaginer part = 4.409070 10.171933

ZSreq21 Value: real and imaginer part = 4.603512 15.079626

ZSreq22 Value: real and imaginer part = 4.362988 6.570466

f(Hz): 600000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T2

Zireq1 Value: real and imaginer part = 10.874021 2.411289

Zireq2 Value: real and imaginer part = 10.766863 -13.998788

ZLreq1 Value: real and imaginer part = 14.683831 -289.894010

ZLreq2 Value: real and imaginer part = 0.347628 24.665124

ZSreq11 Value: real and imaginer part = 4.517697 -5.364705

ZSreq12 Value: real and imaginer part = 4.248819 -0.891931

ZSreq21 Value: real and imaginer part = 4.460808 16.873256

ZSreq22 Value: real and imaginer part = 4.215873 12.425178

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq27 Value: real and imaginer part = 3.768701 -3.321330

Zireq28 Value: real and imaginer part = 3.737014 -8.173866

ZLreq1 Value: real and imaginer part = 0.639667 111.512640

ZLreq2 Value: real and imaginer part = 0.251967 63.809163

ZSreq11 Value: real and imaginer part = 4.237869 -0.647441

ZSreq12 Value: real and imaginer part = 4.092250 7.235124

ZSreq21 Value: real and imaginer part = 4.203378 12.109674

ZSreq22 Value: real and imaginer part = 4.099770 4.276520

f(Hz): 700000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T2

Zireq1 Value: real and imaginer part = 10.437313 3.850967

Zireq2 Value: real and imaginer part = 9.576921 -9.727156

ZLreq1 Value: real and imaginer part = 18.569275 -273.027554

ZLreq2 Value: real and imaginer part = 0.422727 28.045676

ZSreq11 Value: real and imaginer part = 4.401286 -6.933821

ZSreq12 Value: real and imaginer part = 4.060720 -2.538688

ZSreq21 Value: real and imaginer part = 3.929118 12.121140

ZSreq22 Value: real and imaginer part = 3.809443 7.933020

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq21 Value: real and imaginer part = 4.663372 -1.107159

Zireq22 Value: real and imaginer part = 4.475003 -4.079870

ZLreq1 Value: real and imaginer part = 1.040759 105.949700

ZLreq2 Value: real and imaginer part = 0.480272 69.484984

ZSreq11 Value: real and imaginer part = 4.105203 -3.232488

ZSreq12 Value: real and imaginer part = 3.799012 5.226514

ZSreq21 Value: real and imaginer part = 3.930947 -0.078856

ZSreq22 Value: real and imaginer part = 3.812658 8.156996

f(Hz): 800000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T2

Zireq1 Value: real and imaginer part = 10.081365 5.302059

Zireq2 Value: real and imaginer part = 8.768130 -7.422707

ZLreq1 Value: real and imaginer part = 71.518238 -488.483317

ZLreq2 Value: real and imaginer part = 0.557472 32.761200

ZSreq11 Value: real and imaginer part = 4.320845 -8.523303

ZSreq12 Value: real and imaginer part = 3.906035 -4.186952

ZSreq21 Value: real and imaginer part = 3.589532 9.580391

ZSreq22 Value: real and imaginer part = 3.525364 5.572234

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq17 Value: real and imaginer part = 5.028551 -0.199243

Zireq18 Value: real and imaginer part = 4.945351 -1.005416

ZLreq1 Value: real and imaginer part = 1.163111 93.156706

ZLreq2 Value: real and imaginer part = 0.931171 82.802971

ZSreq11 Value: real and imaginer part = 3.897813 -4.081152

ZSreq12 Value: real and imaginer part = 3.533537 4.140499

ZSreq21 Value: real and imaginer part = 3.833117 -3.203984

ZSreq22 Value: real and imaginer part = 3.526988 4.933576

f(Hz): 900000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T2

Zireq1 Value: real and imaginer part = 9.980402 7.470155

Zireq2 Value: real and imaginer part = 8.025518 -5.098120

ZLreq1 Value: real and imaginer part = 35.962474 -303.306602

ZLreq2 Value: real and imaginer part = 0.682865 32.624625

ZSreq11 Value: real and imaginer part = 4.374470 -10.967654

ZSreq12 Value: real and imaginer part = 3.844066 -6.627122

ZSreq21 Value: real and imaginer part = 3.277045 7.035124

ZSreq22 Value: real and imaginer part = 3.268552 3.199936

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq15 Value: real and imaginer part = 5.459698 4.091406

Zireq16 Value: real and imaginer part = 4.744627 -0.505902

ZLreq1 Value: real and imaginer part = 4.285836 150.831008

ZLreq2 Value: real and imaginer part = 0.867174 66.379289

ZSreq11 Value: real and imaginer part = 4.062643 -8.589077

ZSreq12 Value: real and imaginer part = 3.376488 -0.337096

ZSreq21 Value: real and imaginer part = 3.562271 -3.431570

ZSreq22 Value: real and imaginer part = 3.258517 4.146221

f(Hz): 1000000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T2

Zireq1 Value: real and imaginer part = 12.371589 9.810759

Zireq2 Value: real and imaginer part = 9.194783 -5.778818

ZLreq1 Value: real and imaginer part = 15.257959 -151.099397

ZLreq2 Value: real and imaginer part = 0.859162 28.545905

ZSreq11 Value: real and imaginer part = 5.676070 -14.849809

ZSreq12 Value: real and imaginer part = 4.725530 -9.820321

ZSreq21 Value: real and imaginer part = 3.794205 8.170606

ZSreq22 Value: real and imaginer part = 3.672353 3.970712

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq23 Value: real and imaginer part = 4.887559 5.730953

Zireq24 Value: real and imaginer part = 3.903978 0.904214

ZLreq1 Value: real and imaginer part = 4.352876 146.790592

ZLreq2 Value: real and imaginer part = 0.932787 71.044350

ZSreq11 Value: real and imaginer part = 4.852754 -10.600954

ZSreq12 Value: real and imaginer part = 3.824707 -1.540031

ZSreq21 Value: real and imaginer part = 4.091842 -4.896602

ZSreq22 Value: real and imaginer part = 3.674404 2.876274

f(Hz): 1200000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T2

Zireq1 Value: real and imaginer part = 12.709923 14.027418

Zireq2 Value: real and imaginer part = 7.344079 -2.953741

ZLreq1 Value: real and imaginer part = 14.668979 -118.439079

ZLreq2 Value: real and imaginer part = 1.095279 31.841172

ZSreq11 Value: real and imaginer part = 6.217754 -20.100220

ZSreq12 Value: real and imaginer part = 4.883114 -14.924675

ZSreq21 Value: real and imaginer part = 3.035525 4.884931

ZSreq22 Value: real and imaginer part = 3.001018 1.171176

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq23 Value: real and imaginer part = 4.536345 10.775331

Zireq24 Value: real and imaginer part = 2.523217 4.404434

ZLreq1 Value: real and imaginer part = 14.962889 232.084551

ZLreq2 Value: real and imaginer part = 1.011825 75.775368

ZSreq11 Value: real and imaginer part = 5.018330 -15.522180

ZSreq12 Value: real and imaginer part = 3.527682 -6.904236

ZSreq21 Value: real and imaginer part = 3.564785 -7.216882

ZSreq22 Value: real and imaginer part = 3.092027 -1.669784

f(Hz): 1400000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T2

Zireq1 Value: real and imaginer part = 14.752215 15.830893

Zireq2 Value: real and imaginer part = 7.623846 -0.744264

ZLreq1 Value: real and imaginer part = 16.968820 -95.618401

ZLreq2 Value: real and imaginer part = 1.640132 31.753639

ZSreq11 Value: real and imaginer part = 7.856092 -24.097588

ZSreq12 Value: real and imaginer part = 5.810177 -18.229419

ZSreq21 Value: real and imaginer part = 3.141540 2.708837

ZSreq22 Value: real and imaginer part = 3.120917 -1.131108

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq25 Value: real and imaginer part = 5.011189 12.948106

Zireq26 Value: real and imaginer part = 3.016229 8.309350

ZLreq1 Value: real and imaginer part = 19.993974 215.431475

ZLreq2 Value: real and imaginer part = 2.305604 90.632452

ZSreq11 Value: real and imaginer part = 5.828249 -18.290284

ZSreq12 Value: real and imaginer part = 3.754369 -8.796689

ZSreq21 Value: real and imaginer part = 4.227221 -11.668482

ZSreq22 Value: real and imaginer part = 3.340345 -5.151788

f(Hz): 1600000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T2

Zireq1 Value: real and imaginer part = 15.678713 16.109208

Zireq2 Value: real and imaginer part = 7.206414 0.569972

ZLreq1 Value: real and imaginer part = 13.702855 -67.608921

ZLreq2 Value: real and imaginer part = 1.960085 28.100672

ZSreq11 Value: real and imaginer part = 9.142268 -26.139855

ZSreq12 Value: real and imaginer part = 6.386978 -19.824483

ZSreq21 Value: real and imaginer part = 2.982281 -2.480012

ZSreq22 Value: real and imaginer part = 2.967384 1.277952

Zireq3 Value: real and imaginer part = 15.223037 16.406254 (Uydurma Devresi İçin Seçildi)

Zireq4 Value: real and imaginer part = 6.709884 0.792087

ZLreq1 Value: real and imaginer part = 14.806137 -73.103320

ZLreq2 Value: real and imaginer part = 1.878069 29.775767

ZSreq11 Value: real and imaginer part = 9.741052 -27.279903

ZSreq12 Value: real and imaginer part = 5.957600 -18.607263

ZSreq21 Value: real and imaginer part = 3.022805 -3.378226

ZSreq22 Value: real and imaginer part = 2.980402 1.675525

Zireq5 Value: real and imaginer part = 14.738918 16.680798

Zireq6 Value: real and imaginer part = 6.216861 1.050301

ZLreq1 Value: real and imaginer part = 16.125423 -79.381602

ZLreq2 Value: real and imaginer part = 1.796117 31.542390

ZSreq11 Value: real and imaginer part = 10.142307 -28.010308

ZSreq12 Value: real and imaginer part = 5.662197 -17.714135

ZSreq21 Value: real and imaginer part = 3.063594 -4.067394

ZSreq22 Value: real and imaginer part = 2.984970 1.797860

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq31 Value: real and imaginer part = 3.987355 14.773027

Zireq32 Value: real and imaginer part = 1.996293 11.121174

ZLreq1 Value: real and imaginer part = 24.677405 238.015757

ZLreq2 Value: real and imaginer part = 2.638114 106.933499

ZSreq11 Value: real and imaginer part = 6.172289 -19.226922

ZSreq12 Value: real and imaginer part = 3.914907 -10.822727

ZSreq21 Value: real and imaginer part = 4.344769 -12.898275

ZSreq22 Value: real and imaginer part = 3.599375 -8.972795

f(Hz): 1800000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T2

Zireq1 Value: real and imaginer part = 17.365485 18.671711

Zireq2 Value: real and imaginer part = 7.051181 2.985286

ZLreq1 Value: real and imaginer part = 16.931284 -62.942039

ZLreq2 Value: real and imaginer part = 2.544557 28.497602

ZSreq11 Value: real and imaginer part = 11.535824 -31.569006

ZSreq12 Value: real and imaginer part = 7.536243 -24.524618

ZSreq21 Value: real and imaginer part = 2.941634 -1.022221

ZSreq22 Value: real and imaginer part = 2.898470 -4.772079

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq31 Value: real and imaginer part = 4.860741 18.484076

Zireq32 Value: real and imaginer part = 2.267693 14.540461

ZLreq1 Value: real and imaginer part = 62.474190 302.225720

ZLreq2 Value: real and imaginer part = 4.389543 115.361267

ZSreq11 Value: real and imaginer part = 7.259334 -23.918457

ZSreq12 Value: real and imaginer part = 4.033184 -14.134431

ZSreq21 Value: real and imaginer part = 4.674564 -16.733051

ZSreq22 Value: real and imaginer part = 3.617649 -12.014330

f(Hz): 2000000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T2

Zireq1 Value: real and imaginer part = 19.823313 18.511094 (Uydurma Devresi İçin Seçildi)

Zireq2 Value: real and imaginer part = 6.881212 3.774315

ZLreq1 Value: real and imaginer part = 19.093266 -53.228242

ZLreq2 Value: real and imaginer part = 2.822163 23.548676

ZSreq11 Value: real and imaginer part = 17.417389 -36.936128

ZSreq12 Value: real and imaginer part = 10.075717 -28.744297

ZSreq21 Value: real and imaginer part = 2.943717 -1.594298

ZSreq22 Value: real and imaginer part = 2.792628 -5.355500

Zireq3 Value: real and imaginer part = 19.444328 18.996514

Zireq4 Value: real and imaginer part = 6.350915 4.087442

ZLreq1 Value: real and imaginer part = 20.382765 -56.951030

ZLreq2 Value: real and imaginer part = 2.679295 25.173342

ZSreq11 Value: real and imaginer part = 18.943774 -38.182405

ZSreq12 Value: real and imaginer part = 8.965654 -27.029478

ZSreq21 Value: real and imaginer part = 2.962642 -1.379104

ZSreq22 Value: real and imaginer part = 2.810511 -6.392840

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq31 Value: real and imaginer part = 7.380471 21.633623

Zireq32 Value: real and imaginer part = 2.178014 15.709743

ZLreq1 Value: real and imaginer part = 335.263331 384.851102

ZLreq2 Value: real and imaginer part = 3.724368 91.941990

ZSreq11 Value: real and imaginer part = 10.881107 -29.883062

ZSreq12 Value: real and imaginer part = 4.427859 -16.738547

ZSreq21 Value: real and imaginer part = 4.708681 -17.672216

ZSreq22 Value: real and imaginer part = 3.592149 -13.296452

f(Hz): 2500000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T2

Zireq1 Value: real and imaginer part = 20.464231 15.910782 (Uydurma Devresi İçin Seçildi)

Zireq2 Value: real and imaginer part = 8.845541 9.419066

ZLreq1 Value: real and imaginer part = 14.612598 -12.716895

ZLreq2 Value: real and imaginer part = 5.606092 22.931710

ZSreq11 Value: real and imaginer part = 41.901520 -35.858840

ZSreq12 Value: real and imaginer part = 17.235650 -34.411879

ZSreq21 Value: real and imaginer part = 3.864128 -6.358702

ZSreq22 Value: real and imaginer part = 3.534482 -11.167511

Zireq3 Value: real and imaginer part = 20.585663 16.464484

Zireq4 Value: real and imaginer part = 8.310352 9.605894

ZLreq1 Value: real and imaginer part = 15.188629 -14.305379

ZLreq2 Value: real and imaginer part = 5.386706 24.407556

ZSreq11 Value: real and imaginer part = 46.664783 -33.212766

ZSreq12 Value: real and imaginer part = 14.375891 -32.507426

ZSreq21 Value: real and imaginer part = 3.937047 -5.919101

ZSreq22 Value: real and imaginer part = 3.581418 -12.326194

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq43 Value: real and imaginer part = 9.728759 27.104552

Zireq44 Value: real and imaginer part = 4.940187 24.429031

ZLreq1 Value: real and imaginer part = 290.598385 -114.500173

ZLreq2 Value: real and imaginer part = 46.347621 172.804562

ZSreq11 Value: real and imaginer part = 20.699981 -36.105518

ZSreq12 Value: real and imaginer part = 5.571101 -21.028880

ZSreq21 Value: real and imaginer part = 10.546537 -28.975536

ZSreq22 Value: real and imaginer part = 4.987046 -19.465691

f(Hz): 3000000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T1

Zireq1 Value: real and imaginer part = 4.054197 29.244129

Zireq2 Value: real and imaginer part = 2.664493 21.833889 (Uydurma Devresi İçin Seçildi)

ZLreq1 Value: real and imaginer part = 21.798685 112.494899

ZLreq2 Value: real and imaginer part = 4.012863 50.960668

ZSreq11 Value: real and imaginer part = 10.606151 -28.977043

ZSreq12 Value: real and imaginer part = 7.046540 -24.815366

ZSreq21 Value: real and imaginer part = 6.413319 -23.575885

ZSreq22 Value: real and imaginer part = 5.194411 -19.106255

Zireq3 Value: real and imaginer part = 4.836531 30.138633

Zireq4 Value: real and imaginer part = 3.069558 20.716703

ZLreq1 Value: real and imaginer part = 35.450493 130.093150

ZLreq2 Value: real and imaginer part = 4.124636 45.914834

ZSreq11 Value: real and imaginer part = 12.657913 -30.354006

ZSreq12 Value: real and imaginer part = 7.002089 -24.737135

ZSreq21 Value: real and imaginer part = 6.433563 -23.620436

ZSreq22 Value: real and imaginer part = 5.111028 -17.322443

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq65 Value: real and imaginer part = 19.228489 24.033982

Zireq66 Value: real and imaginer part = 18.695481 21.191850

ZLreq1 Value: real and imaginer part = 26.118696 -16.880399

ZLreq2 Value: real and imaginer part = 20.331545 -8.127872

ZSreq11 Value: real and imaginer part = 23.238534 -3.279189

ZSreq12 Value: real and imaginer part = 7.386817 -25.380134

ZSreq21 Value: real and imaginer part = 17.737489 -3.006880

ZSreq22 Value: real and imaginer part = 7.651519 -25.783118

f(Hz): 3500000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts T1 and T2

Zireq1 Value: real and imaginer part = 10.602556 37.612174

Zireq2 Value: real and imaginer part = 4.045318 26.861353 (Uydurma Devresi İçin Seçildi)

ZLreq1 Value: real and imaginer part = 86.844260 73.189873

ZLreq2 Value: real and imaginer part = 5.385347 33.694811

ZSreq11 Value: real and imaginer part = 16.435618 -25.770091

ZSreq12 Value: real and imaginer part = 11.824104 -26.482325

ZSreq21 Value: real and imaginer part = 10.514578 -26.034618

ZSreq22 Value: real and imaginer part = 7.244273 -22.494194

Zireq3 Value: real and imaginer part = 11.202612 37.487154

Zireq4 Value: real and imaginer part = 4.431126 26.385066

ZLreq1 Value: real and imaginer part = 92.986362 64.061005

ZLreq2 Value: real and imaginer part = 5.753853 31.979499

ZSreq11 Value: real and imaginer part = 17.511741 -24.982270

ZSreq12 Value: real and imaginer part = 11.172945 -26.299425

ZSreq21 Value: real and imaginer part = 11.600654 -26.427916

ZSreq22 Value: real and imaginer part = 6.949247 -21.440164

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq17 Value: real and imaginer part = 14.402578 35.629264

Zireq18 Value: real and imaginer part = 7.547640 24.390352

ZLreq1 Value: real and imaginer part = 83.737559 8.869630

ZLreq2 Value: real and imaginer part = 9.097470 22.472382

ZSreq11 Value: real and imaginer part = 19.579485 -19.654449

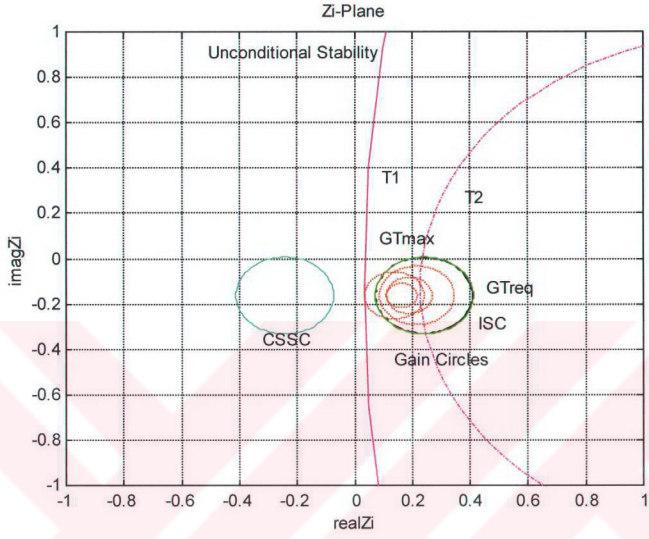
ZSreq12 Value: real and imaginer part = 9.275188 -25.270862

ZSreq21 Value: real and imaginer part = 19.387990 -21.912515

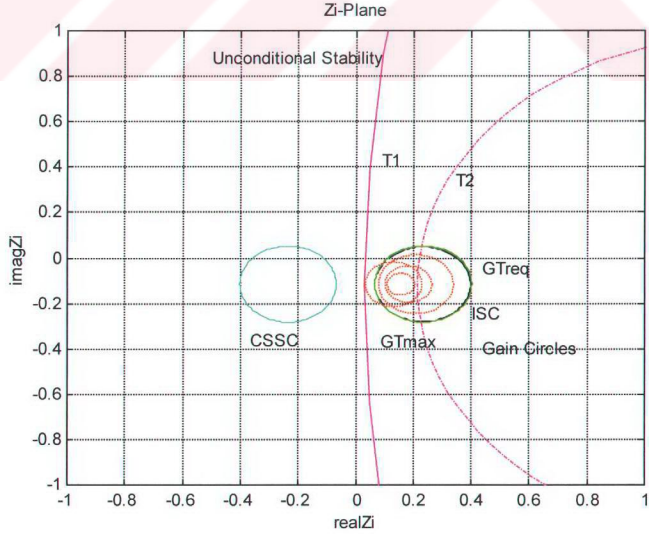
ZSreq22 Value: real and imaginer part = 7.947256 -16.655797

6.1.2.4 Elde Edilen Daireler

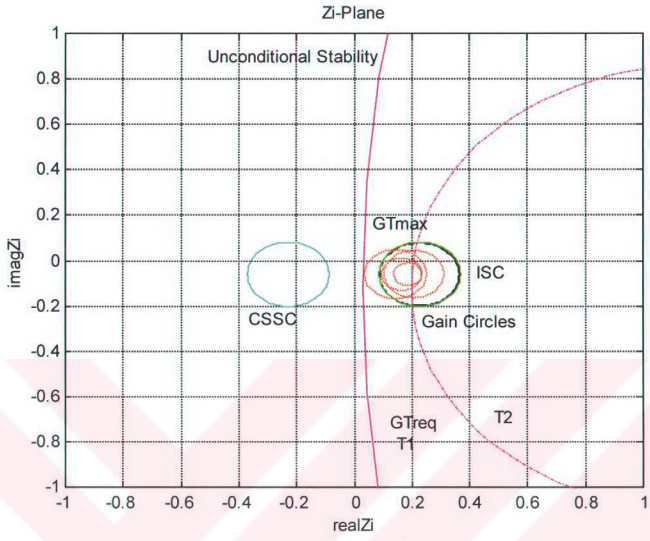
f=0.5GHz: G_{Tmax} G_{Treq} CSSC ISC Gain Circles T1 T2



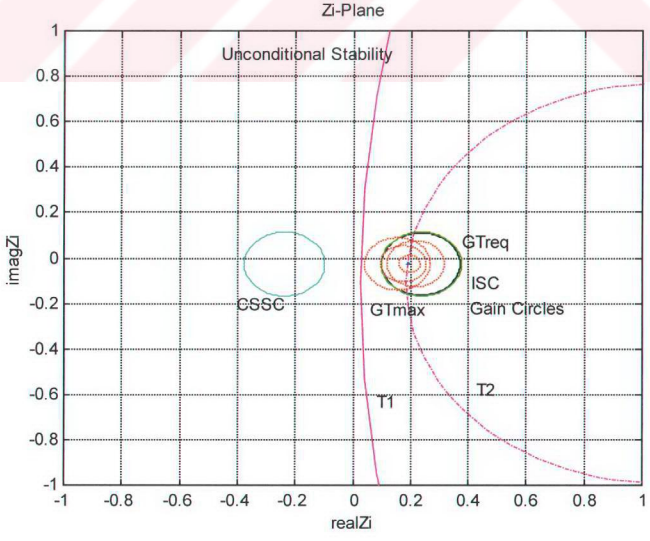
f=0.6GHz:



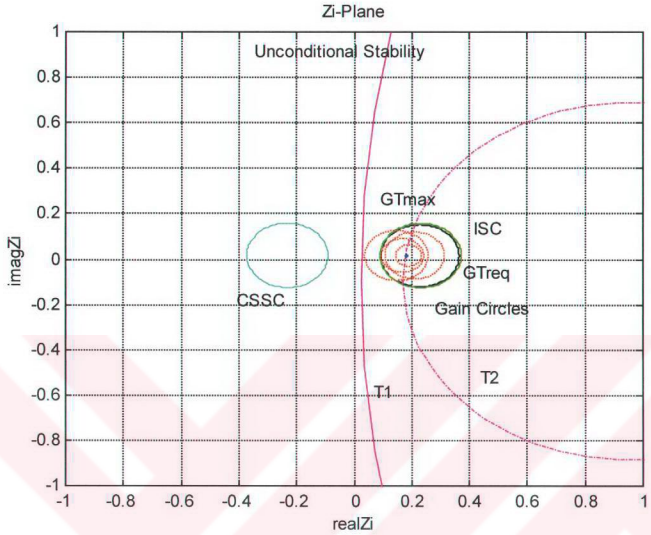
f=0.7GHz:



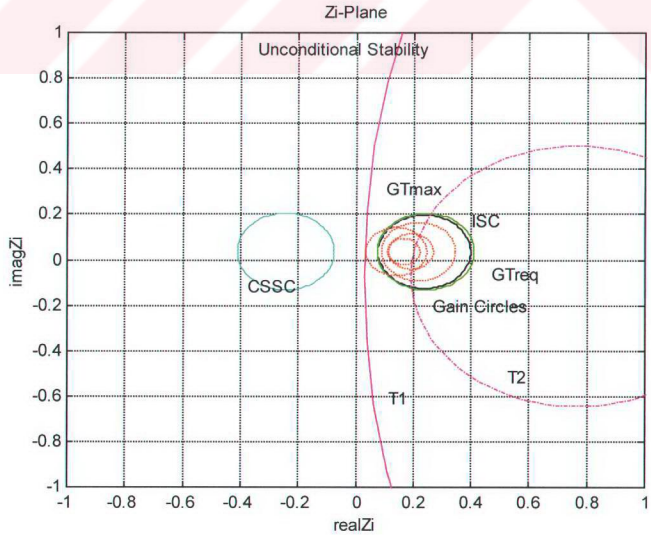
f=0.8GHz:



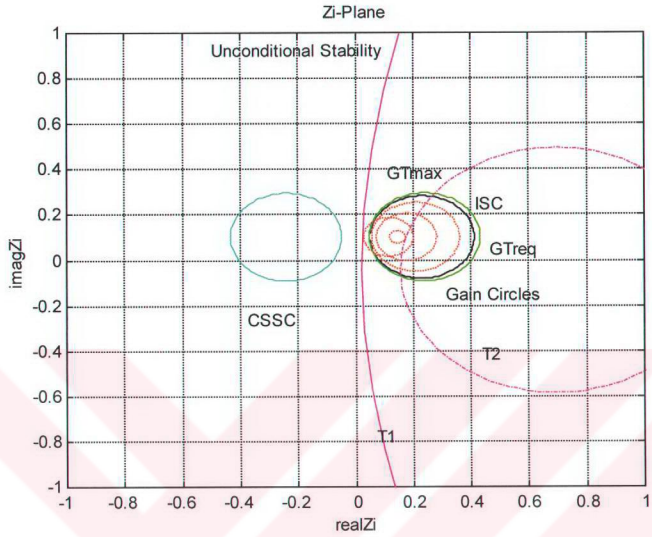
f=0.9GHz:



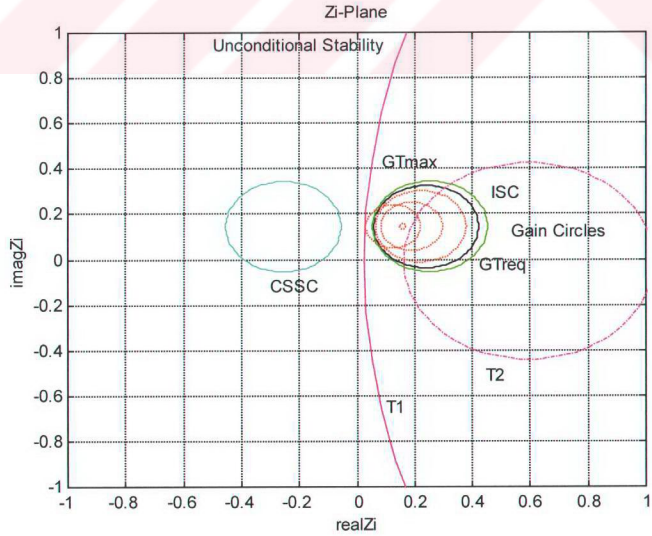
f=1.0GHz:



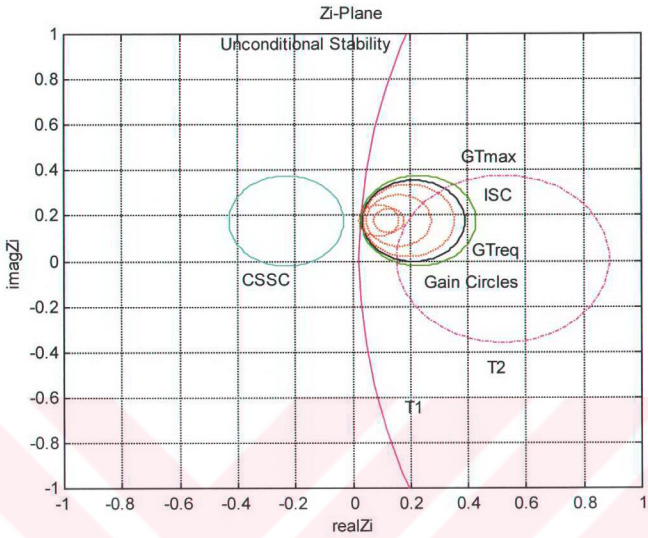
f=1.2GHz:



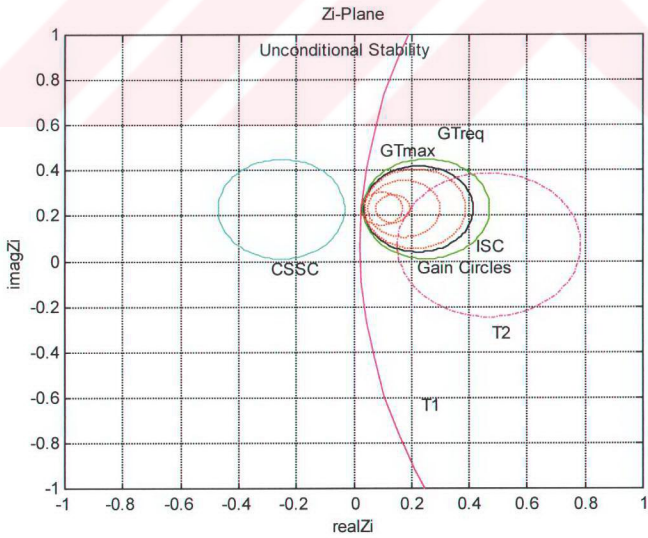
f=1.4GHz:



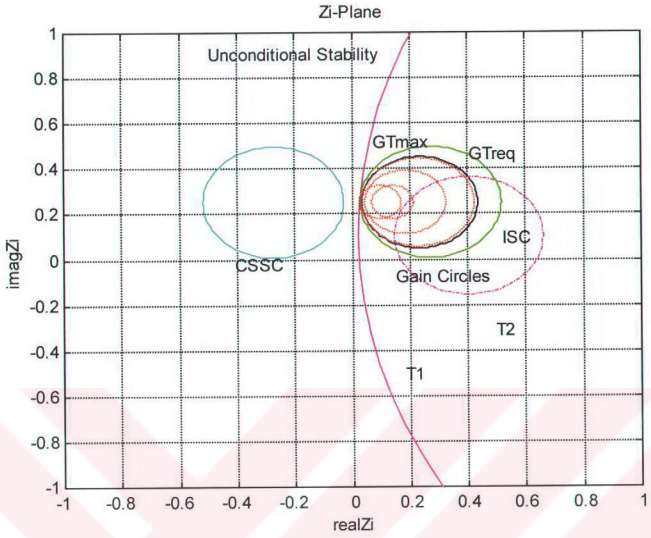
f=1.6GHz:



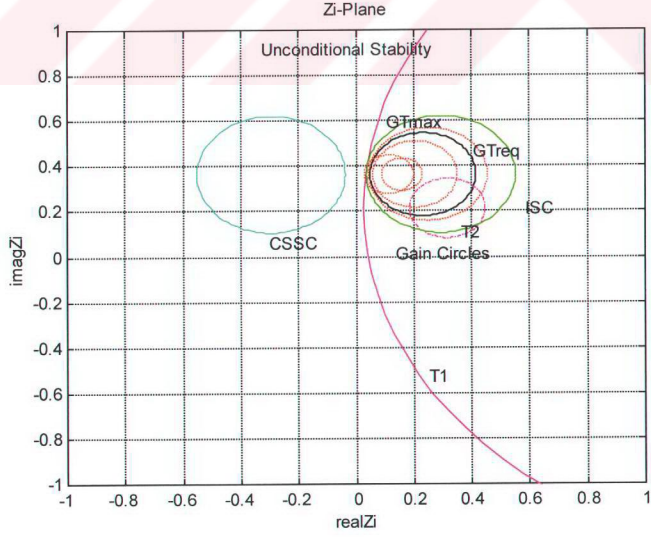
f=1.8GHz:



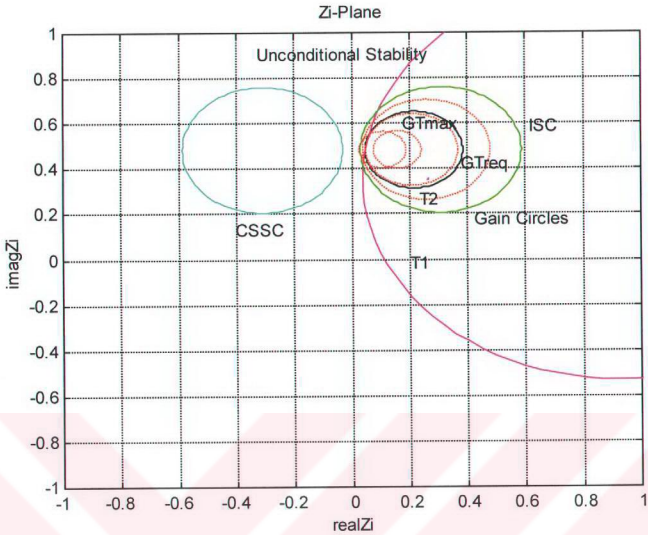
f=2.0GHz:



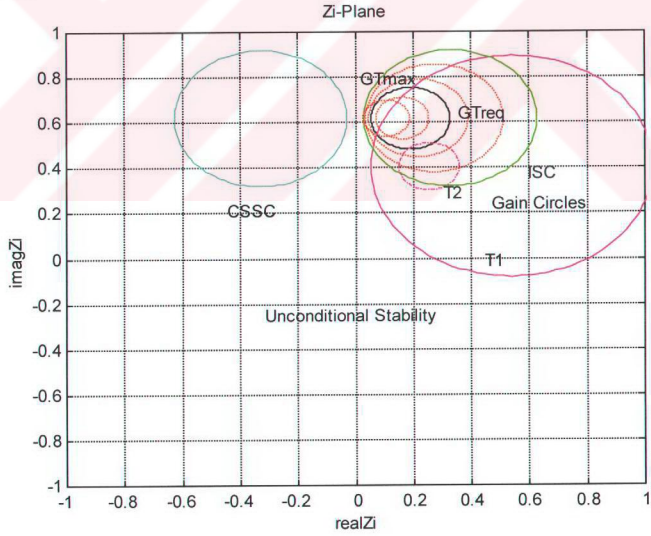
f=2.5GHz:



f=3.0GHz:



f=3.5GHz:



Şekil 6.1.1 Performans Analizi sonucunda farklı frekanslarda elde edilen daireler

6.1.2.5 Daire Kesişimlerinin Özetini İçeren Dosya (out_gra.txt)

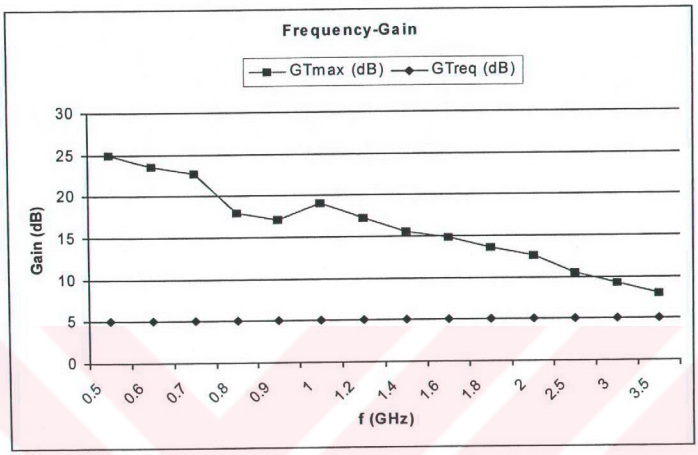
Transistor: ne02135c(VCE=10V, IC=20mA) from Neural Network

Terminations for GTmax and One of Intersection points on T1/T2 for GTreq:

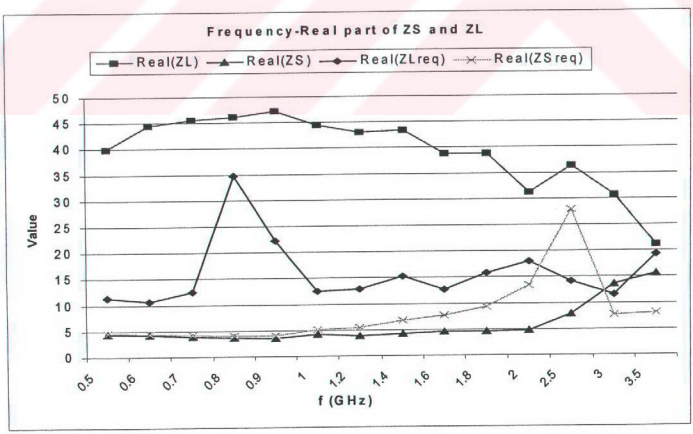
Vireq	Freq(dB)	f(Hz)	GTmax (dB)	Real(ZL)	Imag(ZL)	Real(ZS)	Imag(ZS)	GTreq (dB)	Real(ZLreq)	Imag(ZLreq)	Real(ZSreq)	Imag(ZSreq)
1.5	4.5	500000000	24.946712	39.864893	52.505761	4.494323	12.800920	5.0	11.315847	-281.002463	4.524513	-0.825067
1.5	4.5	600000000	23.498857	44.489503	54.276077	4.155577	1.592590	5.0	10.600229	-237.278148	4.372128	-3.213307
1.5	4.5	700000000	22.736585	45.517755	44.896292	3.933774	-0.142904	5.0	12.476389	-214.540036	4.222784	-4.854890
1.5	4.5	800000000	18.010374	46.140687	44.671759	3.612771	0.892663	5.0	34.820439	-328.177821	4.114916	-6.567547
1.5	4.5	900000000	17.120170	47.162384	43.994254	3.430617	-1.399036	5.0	22.324149	-229.412130	4.115371	-9.020417
1.5	4.5	1000000000	19.059597	44.575364	47.433034	4.180228	-5.743288	5.0	12.528906	-131.575627	5.185716	-12.460471
1.5	4.5	1200000000	17.207207	42.928912	55.907590	3.808704	-9.056797	5.0	12.817019	-106.419972	5.544254	-17.668185
1.5	4.5	1400000000	15.598156	43.468934	53.122188	4.223997	-11.651272	5.0	15.213134	-87.113060	6.802323	-21.298190
1.5	4.5	1600000000	14.858138	38.711179	61.755573	4.550544	-13.773671	5.0	12.764893	-62.739587	7.648415	-22.978031
1.5	4.5	1800000000	13.628425	38.858365	61.208280	4.631245	-16.574767	5.0	15.886455	-58.731561	9.376657	-28.099301
1.5	4.5	2000000000	12.582872	31.310530	53.396584	4.737618	-17.764142	5.0	17.970418	-49.864212	13.435799	-33.037335
1.5	4.5	2500000000	10.559413	36.328916	51.868256	7.985434	-25.635646	5.0	14.074605	-11.192430	27.897427	-37.935101
1.5	4.5	3000000000	9.318865	30.675524	50.989250	13.530237	-30.800761	5.0	11.535804	91.944040	7.647767	-25.777602
1.5	4.5	3500000000	8.033522	21.056922	42.669700	15.706389	-26.134504	5.0	19.292194	8.533826	8.066211	-16.487239

6.1.2.6 Frekans-Kazanç ve ZL,ZS Değişimi

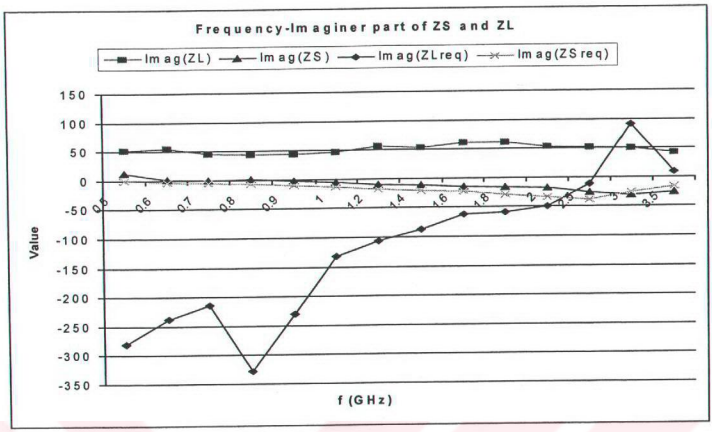
Bu verileri grafiksel olarak ifade edersek aşağıdaki durum elde edilir.



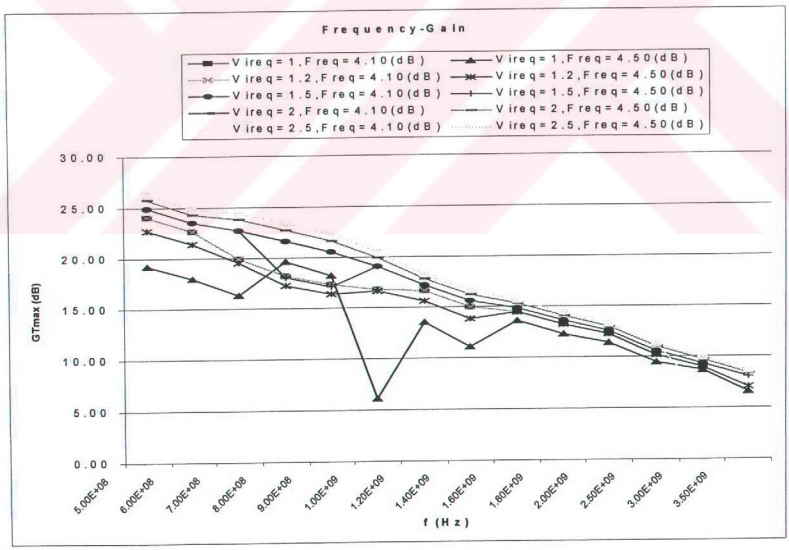
Şekil 6.1.2 Vireq=1.5, Freq=4.5 dB için maksimum Kazanç-Frekans değişimi



Şekil 6.1.3 Vireq=1.5, Freq=4.5 dB için Frekans-Re{ZL/ZS} değişimi



Şekil 6.1.4 Vireq=1.5, Freq=4.5 dB için Frekans-Im{ZL/ZS} değişimi

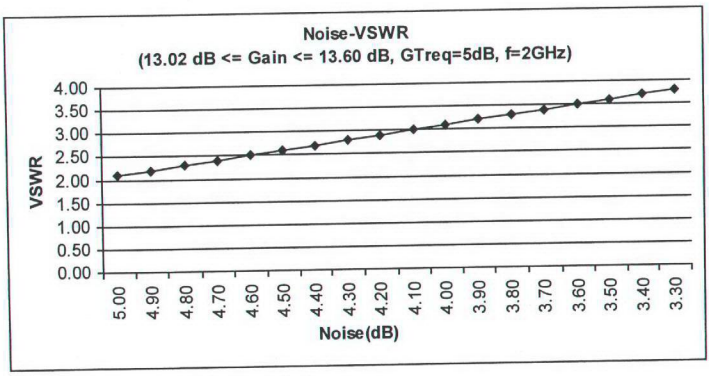


Şekil 6.1.5 Çeşitli Vireq ve Freq'a yönelik maksimum kazanç değişimi

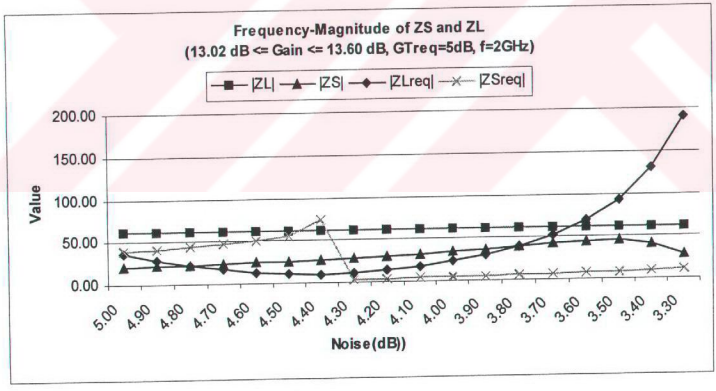
6.1.2.7 Kazanç-VSWR-Gürültü Değişimleri

Çizelge 6.1.1 f=2GHz için Kazanç-VSWR-Gürültü değişimleri

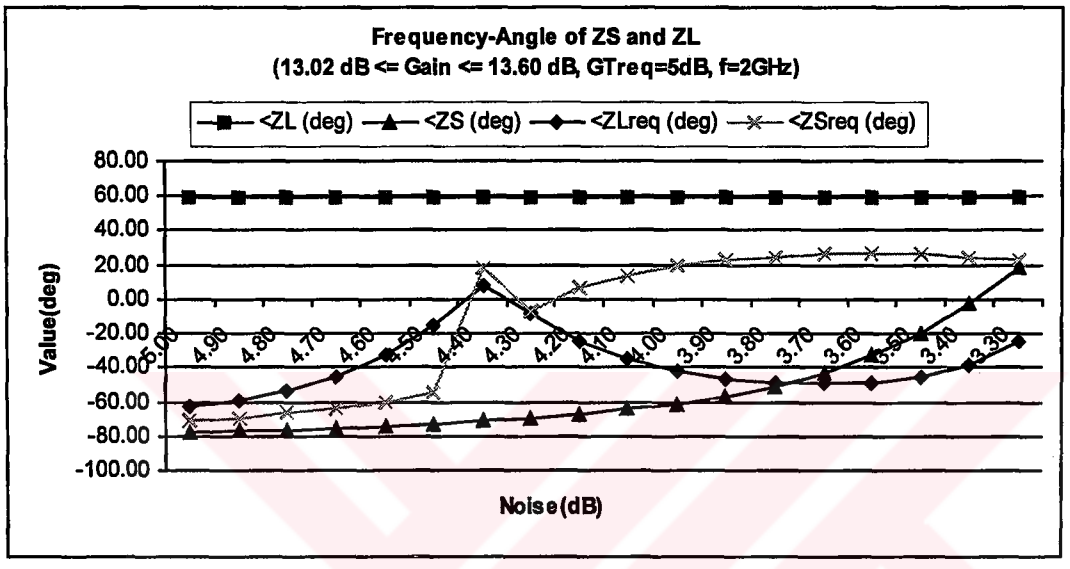
Vireq	Freq	f	Gtmax	Real	Imag	Real	Imag	Gtreq	Real	Imag	Real	Imag
	(dB)	(Hz)	(dB)	(ZL)	(ZL)	(ZS)	(ZS)	(dB)	(ZLreq)	(ZLreq)	(ZSreq)	(ZSreq)
2.10	5.00	2.00E+09	13.02	31.31	53.40	4.23	-19.86	5.00	16.99	-32.55	12.37	-36.70
2.20	4.90	2.00E+09	13.08	31.31	53.40	4.71	-20.73	5.00	14.84	-24.46	14.60	-38.54
2.30	4.80	2.00E+09	13.13	31.31	53.40	5.26	-21.63	5.00	13.22	-17.75	17.33	-40.36
2.40	4.70	2.00E+09	13.17	31.31	53.40	5.90	-22.57	5.00	11.96	-12.05	20.77	-42.12
2.50	4.60	2.00E+09	13.22	31.31	53.40	6.66	-23.55	5.00	10.93	-7.09	25.30	-43.74
2.60	4.50	2.00E+09	13.26	31.31	53.40	7.57	-24.58	5.00	10.09	-2.71	32.61	-45.23
2.70	4.40	2.00E+09	13.29	31.31	53.40	8.68	-25.66	5.00	9.35	1.32	70.02	23.24
2.80	4.30	2.00E+09	13.33	31.31	53.40	10.04	-26.80	5.00	10.52	-1.44	3.41	-0.41
2.90	4.20	2.00E+09	13.36	31.31	53.40	11.73	-27.97	5.00	12.25	-5.56	3.74	0.47
3.00	4.10	2.00E+09	13.40	31.31	53.40	13.89	-29.13	5.00	14.43	-10.25	4.08	1.05
3.10	4.00	2.00E+09	13.43	31.31	53.40	16.66	-30.18	5.00	17.24	-15.68	4.46	1.55
3.20	3.90	2.00E+09	13.45	31.31	53.40	20.28	-30.91	5.00	21.00	-22.10	4.88	2.03
3.30	3.80	2.00E+09	13.48	31.31	53.40	24.98	-30.89	5.00	26.21	-29.84	5.37	2.47
3.40	3.70	2.00E+09	13.51	31.31	53.40	30.90	-29.25	5.00	33.80	-39.34	5.94	2.90
3.50	3.60	2.00E+09	13.53	31.31	53.40	37.53	-24.50	5.00	45.58	-51.19	6.61	3.28
3.60	3.50	2.00E+09	13.55	31.31	53.40	42.50	-14.94	5.00	65.39	-65.71	7.39	3.61
3.70	3.40	2.00E+09	13.58	31.31	53.40	40.76	-1.43	5.00	102.07	-80.76	8.34	3.85
3.80	3.30	2.00E+09	13.60	31.31	53.40	27.60	9.08	5.00	173.52	-78.25	9.47	3.93



Şekil 6.1.6 Çeşitli Vireq ve Freq'a yönelik maksimum kazancın sınırlı kalması durumu



Şekil 6.1.7 Maksimum kazancın sınırlı kalması durumunda sonlandırmaların genlikleri



Şekil 6.1.8 Maksimum kazancın sınırlı kalması durumunda sonlandırmaların açıları

6.1.2.8 Gürültü-VSWR-Kazanç Değişimleri

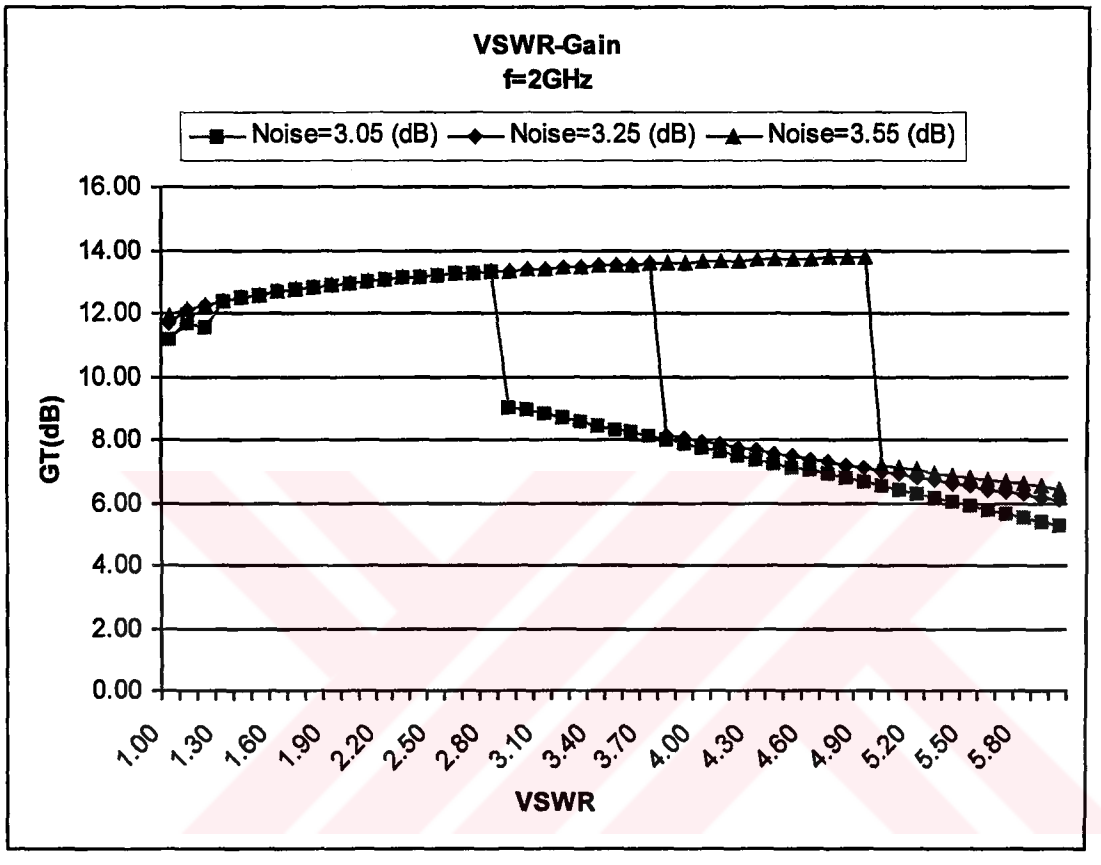
Freq=3.05 dB iken elde edilen Gürültü-VSWR-Kazanç değişimleri aşağıda verilmiştir.

Çizelge 6.1.2 Freq=3.05 dB için Gürültü-VSWR-Kazanç değişimleri

Vireq	Freq (dB)	f (Hz)	Gtmax (dB)	Real (ZL)	Imag (ZL)	Real (ZS)	Imag (ZS)	Gtreq (dB)	Real (ZLreq)	Imag (ZLreq)	Real (ZSreq)	Imag (ZSreq)
1.00	3.05	2.00E+09	11.15	30.68	17.92	10.68	-10.20	5.00	8.61	-26.69	22.59	-12.50
1.10	3.05	2.00E+09	11.66	34.95	43.59	10.68	-10.20	5.00	26.60	-88.43	18.32	-13.89
1.20	3.05	2.00E+09	11.53	31.69	52.68	10.68	-10.20	5.00	60.93	-152.96	17.02	-13.87
1.30	3.05	2.00E+09	12.37	31.31	53.40	12.87	-12.36	5.00	6.31	-6.78	9.10	-4.65
1.40	3.05	2.00E+09	12.48	31.31	53.40	14.34	-13.18	5.00	7.84	-13.23	9.34	-3.06
1.50	3.05	2.00E+09	12.58	31.31	53.40	15.73	-13.65	5.00	9.57	-19.39	9.58	-2.23
1.60	3.05	2.00E+09	12.67	31.31	53.40	17.10	-13.88	5.00	11.60	-25.64	9.80	-1.67
1.70	3.05	2.00E+09	12.76	31.31	53.40	18.47	-13.88	5.00	14.02	-32.16	10.01	-1.24
1.80	3.05	2.00E+09	12.83	31.31	53.40	19.85	-13.67	5.00	16.94	-39.06	10.19	-0.88
1.90	3.05	2.00E+09	12.90	31.31	53.40	21.21	-13.23	5.00	20.50	-46.44	10.37	-0.58
2.00	3.05	2.00E+09	12.96	31.31	53.40	22.54	-12.53	5.00	24.87	-54.38	10.54	-0.31
2.10	3.05	2.00E+09	13.02	31.31	53.40	23.79	-11.57	5.00	30.29	-62.96	10.71	-0.07
2.20	3.05	2.00E+09	13.08	31.31	53.40	24.91	-10.31	5.00	37.08	-72.26	10.87	0.15
2.30	3.05	2.00E+09	13.13	31.31	53.40	25.81	-8.76	5.00	45.67	-82.28	11.02	0.34
2.40	3.05	2.00E+09	13.17	31.31	53.40	26.41	-6.93	5.00	56.64	-92.97	11.17	0.53
2.50	3.05	2.00E+09	13.22	31.31	53.40	26.58	-4.85	5.00	70.75	-104.10	11.32	0.70
2.60	3.05	2.00E+09	13.26	31.31	53.40	26.20	-2.57	5.00	89.00	-115.15	11.47	0.86
2.70	3.05	2.00E+09	13.29	31.31	53.40	25.06	-0.21	5.00	112.60	-125.03	11.61	1.00
2.80	3.05	2.00E+09	13.33	31.31	53.40	22.70	2.13	5.00	142.78	-131.74	11.76	1.14
2.90	3.05	2.00E+09	9.04	30.96	54.02	18.48	3.58	5.00	180.20	-131.85	11.90	1.28
3.00	3.05	2.00E+09	8.93	30.04	55.53	18.48	3.58	5.00	223.66	-120.43	12.03	1.40
3.10	3.05	2.00E+09	8.81	29.12	56.90	18.48	3.58	5.00	268.18	-92.16	12.17	1.52
3.20	3.05	2.00E+09	8.70	28.19	58.14	18.48	3.58	5.00	304.09	-44.93	12.31	1.63
3.30	3.05	2.00E+09	8.58	27.27	59.27	18.48	3.58	5.00	320.42	15.61	12.44	1.74
3.40	3.05	2.00E+09	8.46	26.37	60.30	18.48	3.58	5.00	312.40	76.68	12.58	1.85
3.50	3.05	2.00E+09	8.34	25.49	61.23	18.48	3.58	5.00	285.11	126.08	12.71	1.95

3.60	3.05	2.00E+09	8.22	24.63	62.08	18.48	3.58	5.00	248.66	158.86	12.85	2.04
3.70	3.05	2.00E+09	8.10	23.79	62.85	18.48	3.58	5.00	211.52	176.66	12.98	2.13
3.80	3.05	2.00E+09	7.98	22.98	63.55	18.48	3.58	5.00	178.18	183.75	13.12	2.22
3.90	3.05	2.00E+09	7.86	22.19	64.20	18.48	3.58	5.00	150.07	184.12	13.25	2.30
4.00	3.05	2.00E+09	7.74	21.43	64.79	18.48	3.58	5.00	127.02	180.65	13.39	2.39
4.10	3.05	2.00E+09	7.62	20.69	65.32	18.48	3.58	5.00	108.32	175.19	13.52	2.46
4.20	3.05	2.00E+09	7.50	19.98	65.82	18.48	3.58	5.00	93.17	168.83	13.66	2.54
4.30	3.05	2.00E+09	7.38	19.29	66.27	18.48	3.58	5.00	80.81	162.18	13.80	2.61
4.40	3.05	2.00E+09	7.26	18.63	66.68	18.48	3.58	5.00	70.67	155.59	13.94	2.68
4.50	3.05	2.00E+09	7.14	17.99	67.06	18.48	3.58	5.00	62.27	149.24	14.08	2.75
4.60	3.05	2.00E+09	7.02	17.38	67.41	18.48	3.58	5.00	55.24	143.20	14.22	2.82
4.70	3.05	2.00E+09	6.90	16.79	67.74	18.48	3.58	5.00	49.31	137.51	14.37	2.88
4.80	3.05	2.00E+09	6.77	16.21	68.03	18.48	3.58	5.00	44.26	132.15	14.52	2.95
4.90	3.05	2.00E+09	6.65	15.66	68.31	18.48	3.58	5.00	39.92	127.13	14.67	3.01
5.00	3.05	2.00E+09	6.53	15.13	68.56	18.48	3.58	5.00	36.16	122.40	14.83	3.07
5.10	3.05	2.00E+09	6.41	14.62	68.80	18.48	3.58	5.00	32.87	117.94	14.99	3.12
5.20	3.05	2.00E+09	6.28	14.12	69.02	18.48	3.58	5.00	29.97	113.72	15.15	3.18
5.30	3.05	2.00E+09	6.16	13.65	69.22	18.48	3.58	5.00	27.40	109.71	15.33	3.23
5.40	3.05	2.00E+09	6.04	13.19	69.40	18.48	3.58	5.00	25.09	105.88	15.51	3.29
5.50	3.05	2.00E+09	5.91	12.74	69.58	18.48	3.58	5.00	23.01	102.19	15.70	3.34
5.60	3.05	2.00E+09	5.79	12.31	69.74	18.48	3.58	5.00	21.12	98.62	15.90	3.38
5.70	3.05	2.00E+09	5.66	11.90	69.89	18.48	3.58	5.00	19.37	95.11	16.12	3.43
5.80	3.05	2.00E+09	5.54	11.50	70.03	18.48	3.58	5.00	17.74	91.62	16.35	3.47
5.90	3.05	2.00E+09	5.41	11.11	70.15	18.48	3.58	5.00	16.19	88.08	16.62	3.52
6.00	3.05	2.00E+09	5.28	10.73	70.28	18.48	3.58	5.00	14.67	84.37	16.93	3.55

Bu durumun grafiksel ifadesi diğer gürültü değerlerindeki değişimlerde göz önüne alındığında aşağıdaki şekilde olmaktadır.



Şekil 6.1.9 f=2GHz için Gürültü-VSWR-Kazanç değişimleri

6.1.2.9 VSWR-Gürültü-Kazanç Değişimleri

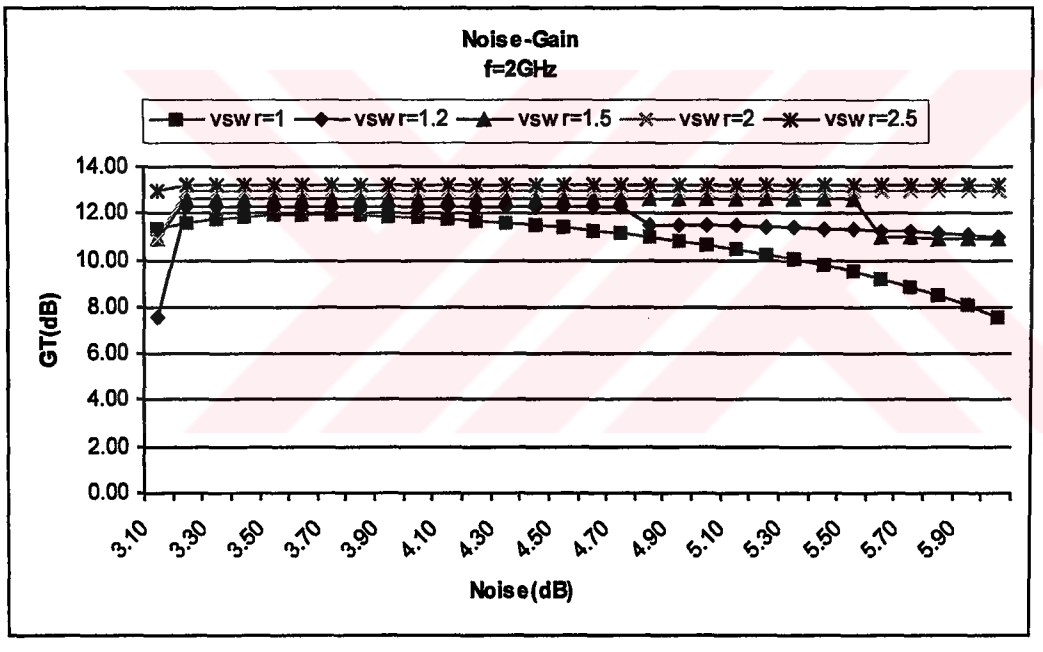
Vireq=1 iken elde edilen VSWR-Gürültü-Kazanç değişimleri aşağıda verilmiştir.

Çizelge 6.1.3 Vireq=1 için VSWR-Gürültü- Kazanç değişimleri

Vireq	Freq	f	Gtmax	Real	Imag	Real	Imag	Gtreq	Real	Imag	Real	Imag
	(dB)	(Hz)	(dB)	(ZL)	(ZL)	(ZS)	(ZS)	(dB)	(ZLreq)	(ZLreq)	(ZSreq)	(ZSreq)
1.00	3.10	2.00E+09	11.36	32.91	22.33	9.98	-10.67	5.00	10.01	-33.79	22.38	-14.69
1.00	3.20	2.00E+09	11.62	35.24	30.37	8.86	-11.33	5.00	13.35	-48.71	21.05	-18.05
1.00	3.30	2.00E+09	11.78	35.70	37.33	7.99	-11.77	5.00	17.85	-65.72	19.23	-20.28
1.00	3.40	2.00E+09	11.87	35.03	43.21	7.29	-12.09	5.00	24.25	-86.16	17.32	-21.68
1.00	3.50	2.00E+09	11.92	33.67	48.12	6.70	-12.33	5.00	33.92	-111.83	15.53	-22.49
1.00	3.60	2.00E+09	11.94	31.95	52.18	6.20	-12.52	5.00	49.52	-145.44	13.91	-22.91
1.00	3.70	2.00E+09	11.94	30.04	55.54	5.76	-12.67	5.00	77.00	-191.36	12.49	-23.06
1.00	3.80	2.00E+09	11.92	28.06	58.30	5.37	-12.79	5.00	131.23	-255.90	11.24	-23.03
1.00	3.90	2.00E+09	11.88	26.11	60.58	5.03	-12.89	5.00	254.55	-340.43	10.15	-22.88
1.00	4.00	2.00E+09	11.83	24.21	62.47	4.73	-12.97	5.00	555.91	-361.75	9.20	-22.66
1.00	4.10	2.00E+09	11.76	22.39	64.03	4.45	-13.04	5.00	864.75	92.98	8.37	-22.38
1.00	4.20	2.00E+09	11.69	20.68	65.33	4.20	-13.10	5.00	511.73	479.05	7.64	-22.07
1.00	4.30	2.00E+09	11.60	19.06	66.41	3.98	-13.16	5.00	239.96	440.25	7.00	-21.74
1.00	4.40	2.00E+09	11.50	17.55	67.32	3.77	-13.20	5.00	128.38	359.89	6.44	-21.40
1.00	4.50	2.00E+09	11.39	16.14	68.07	3.58	-13.24	5.00	77.95	299.53	5.93	-21.06
1.00	4.60	2.00E+09	11.27	14.82	68.71	3.41	-13.27	5.00	51.78	256.55	5.48	-20.71
1.00	4.70	2.00E+09	11.14	13.58	69.24	3.24	-13.30	5.00	36.67	225.13	5.08	-20.37
1.00	4.80	2.00E+09	10.99	12.43	69.69	3.09	-13.33	5.00	27.20	201.31	4.72	-20.02
1.00	4.90	2.00E+09	10.83	11.36	70.07	2.95	-13.35	5.00	20.89	182.67	4.39	-19.68
1.00	5.00	2.00E+09	10.66	10.36	70.39	2.82	-13.37	5.00	16.49	167.66	4.09	-19.35
1.00	5.10	2.00E+09	10.47	9.42	70.66	2.70	-13.39	5.00	13.30	155.28	3.82	-19.01
1.00	5.20	2.00E+09	10.26	8.54	70.89	2.59	-13.41	5.00	10.90	144.84	3.58	-18.68
1.00	5.30	2.00E+09	10.03	7.72	71.08	2.48	-13.42	5.00	9.07	135.90	3.35	-18.35
1.00	5.40	2.00E+09	9.79	6.95	71.24	2.38	-13.43	5.00	7.63	128.09	3.14	-18.02
1.00	5.50	2.00E+09	9.51	6.22	71.38	2.28	-13.45	5.00	6.47	121.19	2.95	-17.70
1.00	5.60	2.00E+09	9.21	5.54	71.49	2.19	-13.46	5.00	5.54	114.98	2.77	-17.36

1.00	5.70	2.00E+09	8.87	4.90	71.59	2.11	-13.47	5.00	4.76	109.32	2.61	-17.03
1.00	5.80	2.00E+09	8.49	4.30	71.67	2.03	-13.48	5.00	4.11	104.08	2.45	-16.68
1.00	5.90	2.00E+09	8.07	3.73	71.73	1.95	-13.48	5.00	3.56	99.14	2.31	-16.32
1.00	6.00	2.00E+09	7.58	3.19	71.78	1.88	-13.49	5.00	3.08	94.38	2.17	-15.94

Bu durumun grafiksel ifadesi diğer VSWR değerlerindeki değişimlerde göz önüne alındığında aşağıdaki şekilde olmaktadır.



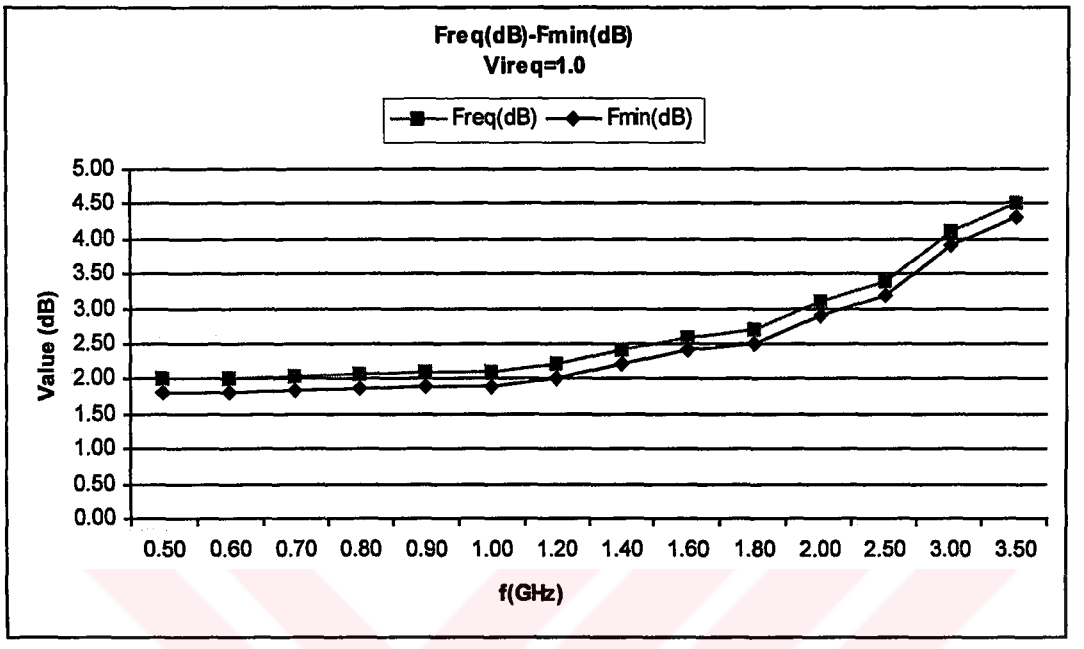
Şekil 6.1.10 f=2GHz için VSWR-Gürültü- Kazanç değişimleri

6.1.2.10 Gürültünün Minimum Gürültüyü İzlemsi Durumundaki Değişimler

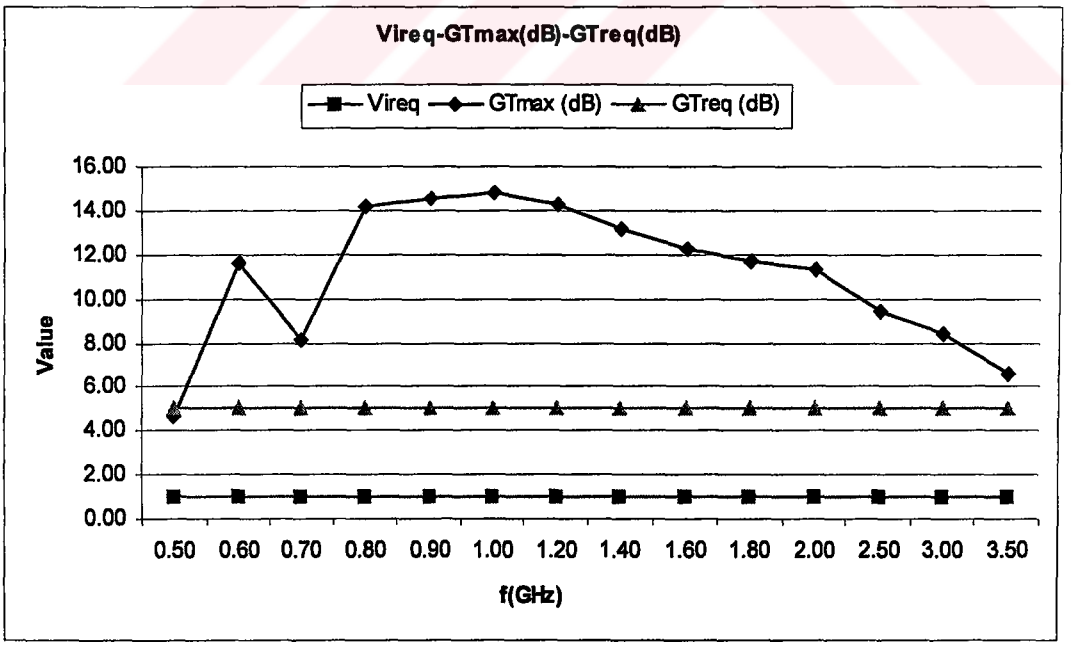
Aşağıda Freq=Fmin+DeltaF olması durumundaki değişimler görülmektedir.

Çizelge 6.1.4 Vireq=1 için Freq=Fmin+DeltaF değişimleri

Vireq	Freq (dB)	f (GHz)	GTmax (dB)	Real (ZL)	Imag (ZL)	Real (ZS)	Imag (ZS)	Gtreq (dB)	Real (ZLreq)	Imag (ZLreq)	Real (ZSreq)	Imag (ZSreq)
1.00	2.00	0.50	4.62	NaN	-17.60	20.68	7.67	5.00	NaN	NaN	NaN	NaN
1.00	2.01	0.60	11.62	3.31	-17.24	19.29	5.89	5.00	0.88	-30.09	19.48	3.13
1.00	2.03	0.70	8.13	1.58	-11.64	18.06	3.76	5.00	0.86	-17.98	18.27	2.34
1.00	2.07	0.80	14.14	9.52	-9.27	16.67	2.27	5.00	1.57	-33.50	17.87	-1.82
1.00	2.10	0.90	14.54	12.69	-7.03	15.50	0.49	5.00	1.85	-34.62	17.46	-4.16
1.00	2.11	1.00	14.82	19.45	-2.10	15.24	-0.37	5.00	2.73	-38.54	18.63	-6.37
1.00	2.20	1.20	14.26	24.02	5.68	13.56	-2.99	5.00	3.49	-33.16	19.67	-9.47
1.00	2.40	1.40	13.14	27.93	10.35	13.05	-5.11	5.00	5.12	-33.08	20.52	-11.14
1.00	2.60	1.60	12.27	25.13	8.86	12.05	-6.16	5.00	5.75	-29.68	19.52	-11.35
1.00	2.70	1.80	11.74	30.78	14.71	11.36	-9.04	5.00	7.62	-30.58	20.98	-13.78
1.00	3.10	2.00	11.36	32.91	22.33	9.98	-10.67	5.00	10.01	-33.79	22.38	-14.69
1.00	3.40	2.50	9.40	31.76	25.55	10.27	-16.05	5.00	11.93	-17.48	22.08	-16.94
1.00	4.10	3.00	8.42	31.13	30.57	9.38	-22.40	5.00	17.42	-16.57	21.19	-22.53
1.00	4.50	3.50	6.57	18.35	17.93	10.63	-26.09	5.00	18.61	-2.96	17.32	-25.15



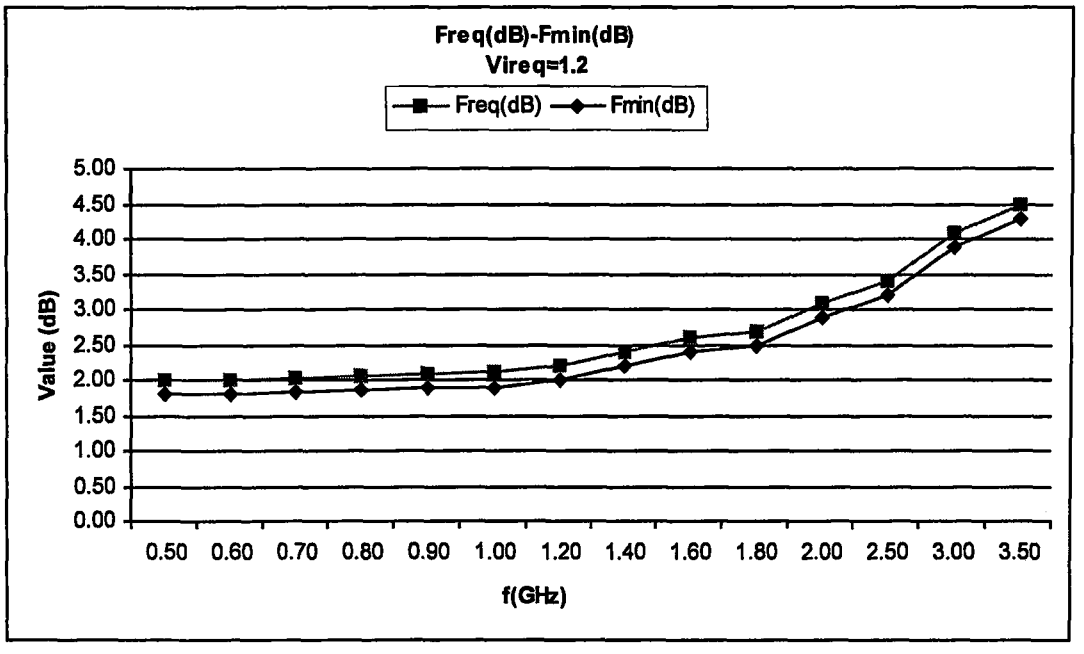
Şekil 6.1.11 Vireq=1 için Freq-Fmin değişimi



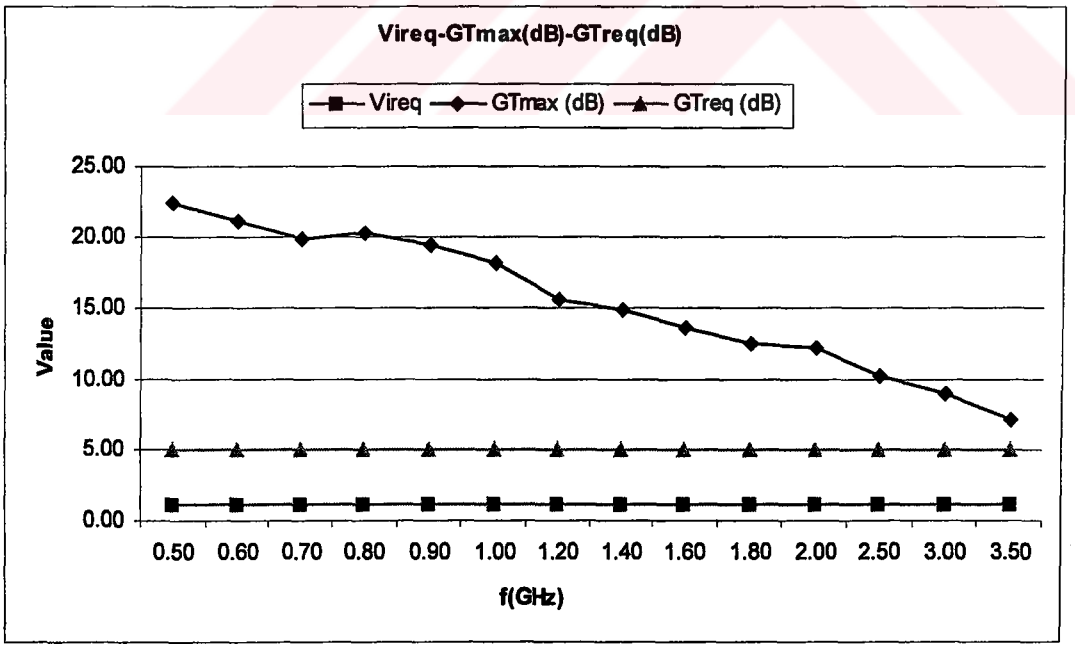
Şekil 6.1.12 Vireq=1 için Freq-Fmin'e karşılık gelen Kazanç değişimleri

Çizelge 6.1.5 Vireq=1.2 için Freq=Fmin+DeltaF değişimleri

Vireq	Freq (dB)	f (GHz)	Gtmax (dB)	Real (ZL)	Imag (ZL)	Real (ZS)	Imag (ZS)	Gtreq (dB)	Real (ZLreq)	Imag (ZLreq)	Real (ZSreq)	Imag (ZSreq)
1.20	2.00	0.50	22.42	46.28	25.67	20.68	7.67	5.00	13.26	-333.22	21.17	1.37
1.20	2.01	0.60	21.14	49.44	27.90	19.29	5.89	5.00	11.79	-274.61	20.22	-0.41
1.20	2.03	0.70	19.89	46.59	34.10	18.06	3.76	5.00	14.20	-251.53	19.30	-1.39
1.20	2.07	0.80	20.38	46.45	42.68	17.48	-0.81	5.00	61.97	-486.41	18.17	-2.50
1.20	2.10	0.90	19.39	47.43	40.28	16.75	-2.86	5.00	35.86	-326.93	17.43	-4.10
1.20	2.11	1.00	18.16	44.58	47.43	16.22	-2.90	5.00	41.49	-307.04	17.84	-5.45
1.20	2.20	1.20	15.66	43.87	52.07	13.56	-2.99	5.00	21.04	-169.99	17.74	-8.27
1.20	2.40	1.40	14.97	43.47	53.12	14.68	-7.49	5.00	40.50	-198.18	17.53	-9.86
1.20	2.60	1.60	13.58	42.36	49.65	12.05	-6.16	5.00	27.60	-139.44	17.17	-10.63
1.20	2.70	1.80	12.47	40.55	56.70	11.36	-9.04	5.00	4.52	-0.34	11.29	1.99
1.20	3.10	2.00	12.25	31.31	53.40	11.24	-12.28	5.00	100.45	-205.21	16.26	-15.13
1.20	3.40	2.50	10.17	36.33	51.87	8.42	-22.87	5.00	13.64	-17.58	26.96	-31.45
1.20	4.10	3.00	8.99	30.68	50.99	12.47	-24.67	5.00	172.11	-88.29	14.96	-25.22
1.20	4.50	3.50	7.05	22.33	39.46	10.63	-26.09	5.00	86.65	-15.50	15.54	-26.20



Şekil 6.1.13 Vireq=1.2 için Freq-Fmin değişimi



Şekil 6.1.14 Vireq=1.2 için Freq-Fmin'e karşılık gelen Kazanç değişimleri

6.1.3 Empedans Verisi Uydurma Sonuçları

6.1.3.1 Sabit Gürültü değeri durumu

Önceki kısımlarda bulunan empedans verileri ile uydurma devrelerini gerçekleştirmek üzere (Bazan Tolga, 2000) ve (Güneş Filiz, Yarman B. Sıddık, 1999) tarafından yapılan çalışma sonuçları aşağıda verilmiştir.

Çalışma Bandı: 1.5-3.5 GHz, GTreq=5 dB, Vireq=1.5, Freq=4.5 dB

Kullanılan frekans aralığındaki veriler aşağıdaki tabloda verilmiştir.

Çizelge 6.1.6 Uydurma Devresi için kullanılan eleman değerleri

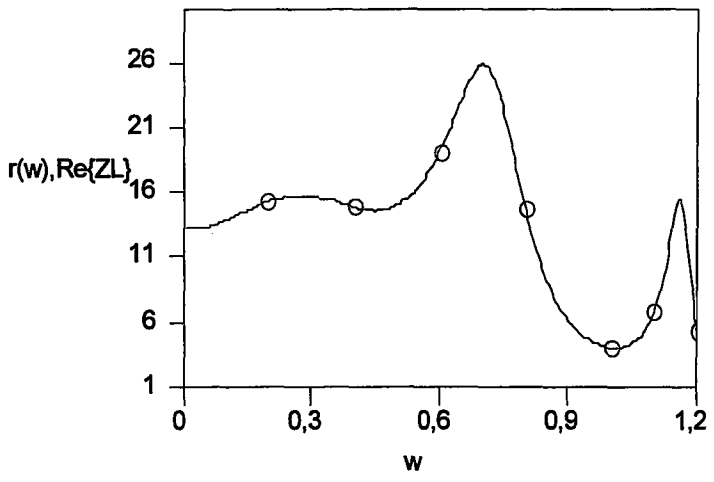
Frekans (GHz)	Z_L (Reel - İmajiner)	Z_S (Reel –İmajiner)
1.5	14.806-73.1j	9.741-27.279j
2	19.093-53.22j	17.417-36.936j
2.5	14.612-12.71j	41.9-35.8588j
3	4.012+50.96j	6.41-23.575j
3.5	5.385+33.69j	10.514-26.034j

R_0=1, $w_0=2\pi f_{max}/1.2$=18.31e9 alınarak 4. Bölümdeki formüllerle elemanların gerçek değerleri F, H, Ohm olarak bulunabilir.

ZL Analizi: İlk yapılacak iş Z_L'nin reel kısmına pozitif ve istenen kurallara uygun minimum hatalı bir fonksiyon uydurmaktır. Bulunacak fonksiyon minimum reaktans fonksiyonunun reel kısmını da ($r(w) = r_{MR}(w)$) temsil edecektir.

$$r(w) := \frac{.391191}{2.9810610^{-2} - .165403w^2 + 1.82325w^4 - 7.69206w^6 + 13.6319w^8 - 10.2746w^{10} + 2.74464w^{12}}$$

r(w)'nın w ile değişimi aşağıdaki gibi sürekli bir eğri olarak bulunacaktır.



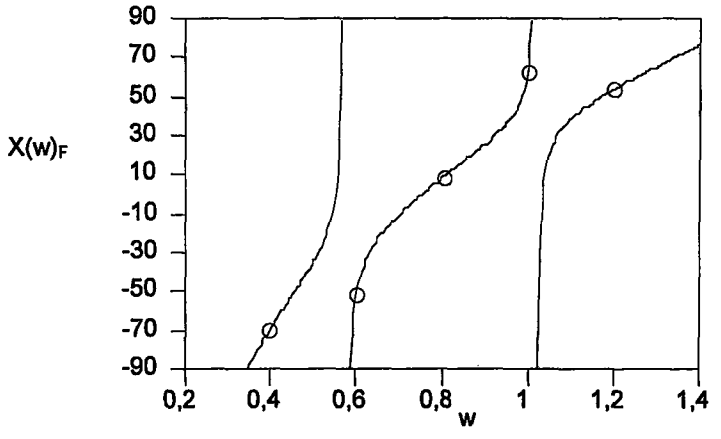
Şekil 6.1.15 r(w)/Re{ZL}-w değişimi

Minimum reaktans fonksiyonunun kendisi, minimum reaktans fonksiyonunun reel kısmı yardımıyla Gewertz Prosedürü kullanılarak kolaylıkla elde edilir ($z(w)_{MR} = z_{MR}(w)$).

$$z(s)_{MR} = \frac{2.26567 + 10.7145s + 18.67537s^2 + 31.76305s^3 + 14.48491s^4 + 17.52192s^5}{.17266 + .71734s + 1.96915s^2 + 2.31452s^3 + 3.66702s^4 + 1.36955s^5 + 1.6567s^6}$$

$$z(w)_{MR} = \frac{\left(2.26567 + 10.7145iw - 18.67537w^2 - 31.76305iw^3 + 14.48491w^4 + 17.52192iw^5\right)}{\left(.17266 + .71734iw - 1.96915w^2 - 2.31452iw^3 + 3.66702w^4 + 1.36955iw^5 - 1.6567w^6\right)}$$

Bu durumda $jx_{MR}(w) = z_{MR}(w) - r_{MR}(w)$ şeklinde bulunabilir. Ayrıca $x_F(w_i) = X_D(w_i) - x_{MR}(w_i)$ eşitliğinden gidilerek Foster fonksiyonu ($X(w)_F = x_F(w)$) bulunabilir.

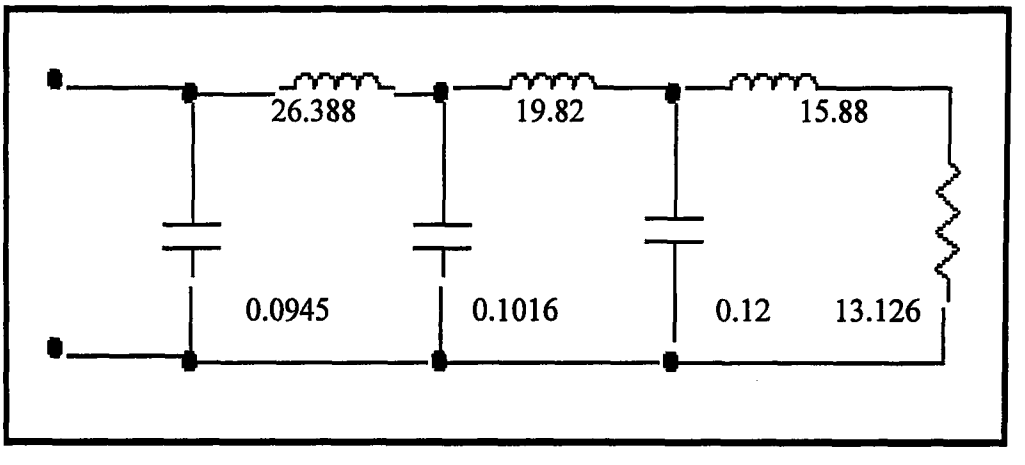


Şekil 6.1.16 ZL için Foster verileri-w değişimi

$$X(w)_F = \frac{-42.1882}{w} + 78.5039w + \frac{1.17511w}{1.02^2 - w^2} + \frac{1.48529w}{.5742^2 - w^2}$$

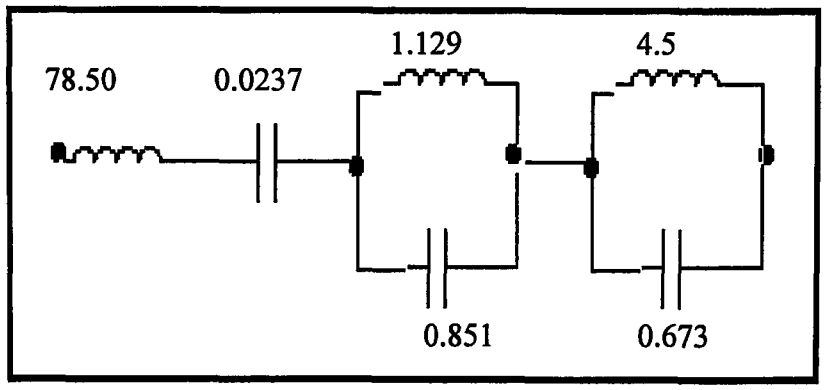
Sonuç olarak minimum reaktans fonksiyonunun ve Foster fonksiyonunun toplamı bize giriş empedansını verir. Devreyi çizerken Foster ve minimum reaktans fonksiyonunu ayrı ayrı çizmek ve sentez etmek bize kolaylık sağlayacaktır.

Aşağıda minimum reaktans fonksiyonunun devresi görülmektedir.



Şekil 6.1.17 ZL için minimum reaktans fonksiyonunun devresi

Aşağıda Foster Fonksiyonunun devresi görülmektedir.

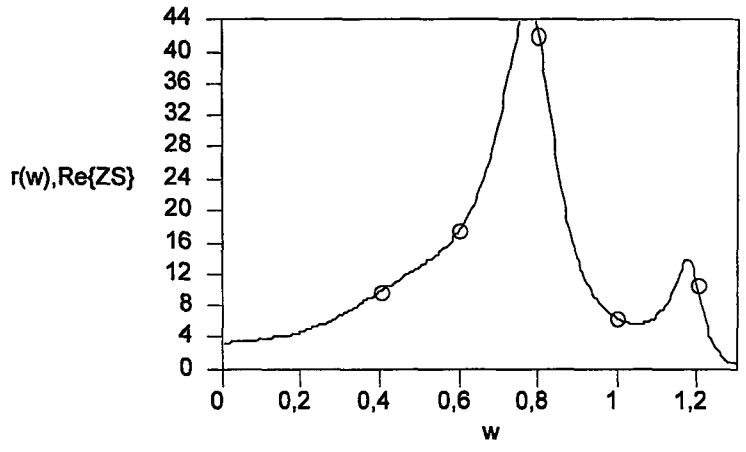


Şekil 6.1.18 ZL için Foster Fonksiyonunun devresi

ZS Analizi: İlk önce ZS verilerinin reel kısmına istenen koşullara uygun bir pozitif reel bir fonksiyon uydurulur. Bu da bize tabi ki minimum reaktans fonksiyonunun reel kısmını verecektir ($r(w) = r_{MR}(w)$).

$$r(w) := \frac{.492774}{.144892 - 1.23264w^2 + 6.06941w^4 - 15.8468w^6 + 20.9899w^8 - 13.1205w^{10} + 3.0726w^{12}}$$

r(w)'nın w ile değişimi aşağıdaki gibi sürekli çizgi olarak bulunacaktır.



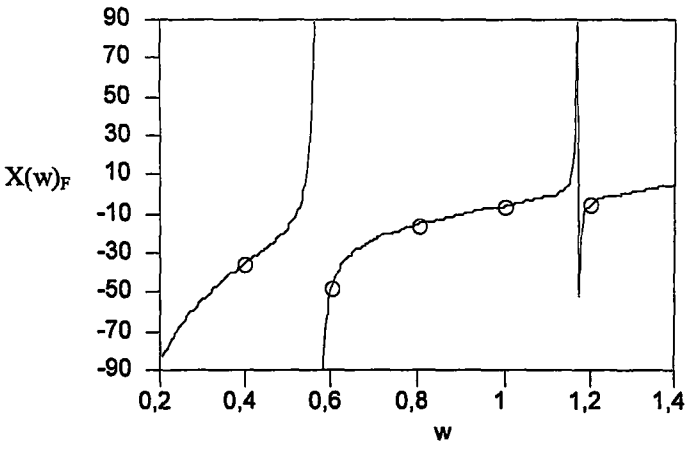
Şekil 6.1.19 r(w)/Re{ZS}-w değişimi

Minimum reaktans fonksiyonunun reel kısmı yardımıyla Gewertz Prosedürü kullanılarak minimum reaktans fonksiyonunun kendisi kolaylıkla elde edilir ($z(w)_{MR} = z_{MR}(w)$).

$$z(s)_{MR} = \frac{1.29456 + 11.43631s + 18.02721s^2 + 34.83862s^3 + 14.34353s^4 + 18.7118s^5}{.38065 + .90512s + 2.69526s^2 + 2.45065s^3 + 4.25754s^4 + 1.34367s^5 + 1.75288s^6}$$

$$z(w)_{MR} = \frac{\left(1.29456 + 11.43631iw - 18.02721w^2 - 34.83862iw^3 + 14.34353w^4 + 18.7118iw^5\right)}{\left(.38065 + .90512iw - 2.69526w^2 - 2.45065iw^3 + 4.25754w^4 + 1.34367iw^5 - 1.75288w^6\right)}$$

Bu durumda $jx_{MR}(w) = z_{MR}(w) - r_{MR}(w)$ şeklinde bulunabilir. Ayrıca $x_F(w_i) = X_D(w_i) - x_{MR}(w_i)$ eşitliğinden gidilerek Foster fonksiyonu $(X(w)_F = x_F(w))$ bulunabilir.

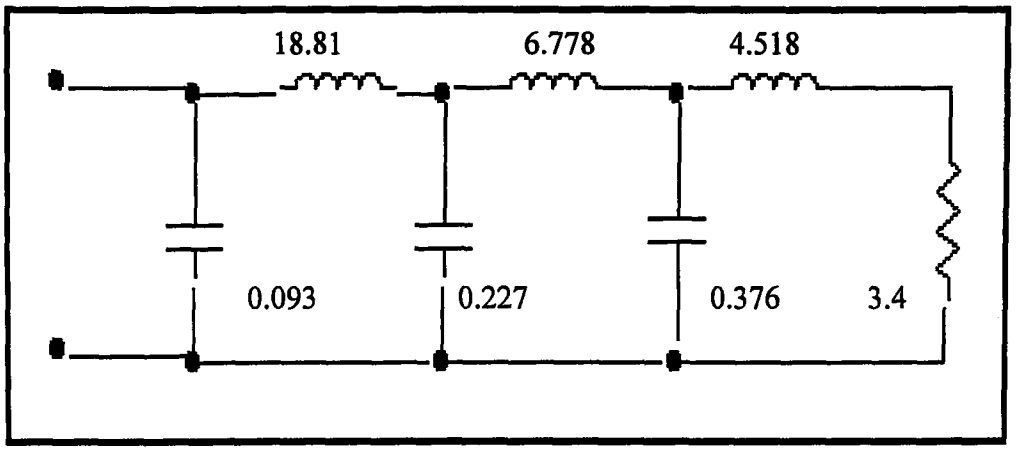


Şekil 6.1.20 ZS için Foster verileri-w değişimi

$$X(w)_F = \frac{-18.1893}{w} + 14.5247w + \frac{1.77073w}{.565^2 - w^2} + \frac{.215252w}{1.18^2 - w^2}$$

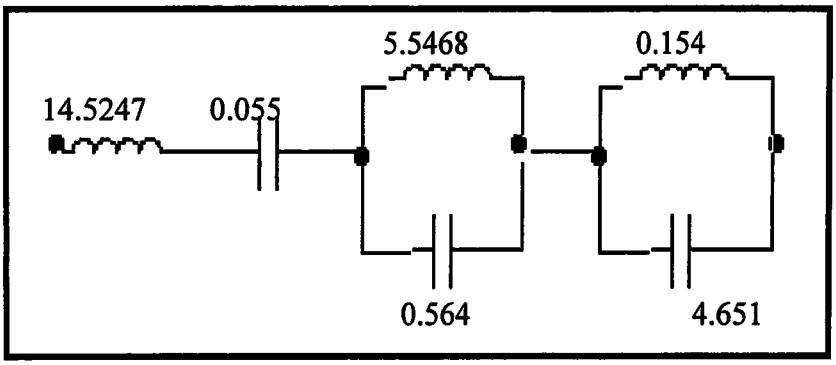
Sonuç olarak minimum reaktans fonksiyonunun ve Foster fonksiyonunun toplamı bize giriş empedansını verir. Devreyi çizerken Fosteri ve minimum reaktans fonksiyonunu ayrı ayrı çizmek ve sentez etmek bize kolaylık sağlayacaktır.

Aşağıda minimum reaktans fonksiyonunun devresi vardır



Şekil 6.1.21 ZS için minimum reaktans fonksiyonunun devresi

Foster fonksiyonunun devresi ise şöyledir:



Şekil 6.1.22 ZS için Foster Fonksiyonunun devresi

6.1.3.2 Gürültünün minimum gürültüyü izlemesi durumu

Gürültünün minimum gürültüyü izlemesi durumundaki ZLreq ve ZSreq değerleri GTreq=5 dB ve Vireq=1.5 değerleri için Çizelge 6.1.7 ve Çizelge 6.1.8'de verilmiştir ve empedans uydurma çalışması yapılmıştır.

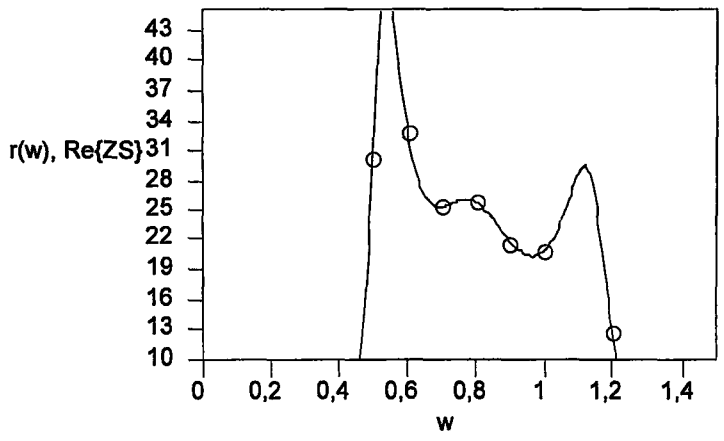
Çizelge 6.1.7 Freq=Fmin+DeltaF durumunda uydurma devresi için kullanılan ZS değerleri

Frekans GHz	Z_S (veriler)
0.5	30.1+8.87j
0.6	32.7-1.27j
0.7	25.3+8.6j
0.8	25.8+10.6j
0.9	21.7+13.7j
1	20.9-3.9j
1.2	12.7+4.85j

R_0=1, $w_0=2\pi f_{max}/1.2$=6.28e9 alınarak 4. Bölümdeki formüllerle elemanların gerçek değerleri F, H, Ohm olarak bulunabilir.

ZS Analizi: Daha önceki kısımda yapılan işlemler burada da tekrarlanırsa aşağıdaki sonuçlar elde edilir.

$$r(w)_{MR}=\frac{6.16\cdot w^4}{1-11.37\cdot w^2+50.41\cdot w^4-110.75w^6+129.277w^8-75.7119w^{10}+17.4352w^{12}}$$



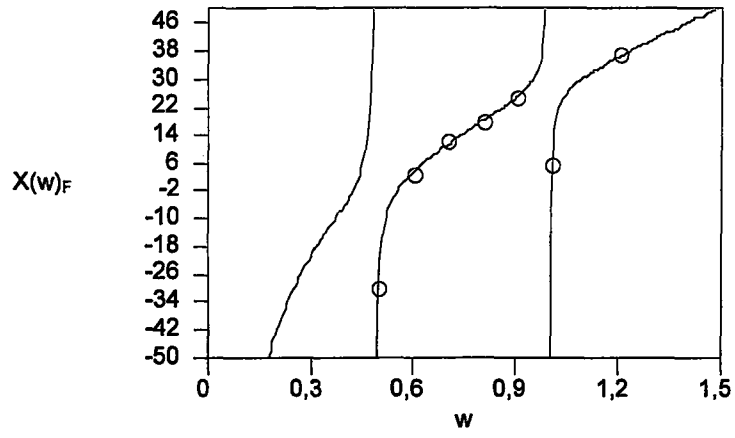
Şekil 6.1.23 r(w)/Re{ZS}-w değişimi

Gewertz Prosedürü ile minimum reaktans fonksiyonu elde edilirse;

$$Z(s)_{MR}=\frac{25.61973s+26.62812s^2+90.20369s^3+34.18497s^4+54.61159s^5}{1+1.03936s+6.22653s^2+3.90589s^3+9.88416s^4+2.61375s^5+4.17555s^6}$$

Sırada Foster fonksiyonunu bulmak var. $x_F(w_i)=X_D(w_i)-x_{MR}(w_i)$ eşitliğinden gidilerek Foster fonksiyonu bulunur.

$$x(w)_F=38.9665\cdot w+\frac{.43557\cdot w}{.99^2-w^2}+\frac{.693855\cdot w}{.488^2-w^2}-\frac{10.2592}{w}$$

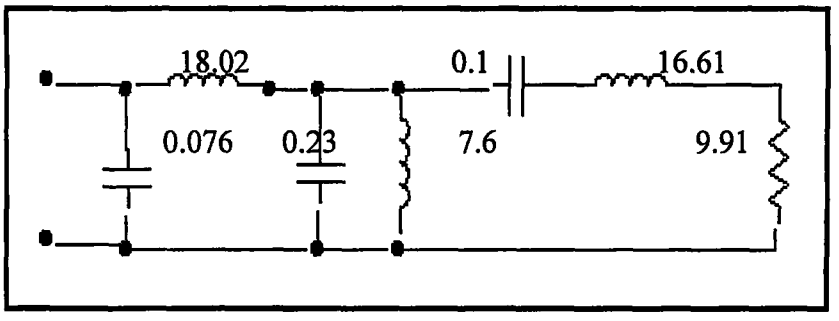


Şekil 6.1.24 ZS için Foster verileri-w değişimi

Sonuç olarak minimum reaktans fonksiyonunun ve Foster fonksiyonunun toplamı bize giriş empedansını verir. Devreyi çizerken Fosteri ve minimum reaktans fonksiyonunu ayrı ayrı çizmek ve sentez etmek bize kolaylık sağlayacaktır.

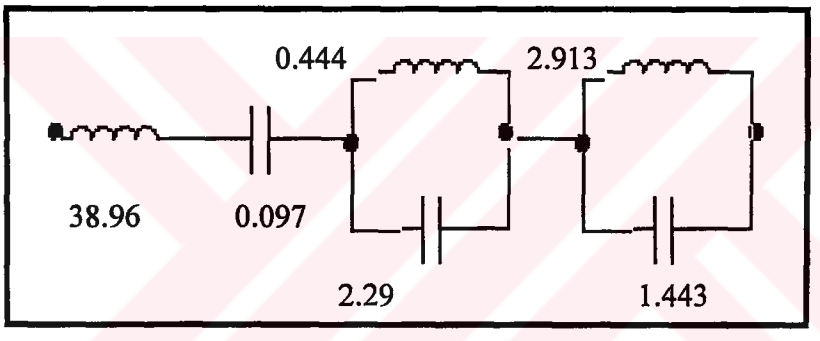
Aşağıda minimum reaktans fonksiyonunun sentezi ve devresi görülmektedir.

$$Z(s)_{MR} = \cfrac{1}{0.076s + \cfrac{1}{18.02s + \cfrac{1}{0.229s + \cfrac{0.1315}{s} + \cfrac{1}{16.61s + \cfrac{1}{0.1 \cdot s} + 9.91}}}}$$



Şekil 6.1.25 ZS için minimum reaktans fonksiyonunun devresi

Foster devresi ise şöyledir:



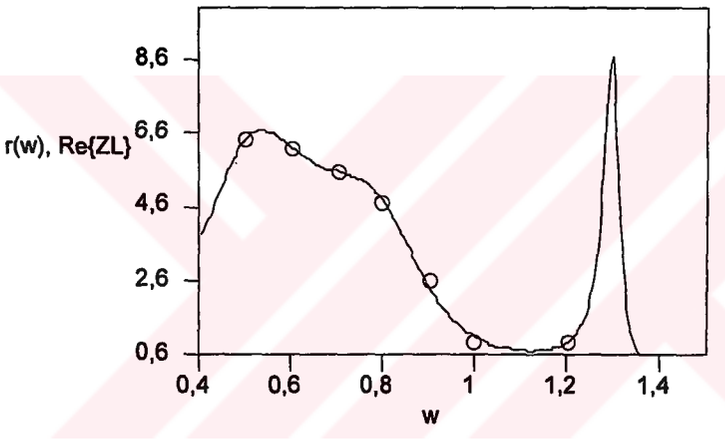
Şekil 6.1.26 ZS için Foster Fonksiyonunun devresi

ZL Analizi: ZL analizi ZS analizine benzer olarak aşağıda yapılmıştır.

Çizelge 6.1.8 Freq=Fmin+DeltaF durumunda uydurma devresi için kullanılan ZL değerleri

Frekans GHz	Z_L (veriler)
0.5	6.41-197.37j
0.6	6.21-167.1j
0.7	5.58-125.4j
0.8	4.72-87.48j
0.9	2.59-40.48j
1	0.907+25.3j
1.2	0.942+40.39j

$$r(w)_{MR} = \frac{0.63}{1 - 10.2715w^2 + 44.84w^4 - 95.53w^6 + 104.223w^8 - 54.33w^{10} + 10.66w^{12}}$$



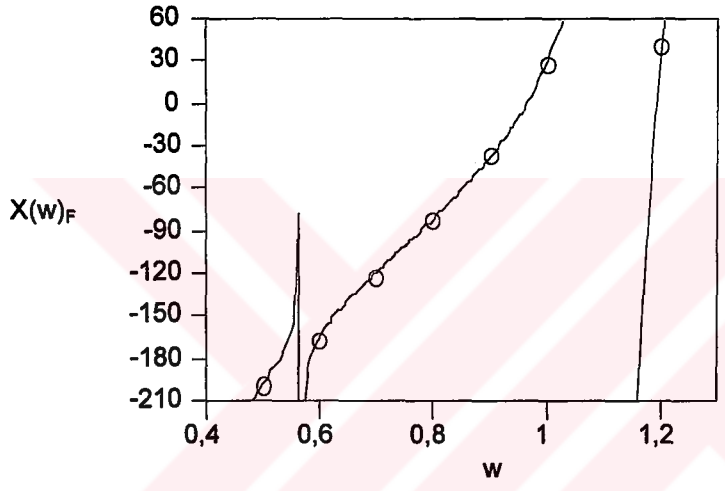
Şekil 6.1.27 r(w)/Re{ZL}-w değişimi

$$Z(s)_{MR} = \frac{.6308 + 7.61733s + 7.79305s^2 + 17.92557s^3 + 5.20221s^4 + 8.30133s^5}{1 + 1.54753s + 6.33318s^2 + 4.26259s^3 + 8.96146s^4 + 2.04632s^5 + 3.26538s^6}$$

$$Z(w)_{MR} = \frac{\left(.6308 + 7.61733iw - 7.79305w^2 - 17.92557iw^3 + 5.20221w^4 + 8.30133iw^5\right)}{\left(1 + 1.54753iw - 6.33318w^2 - 4.26259iw^3 + 8.96146w^4 + 2.04632iw^5 - 3.26538w^6\right)}$$

$x_F(w_i) = X_D(w_i) - x_{MR}(w_i)$ eşitliğinden gidilerek Foster fonksiyonu bulunur.

$$X(w)_F = \frac{115.709}{w} + \frac{51.7751 \cdot w}{1.29^2 - w^2} + \frac{.602041 \cdot w}{.57^2 - w^2} + \frac{18.8473 \cdot w}{1.13^2 - w^2}$$

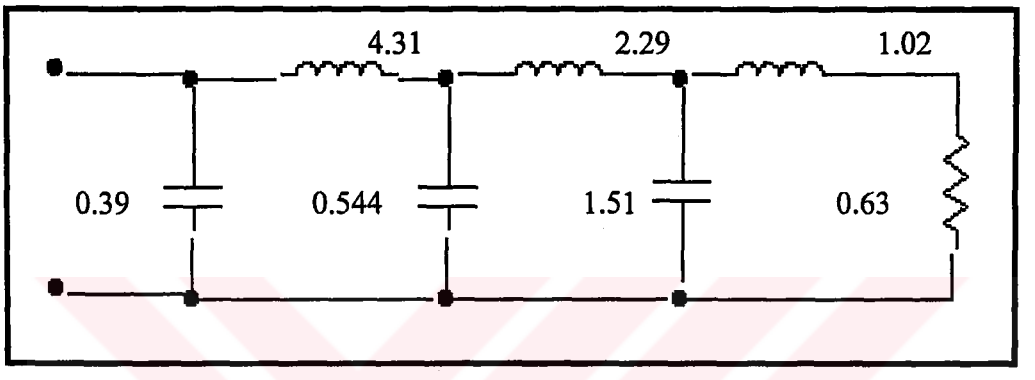


Şekil 6.1.28 ZL için Foster verileri-w değişimi

Sonuç olarak minimum reaktans fonksiyonunun ve Foster fonksiyonunun toplamı bize çıkış empedansını verir. Devreyi çizerken Foster ve minimum reaktans fonksiyonunu ayrı ayrı çizmek ve sentez etmek bize kolaylık sağlayacaktır.

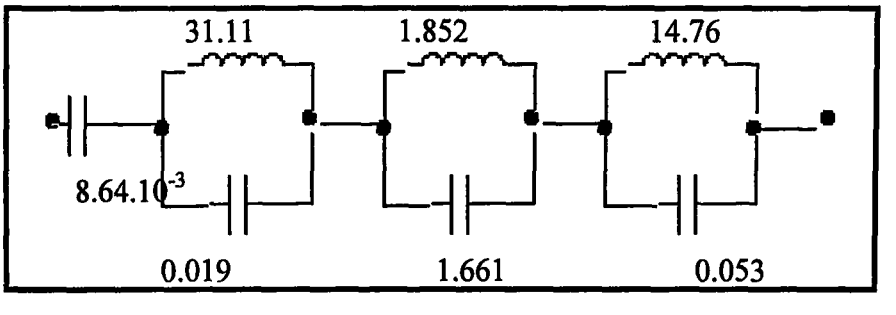
Aşağıda minimum reaktans fonksiyonunun sentezi ve devresi görülmektedir.

$$Z(s)_{MR} = \cfrac{1}{0.39s + \cfrac{1}{4.31s + \cfrac{1}{0.544s + \cfrac{1}{2.29s + \cfrac{1}{1.51s + \cfrac{1}{1.02s + 0.63}}}}}}$$



Şekil 6.1.29 ZL için minimum reaktans fonksiyonunun devresi

Foster fonksiyonu ise şöyledir:



Şekil 6.1.30 ZL için Foster Fonksiyonunun devresi

6.2 Transistor: ne02135(VCE=10V, IC=5mA)

6.2.1 YSA Sonuçları

6.2.1.1 Transistorun Gürültü ve S-parametrelerini İçeren Üretici Verileri

Aşağıdaki dosya transistorun, üreticiden alınan gürültü ve S-parametrelerini içermektedir.

Ne02135.s2p:

```
! FILENAME:      NE02135.S2P  VERSION: 1.0
! NEC PART NUMBER: NE02135      DATE:   4/85
! BIAS CONDITIONS: VCE=10V, IC=5mA
# ang. in deg                          NOISE PARAMETERS (2/81)
#      S11        S21        S12        S22      Fmin.  âopt   RN/50
#GHZ  mag.  ang.   mag.  ang.   mag.  ang.  mag.   ang.  (dB)  mag.  ang.
Z----------------------------------------------------------------
 0.5 0.68  -126   7.18  106   0.08   35  0.51   -53   1.2  0.36   69.0  0.75
 1.0 0.66  -163   4.02   81   0.09   27  0.34   -66   1.5  0.31  124.0  0.62
 1.5 0.65   178   2.75   64   0.10   27  0.31   -74   2.0  0.50  165.0  0.60
 2.0 0.65   163   2.10   52   0.12   30  0.31   -83   2.4  0.44 -175.0  0.55
 2.5 0.66   151   1.68   39   0.13   26  0.31   -95   2.6  0.52 -161.0  0.52
 3.0 0.66   141   1.46   27   0.14   26  0.33  -106   3.1  0.68 -141.0  0.50
 3.5 0.67   129   1.24   17   0.16   26  0.36  -116   3.3  0.71 -139.0  0.47
```

6.2.1.2 Transistorun Gürültü ve S-parametrelerini İçeren YSA Verileri

Bu transistorda verilmeyen gürültü parametresi bulunmamaktadır. Bu nedenle YSA'dan elde edilen çıktı yukarıdaki dosya ile aynı olacaktır.

6.2.2 Performans Analizi Sonuçları

Yukarıda verilen transistorun S-parametreleri ve gürültü parametreleri dosyası da göz önüne alınmak üzere, performans analiz programına aşağıdaki giriş dosyasının verilmesi durumunda şu sonuçlar alınacaktır.

6.2.2.1 Giriş Dosyası (ne02135.inp):

G_{Treq} V_{ireq} F_{req}

5.0 1.5 4.5

6.2.2.2 Daire Kesişimlerini İçeren Çıktı (out_int.txt):

Transistor: ne02135(VCE=10V, IC=5mA)

Detailed information,Terminations for GTmax and all of intersection points on T1/T2 for GTreq:

*** 1. Given Values:

Required GT(dB) value: 5.000000

Required Vi value: 1.500000

Required F(dB) value: 4.500000

Circuit Definition:

f(Hz): 500000000.000000

S11 Module and Angle(Rad): 0.680000 -2.199115

S21 Module and Angle(Rad): 7.180000 1.850049

S12 Module and Angle(Rad): 0.080000 0.610865

S22 Module and Angle(Rad): 0.510000 -0.925025

Fmin 1.318257, GamaOpt Module and Angle(Rad) 0.360000 1.204277, Rn 0.750000

Z11 Real and Imaginer parts: 0.205424 0.086821

Z21 Real and Imaginer parts: 1.934124 10.619447

Z12 Real and Imaginer parts: 0.118892 0.018146

Z22 Real and Imaginer parts: 1.252796 -0.191766

Freq,Fmin,Ropt,Xopt,Rn,Rmod,N,Rcn,Xcn,rn,Rct1,Xct1,rt1,Rct2,Xct2,rt2: 2.818383 1.318257 0.998652 0.771222 0.750000 0.447214 1.592223 2.590875 0.771222 2.390675 6.559167 -0.771222 6.482698 1.213456 -0.771222 0.689326

Conditionally stable: R11>0 & R22>0 & ita>0 & ita<1

R11,R22,ita = 0.205424,1.252796,0.367784

GTDBreq, T1 and T2 circles:

RCGreq= 0.167737, XCGreq= -0.431087, rgreq= 0.510170

Rct1= 6.559167, Xct1= -0.771222, rt1= 6.482698

Rct2= 1.213456, Xct2= -0.771222, rt2= 0.689326

1. Intersection point on T1; RintT1= 0.132314: XintT1= 0.077852

2. Intersection point on T1; RintT1= 0.078498: XintT1= -0.933392

ZireqT11 Value: real and imaginer part = 6.615699 3.892596

ZireqT12 Value: real and imaginer part = 3.924913 -46.669584

1. Intersection point on T2; RintT1= 0.670788: XintT2= -0.346155

2. Intersection point on T2; RintT2= 0.524565: XintT2= -0.795707

ZireqT21 Value: real and imaginer part = 33.539409 -17.307746

ZireqT22 Value: real and imaginer part = 26.228249 -39.785342

GTreq cuts both T1 and T2

ZCGmax(-0.190557 -0.431087), ZCGmin(0.190557 -0.431087), Zct1(6.559167 -0.771222), Zct2(1.213456 -0.771222)

rs 0.518121, rt1 6.482698, rt2 0.689326

(abs(ZCGmax-Zct2) > (rs+rt2)) & ((rs+rt1) > abs(ZCGmax-Zct1)): Case c) T1 cuts CSSC

Zi_1 Value: real and imaginer part = 6.026169 -0.837278

Zi_2 Value: real and imaginer part = 3.864695 -43.730107

ZL_1 Value: real and imaginer part = 320.866335 305.991235

ZL_2 Value: real and imaginer part = 3.922880 16.521902

ZS_1 Value: real and imaginer part = 15.469020 2.850111

ZS_2 Value: real and imaginer part = 10.114137 43.549778

GTMAX Value (dB) = 17.227476

GTMIN Value (dB) = 0.000000

T1 Intersection:

Zireq1 Value: real and imaginer part = 6.615699 3.892596

Zireq2 Value: real and imaginer part = 3.924913 -46.669584

ZLreq1 Value: real and imaginer part = 69.718519 880.821820

ZLreq2 Value: real and imaginer part = 0.212322 15.582143

ZSreq11 Value: real and imaginer part = 16.894143 -1.418703

ZSreq12 Value: real and imaginer part = 16.894142 -1.418703

ZSreq21 Value: real and imaginer part = 10.266149 46.382455

ZSreq22 Value: real and imaginer part = 10.266153 46.382455

T2 Intersection:

Zireq1 Value: real and imaginer part = 33.539409 -17.307746

Zireq2 Value: real and imaginer part = 26.228249 -39.785342

ZLreq1 Value: real and imaginer part = 4.746323 -67.140317

ZLreq2 Value: real and imaginer part = 1.702768 -15.789829

ZSreq11 Value: real and imaginer part = 14.091207 7.592901

ZSreq12 Value: real and imaginer part = 14.091207 7.592901

ZSreq21 Value: real and imaginer part = 10.021000 40.183300

ZSreq22 Value: real and imaginer part = 10.021000 40.183300

Circuit Definition:

f(Hz): 1000000000.000000

Bu sonuçlar her frekans için bu şekilde verilmektedir. Bu bilgilerin özeti aşağıda verilmiş olduğundan burada verilmeyecektir.

==========

6.2.2.3 Bölge-3'de Elde Edilen Çözümlerini İçeren Çıktı (out_oth.txt):

==========

Transistor: ne02135(VCE=10V, IC=5mA)

Terminations that give GTreq in Region-3:

*** 1. Given Values:

Required GT(dB) value: 5.000000

Required Vi value: 1.500000

Required F(dB) value: 4.500000

f(Hz): 500000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts T1 and T2

Zireq1 Value: real and imaginer part = 7.124059 3.922883

Zireq2 Value: real and imaginer part = 4.427187 -46.753654

ZLreq1 Value: real and imaginer part = 101.029430 1018.665742

ZLreq2 Value: real and imaginer part = 0.239270 14.957510

ZSreq11 Value: real and imaginer part = 18.470423 -5.609419

ZSreq12 Value: real and imaginer part = 15.637626 2.315911

ZSreq21 Value: real and imaginer part = 10.539792 49.802866

ZSreq22 Value: real and imaginer part = 10.100581 43.213913

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq71 Value: real and imaginer part = 24.081712 -1.445747 (Uydurma Devresi için Şeçildi)

Zireq72 Value: real and imaginer part = 21.858855 -43.215133

ZLreq1 Value: real and imaginer part = 15.352405 -192.637560

ZLreq2 Value: real and imaginer part = 1.304135 -7.950703

ZSreq11 Value: real and imaginer part = 27.783947 -24.154614

ZSreq12 Value: real and imaginer part = 12.488883 14.343809

ZSreq21 Value: real and imaginer part = 11.065760 54.413076

ZSreq22 Value: real and imaginer part = 10.091270 34.153812

f(Hz): 1000000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T1

Zireq1 Value: real and imaginer part = 5.914338 6.781235

Zireq2 Value: real and imaginer part = 3.821937 -18.192287

ZLreq1 Value: real and imaginer part = 34.112323 322.714726

ZLreq2 Value: real and imaginer part = 0.768095 19.655272

ZSreq11 Value: real and imaginer part = 15.410351 -7.764976

ZSreq12 Value: real and imaginer part = 12.250575 -1.097387

ZSreq21 Value: real and imaginer part = 8.803588 21.163803

ZSreq22 Value: real and imaginer part = 8.762308 15.178699

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq63 Value: real and imaginer part = 20.240513 -2.378787 (Uydurma Devresi İçin Seçildi)

Zireq64 Value: real and imaginer part = 19.472619 -11.543870

ZLreq1 Value: real and imaginer part = 15.463747 -80.202768

ZLreq2 Value: real and imaginer part = 6.340034 -29.178726

ZSreq11 Value: real and imaginer part = 24.394705 -19.450186

ZSreq12 Value: real and imaginer part = 9.218984 10.448660

ZSreq21 Value: real and imaginer part = 13.061479 -3.058818

ZSreq22 Value: real and imaginer part = 8.706312 18.866528

f(Hz): 1500000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T1

Zireq1 Value: real and imaginer part = 5.969054 10.480147

Zireq2 Value: real and imaginer part = 3.247933 -6.262795

ZLreq1 Value: real and imaginer part = 29.720183 197.162584

ZLreq2 Value: real and imaginer part = 1.394293 26.974717

ZSreq11 Value: real and imaginer part = 15.235887 -8.228561

ZSreq12 Value: real and imaginer part = 10.475890 -3.982487

ZSreq21 Value: real and imaginer part = 7.461693 8.808243

ZSreq22 Value: real and imaginer part = 7.343609 3.602533

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq51 Value: real and imaginer part = 16.714434 2.657455 (Uydurm Devresi İçin Seçildi)

Zireq52 Value: real and imaginer part = 15.917573 -2.245593

ZLreq1 Value: real and imaginer part = 26.397544 -62.113348

ZLreq2 Value: real and imaginer part = 12.667900 -25.884037

ZSreq11 Value: real and imaginer part = 39.473484 9.250634

ZSreq12 Value: real and imaginer part = 7.379234 3.358671

ZSreq21 Value: real and imaginer part = 38.399008 12.531706

ZSreq22 Value: real and imaginer part = 7.583639 9.405109

f(Hz): 2000000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T1

Zireq1 Value: real and imaginer part = 5.128699 14.742329

Zireq2 Value: real and imaginer part = 3.684878 0.589843

ZLreq1 Value: real and imaginer part = 26.072928 144.813571

ZLreq2 Value: real and imaginer part = 2.570328 28.566034

ZSreq11 Value: real and imaginer part = 13.427023 -14.711042

ZSreq12 Value: real and imaginer part = 9.919123 -9.458010

ZSreq21 Value: real and imaginer part = 8.709002 2.027365

ZSreq22 Value: real and imaginer part = 8.380303 -3.561942

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq43 Value: real and imaginer part = 14.650613 8.565073 (Uydurma Devresi İçin Seçildi)

Zireq44 Value: real and imaginer part = 14.258039 4.717022

ZLreq1 Value: real and imaginer part = 42.159440 -46.705890

ZLreq2 Value: real and imaginer part = 20.850529 -18.777862

ZSreq11 Value: real and imaginer part = 34.694251 -18.887117

ZSreq12 Value: real and imaginer part = 8.479262 0.716134

ZSreq21 Value: real and imaginer part = 24.417278 -20.367336

ZSreq22 Value: real and imaginer part = 10.483869 6.912039

f(Hz): 2500000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T1

Zireq1 Value: real and imaginer part = 4.255002 17.653392 (Uydurma Devresi İçin Seçildi)

Zireq2 Value: real and imaginer part = 3.160222 7.919544

ZLreq1 Value: real and imaginer part = 13.659414 87.540666

ZLreq2 Value: real and imaginer part = 3.209379 30.671513

ZSreq11 Value: real and imaginer part = 11.125449 -18.021869

ZSreq12 Value: real and imaginer part = 8.053076 -13.199124

ZSreq21 Value: real and imaginer part = 7.237888 -10.418740

ZSreq22 Value: real and imaginer part = 7.119194 -5.307117

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq37 Value: real and imaginer part = 12.430293 12.075513

Zireq38 Value: real and imaginer part = 12.370295 11.542065

ZLreq1 Value: real and imaginer part = 44.086392 -17.310860

ZLreq2 Value: real and imaginer part = 38.797152 -14.803898

ZSreq11 Value: real and imaginer part = 29.048135 -21.290944

ZSreq12 Value: real and imaginer part = 7.383241 -3.932841

ZSreq21 Value: real and imaginer part = 27.788552 -21.839150

ZSreq22 Value: real and imaginer part = 7.612779 -3.084030

f(Hz): 3000000000.000000

Calculation of many other points of Zi in Region-3; GTreq cuts only T2

Zireq1 Value: real and imaginer part = 10.851229 18.257430

Zireq2 Value: real and imaginer part = 11.025591 15.748488

ZLreq1 Value: real and imaginer part = 66.486093 13.185500

ZLreq2 Value: real and imaginer part = 37.787100 3.143262

ZSreq11 Value: real and imaginer part = 20.620079 -29.585393

ZSreq12 Value: real and imaginer part = 4.579257 -21.475028

ZSreq21 Value: real and imaginer part = 23.924656 -5.879466

ZSreq22 Value: real and imaginer part = 4.879851 -11.744351

Sunuçlar bu şekilde verilmektedir. Yer kaplaması nedeniyle ara bölge kesilmiştir.

Zireq27 Value: real and imaginer part = 3.751635 20.490746

Zireq28 Value: real and imaginer part = 4.303168 12.554593 (Uydurma Devresi İçin Seçildi)

ZLreq1 Value: real and imaginer part = 9.129973 62.774531

ZLreq2 Value: real and imaginer part = 5.263143 26.771973

ZSreq11 Value: real and imaginer part = 6.248644 -24.638948

ZSreq12 Value: real and imaginer part = 3.790922 -16.719723

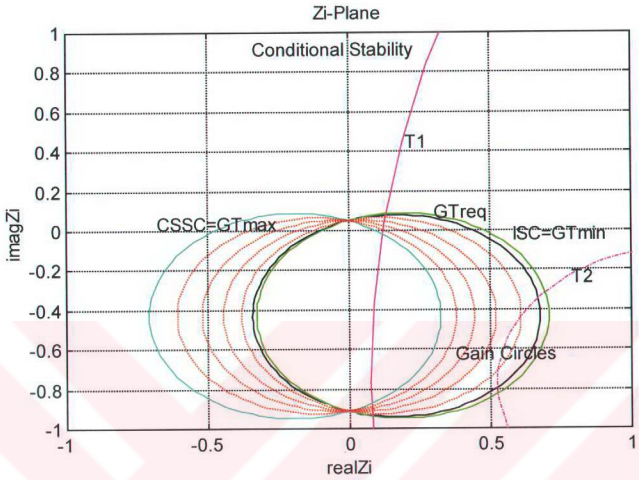
ZSreq21 Value: real and imaginer part = 7.483103 -7.854692

ZSreq22 Value: real and imaginer part = 3.795020 -16.563632

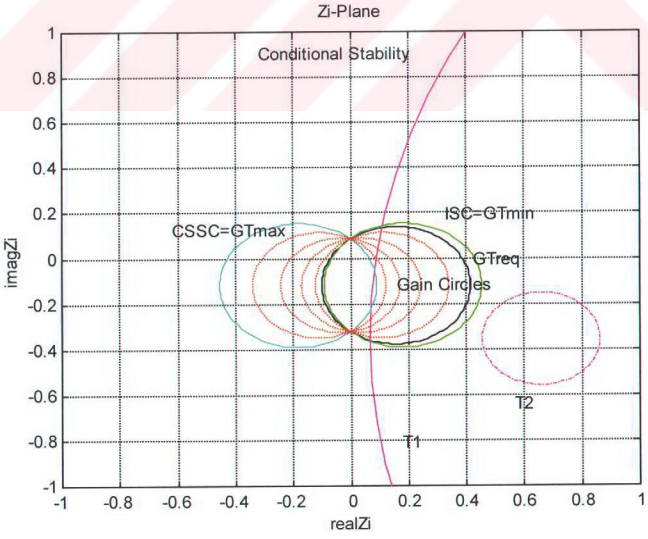
f(Hz): 3500000000.000000

6.2.2.4 Elde Edilen Daireler:

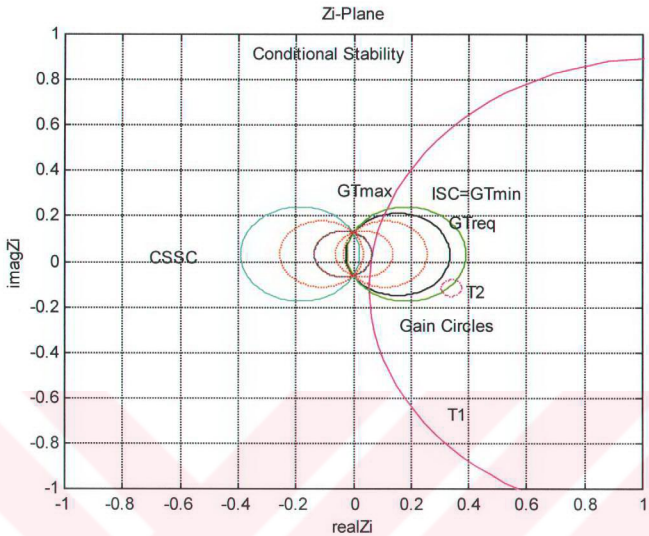
f=0.5GHz: G_{Tmax} G_{Treq} CSSC ISC Gain Circles T1 T2



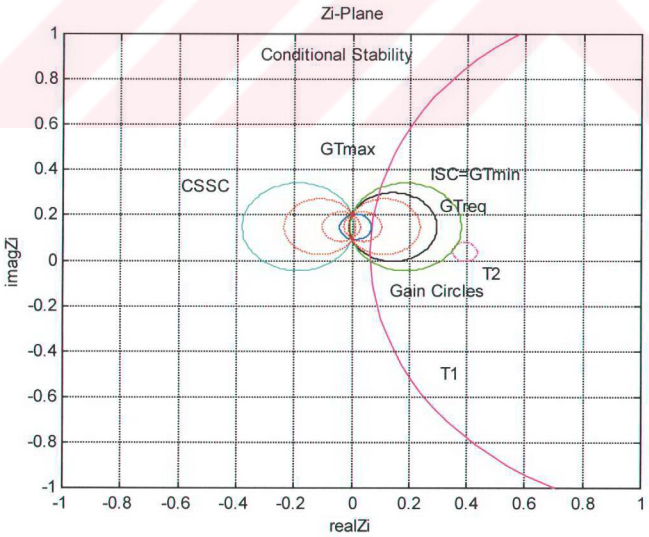
f=1.0GHz:



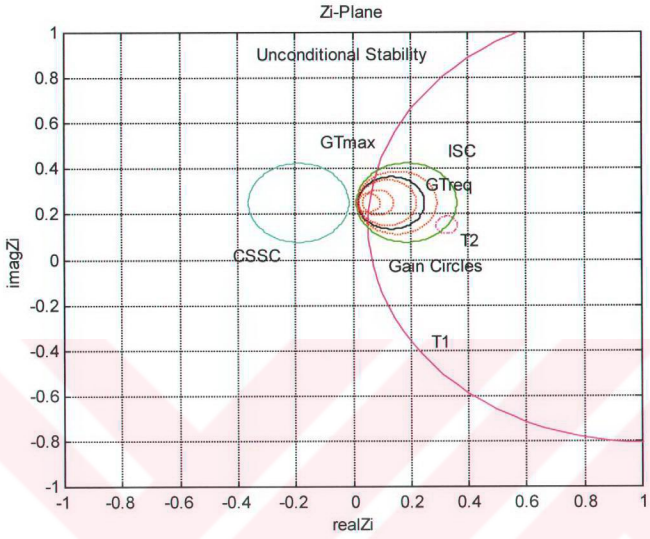
f=1.5GHz:



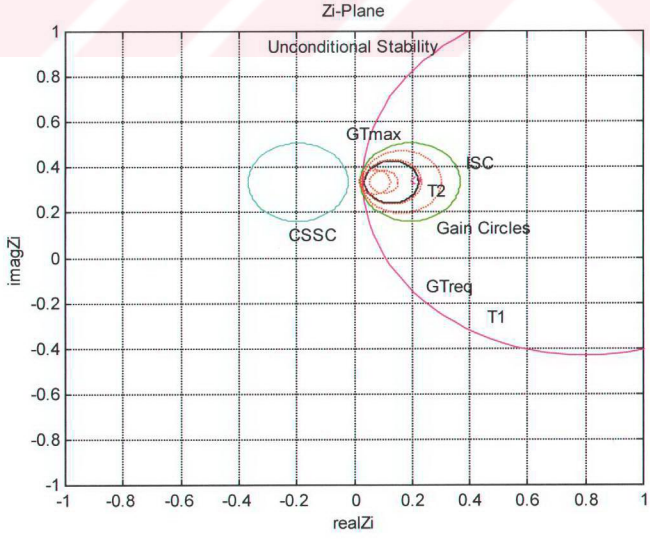
f=2.0GHz:



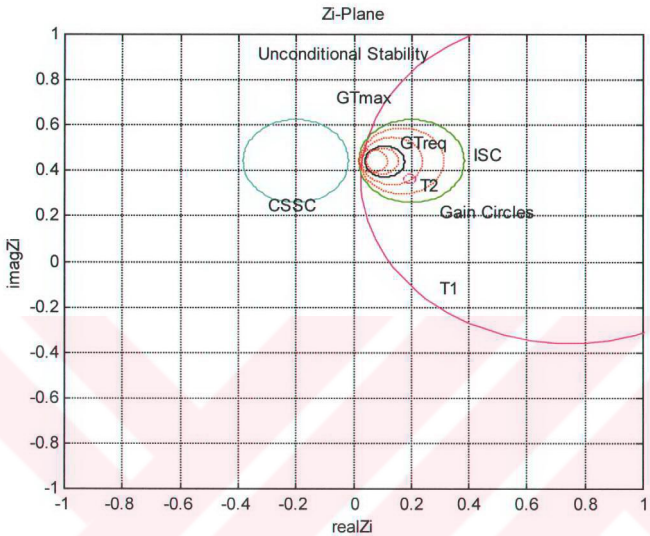
f=2.5GHz:



f=3.0GHz:



f=3.5GHz:



Şekil 6.2.1 Performans Analizi sonucunda farklı frekanslarda elde edilen daireler

6.2.2.5 Daire Kesişimlerinin Özetini İçeren Dosya (out_gra.txt)

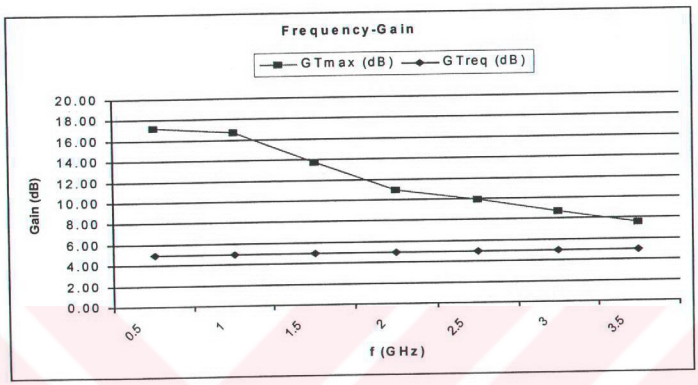
Transistor: ne02135(VCE=10V, IC=5mA)

Terminations for GTmax and One of Intersection points on T1/T2 for GTreq:

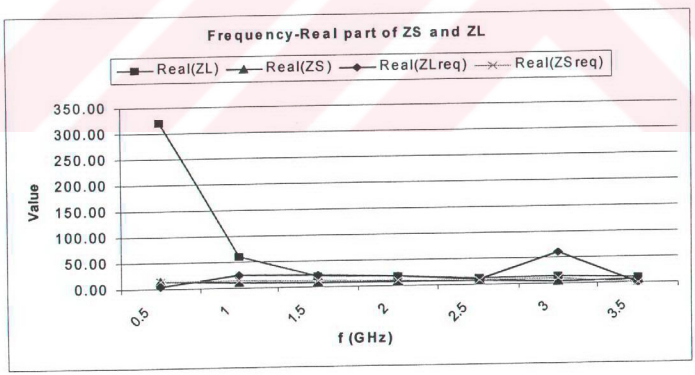
Vireq	Freq(dB)	f(Hz)	GTmax (dB)	Real(ZL)	Imag(ZL)	Real(ZS)	Imag(ZS)	GTreq (dB)	Real(ZLreq)	Imag(ZLreq)	Real(ZSreq)	Imag(ZSreq)
1.5	4.5	500000000	17.227476	320.866335	305.991235	15.469020	2.850111	5.0	4.746323	-67.140317	14.091207	7.592901
1.5	4.5	1000000000	16.763769	60.979805	70.888128	10.641943	3.709178	5.0	25.403403	292.694370	13.552993	-4.148078
1.5	4.5	1500000000	13.695068	21.349069	50.856277	8.162934	0.287619	5.0	22.154672	180.268629	12.254406	-5.978922
1.5	4.5	2000000000	10.974713	19.223872	52.197121	8.824480	-6.241251	5.0	19.708291	134.673635	11.191721	-11.836818
1.5	4.5	2500000000	9.917921	12.873285	47.504815	8.532260	-14.277272	5.0	10.248204	82.090248	9.062400	-15.263588
1.5	4.5	3000000000	8.709726	13.844097	41.187592	4.729928	-12.103118	5.0	59.395393	8.003166	10.476311	-28.428138
1.5	4.5	3500000000	7.550878	11.020453	37.850643	6.754926	-26.676877	5.0	NaN	NaN	NaN	NaN

6.2.2.6 Frekans-Kazanç ve ZL,ZS Değişimi

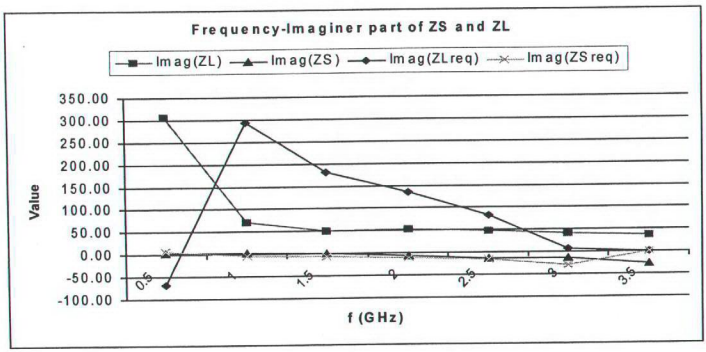
Bu verileri grafiksel olarak ifade edersek aşağıdaki durum elde edilir.



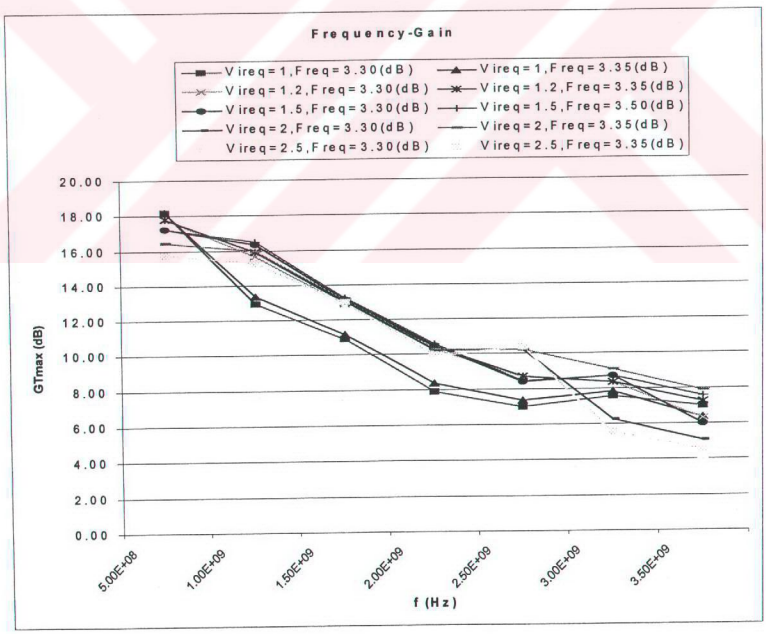
Şekil 6.2.2 Vireq=1.5, Freq=4.5 dB için maksimum Kazanç-Frekans değişimi



Şekil 6.2.3 Vireq=1.5, Freq=4.5 dB için Frekans-Re{ZL/ZS} değişimi



Şekil 6.2.4 Vireq=1.5, Freq=4.5 dB için Frekans-Im{ZL/ZS} değişimi

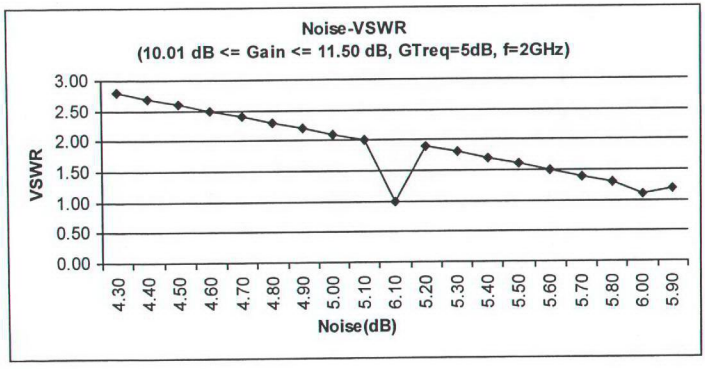


Şekil 6.2.5 Çeşitli Vireq ve Freq'a yönelik maksimum kazanç değişimi

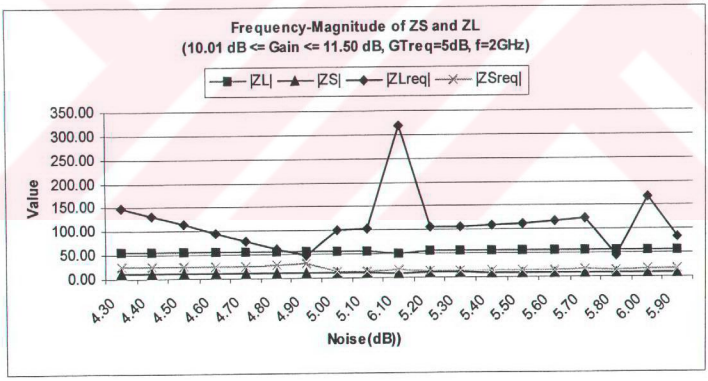
6.2.2.7 Kazanç-VSWR-Gürültü Değişimleri

Çizelge 6.2.1 f=2GHz için Kazanç-VSWR-Gürültü değişimleri

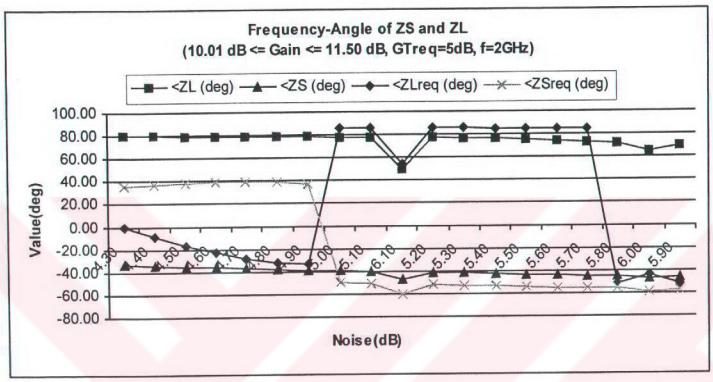
Vireq	Freq (dB)	f (Hz)	Gtmax (dB)	Real (ZL)	Imag (ZL)	Real (ZS)	Imag (ZS)	Gtreq (dB)	Real (ZLreq)	Imag (ZLreq)	Real (ZSreq)	Imag (ZSreq)
2.80	4.30	2.00E+09	10.01	10.13	55.81	9.25	-6.13	5.00	147.09	-1.36	19.53	14.25
2.70	4.40	2.00E+09	10.11	10.27	55.77	9.04	-6.19	5.00	129.73	-20.88	19.47	14.74
2.60	4.50	2.00E+09	10.21	10.42	55.73	8.82	-6.24	5.00	109.05	-32.87	19.52	15.28
2.50	4.60	2.00E+09	10.30	10.59	55.68	8.62	-6.29	5.00	88.35	-37.83	19.78	15.91
2.40	4.70	2.00E+09	10.41	10.80	55.63	8.42	-6.34	5.00	69.55	-37.38	20.39	16.68
2.30	4.80	2.00E+09	10.51	11.03	55.56	8.23	-6.38	5.00	53.23	-33.06	21.73	17.71
2.20	4.90	2.00E+09	10.61	11.29	55.49	8.05	-6.42	5.00	38.65	-25.35	25.14	19.15
2.10	5.00	2.00E+09	10.71	11.60	55.39	7.87	-6.46	5.00	7.59	101.65	9.49	-11.25
2.00	5.10	2.00E+09	10.81	11.96	55.28	7.69	-6.50	5.00	7.84	103.20	9.33	-11.43
1.00	6.10	2.00E+09	10.88	33.56	38.92	6.18	-6.78	5.00	187.29	258.95	8.96	-15.30
1.90	5.20	2.00E+09	10.92	12.38	55.15	7.52	-6.53	5.00	8.16	104.99	9.19	-11.62
1.80	5.30	2.00E+09	11.02	12.87	54.99	7.36	-6.56	5.00	8.58	107.10	9.05	-11.81
1.70	5.40	2.00E+09	11.12	13.45	54.78	7.20	-6.60	5.00	9.12	109.63	8.92	-12.02
1.60	5.50	2.00E+09	11.21	14.16	54.53	7.04	-6.63	5.00	9.85	112.72	8.81	-12.25
1.50	5.60	2.00E+09	11.31	15.03	54.19	6.89	-6.66	5.00	10.87	116.62	8.70	-12.50
1.40	5.70	2.00E+09	11.39	16.13	53.72	6.74	-6.68	5.00	12.36	121.73	8.62	-12.78
1.30	5.80	2.00E+09	11.46	17.58	53.05	6.60	-6.71	5.00	28.66	-35.87	7.82	-11.29
1.10	6.00	2.00E+09	11.47	22.88	49.93	6.32	-6.76	5.00	120.67	-117.39	9.32	-15.56
1.20	5.90	2.00E+09	11.50	19.62	51.98	6.45	-6.73	5.00	51.54	-66.96	8.94	-14.37



Şekil 6.2.6 Çeşitli Vireq ve Freq'a yönelik maksimum kazancın sınırlı kalması durumu



Şekil 6.2.7 Maksimum kazancın sınırlı kalması durumunda sonlandırmaların genlikleri



Şekil 6.2.8 Maksimum kazancın sınırlı kalması durumunda sonlandırmaların açıları

6.2.2.8 Gürültü-VSWR-Kazanç Değişimleri

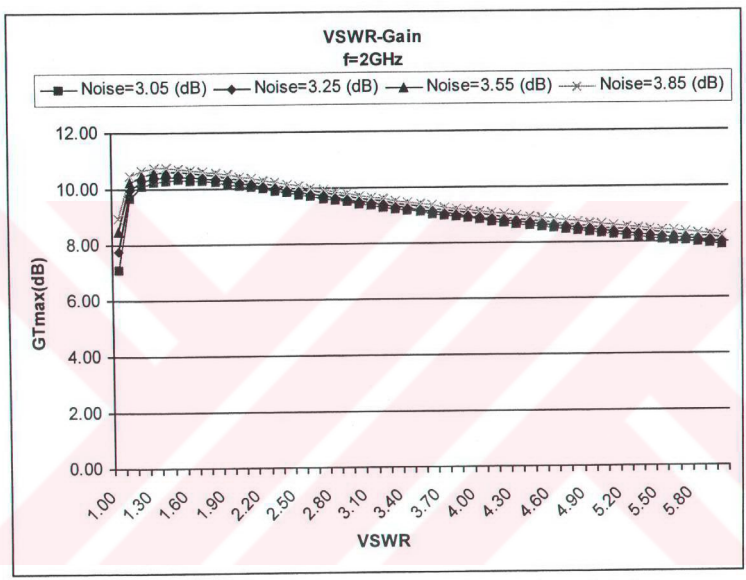
Freq=3.05 dB iken elde edilen Gürültü-VSWR-Kazanç değişimleri aşağıda verilmiştir.

Çizelge 6.2.2 Freq=3.05 dB için Gürültü-VSWR-Kazanç değişimleri

Vireq	Freq (dB)	f (Hz)	Gtmax (dB)	Real (ZL)	Imag (ZL)	Real (ZS)	Imag (ZS)	Gtreq (dB)	Real (ZLreq)	Imag (ZLreq)	Real (ZSreq)	Imag (ZSreq)
1.00	3.05	2.00E+09	7.10	26.66	-12.86	13.05	-4.95	5.00	32.97	-51.78	15.51	-8.55
1.10	3.05	2.00E+09	9.68	39.90	20.68	13.05	-4.95	5.00	245.26	-121.28	16.42	-9.25
1.20	3.05	2.00E+09	10.07	37.20	31.92	13.05	-4.95	5.00	341.20	131.29	16.26	-9.15
1.30	3.05	2.00E+09	10.23	33.96	38.30	13.05	-4.95	5.00	192.49	226.14	16.09	-9.02
1.40	3.05	2.00E+09	10.30	31.03	42.39	13.05	-4.95	5.00	32.67	-38.63	22.30	7.04
1.50	3.05	2.00E+09	10.31	28.50	45.20	13.05	-4.95	5.00	48.03	-52.47	20.97	7.07
1.60	3.05	2.00E+09	10.30	26.35	47.23	13.05	-4.95	5.00	67.39	-63.72	20.45	7.03
1.70	3.05	2.00E+09	10.27	24.49	48.75	13.05	-4.95	5.00	91.98	-71.25	20.21	7.00
1.80	3.05	2.00E+09	10.22	22.89	49.93	13.05	-4.95	5.00	121.86	-72.23	20.12	6.99
1.90	3.05	2.00E+09	10.17	21.50	50.86	13.05	-4.95	5.00	154.91	-62.97	20.10	6.99
2.00	3.05	2.00E+09	10.11	20.27	51.60	13.05	-4.95	5.00	185.97	-40.89	20.13	6.99
2.10	3.05	2.00E+09	10.04	19.19	52.22	13.05	-4.95	5.00	207.95	-7.31	20.18	7.00
2.20	3.05	2.00E+09	9.98	18.23	52.73	13.05	-4.95	5.00	215.79	31.56	20.25	7.01
2.30	3.05	2.00E+09	9.91	17.36	53.15	13.05	-4.95	5.00	209.73	67.83	20.34	7.02
2.40	3.05	2.00E+09	9.84	16.58	53.52	13.05	-4.95	5.00	194.32	96.36	20.43	7.03
2.50	3.05	2.00E+09	9.77	15.88	53.83	13.05	-4.95	5.00	174.86	115.99	20.52	7.04
2.60	3.05	2.00E+09	9.71	15.24	54.10	13.05	-4.95	5.00	155.09	128.06	20.62	7.05
2.70	3.05	2.00E+09	9.64	14.66	54.33	13.05	-4.95	5.00	136.95	134.60	20.72	7.06
2.80	3.05	2.00E+09	9.57	14.12	54.54	13.05	-4.95	5.00	121.10	137.45	20.82	7.06
2.90	3.05	2.00E+09	9.50	13.63	54.72	13.05	-4.95	5.00	107.60	137.95	20.92	7.07
3.00	3.05	2.00E+09	9.44	13.17	54.88	13.05	-4.95	5.00	96.19	137.02	21.02	7.07
3.10	3.05	2.00E+09	9.37	12.75	55.03	13.05	-4.95	5.00	86.59	135.25	21.12	7.08
3.20	3.05	2.00E+09	9.31	12.36	55.16	13.05	-4.95	5.00	78.47	133.01	21.22	7.08
3.30	3.05	2.00E+09	9.25	11.99	55.27	13.05	-4.95	5.00	71.58	130.55	21.31	7.08
3.40	3.05	2.00E+09	9.18	11.65	55.38	13.05	-4.95	5.00	65.69	127.99	21.41	7.08
3.50	3.05	2.00E+09	9.12	11.33	55.47	13.05	-4.95	5.00	60.63	125.44	21.50	7.08
3.60	3.05	2.00E+09	9.06	11.03	55.56	13.05	-4.95	5.00	56.25	122.95	21.60	7.08

3.70	3.05	2.00E+09	9.00	10.75	55.64	13.05	-4.95	5.00	52.43	120.54	21.69	7.08
3.80	3.05	2.00E+09	8.95	10.49	55.71	13.05	-4.95	5.00	49.09	118.23	21.78	7.08
3.90	3.05	2.00E+09	8.89	10.24	55.78	13.05	-4.95	5.00	46.14	116.03	21.87	7.07
4.00	3.05	2.00E+09	8.83	10.00	55.84	13.05	-4.95	5.00	43.52	113.94	21.95	7.07
4.10	3.05	2.00E+09	8.78	9.78	55.90	13.05	-4.95	5.00	41.19	111.95	22.04	7.06
4.20	3.05	2.00E+09	8.73	9.57	55.95	13.05	-4.95	5.00	39.10	110.07	22.12	7.06
4.30	3.05	2.00E+09	8.67	9.37	56.00	13.05	-4.95	5.00	37.22	108.28	22.21	7.05
4.40	3.05	2.00E+09	8.62	9.18	56.04	13.05	-4.95	5.00	35.52	106.59	22.29	7.04
4.50	3.05	2.00E+09	8.57	9.00	56.09	13.05	-4.95	5.00	33.98	104.98	22.37	7.03
4.60	3.05	2.00E+09	8.52	8.83	56.13	13.05	-4.95	5.00	32.57	103.46	22.45	7.03
4.70	3.05	2.00E+09	8.47	8.66	56.16	13.05	-4.95	5.00	31.28	102.01	22.53	7.02
4.80	3.05	2.00E+09	8.42	8.50	56.20	13.05	-4.95	5.00	30.10	100.63	22.60	7.01
4.90	3.05	2.00E+09	8.38	8.35	56.23	13.05	-4.95	5.00	29.02	99.31	22.68	7.00
5.00	3.05	2.00E+09	8.33	8.21	56.26	13.05	-4.95	5.00	28.01	98.06	22.76	6.99
5.10	3.05	2.00E+09	8.28	8.07	56.29	13.05	-4.95	5.00	27.08	96.86	22.83	6.97
5.20	3.05	2.00E+09	8.24	7.94	56.32	13.05	-4.95	5.00	26.22	95.71	22.90	6.96
5.30	3.05	2.00E+09	8.19	7.81	56.34	13.05	-4.95	5.00	25.42	94.62	22.97	6.95
5.40	3.05	2.00E+09	8.15	7.69	56.36	13.05	-4.95	5.00	24.67	93.56	23.04	6.94
5.50	3.05	2.00E+09	8.11	7.58	56.39	13.05	-4.95	5.00	23.97	92.56	23.11	6.92
5.60	3.05	2.00E+09	8.07	7.46	56.41	13.05	-4.95	5.00	23.31	91.59	23.18	6.91
5.70	3.05	2.00E+09	8.03	7.36	56.43	13.05	-4.95	5.00	22.70	90.66	23.25	6.90
5.80	3.05	2.00E+09	7.99	7.25	56.45	13.05	-4.95	5.00	22.12	89.76	23.32	6.88
5.90	3.05	2.00E+09	7.95	7.15	56.47	13.05	-4.95	5.00	21.57	88.90	23.38	6.87
6.00	3.05	2.00E+09	7.91	7.05	56.48	13.05	-4.95	5.00	21.05	88.06	23.45	6.85

Bu durumun grafiksel ifadesi diğer gürültü değerlerindeki değişimlerde göz önüne alındığında aşağıdaki şekilde olmaktadır.



Şekil 6.2.9 f=2GHz için Gürültü-VSWR-Kazanç değişimleri

6.2.2.9 VSWR-Gürültü-Kazanç Değişimleri

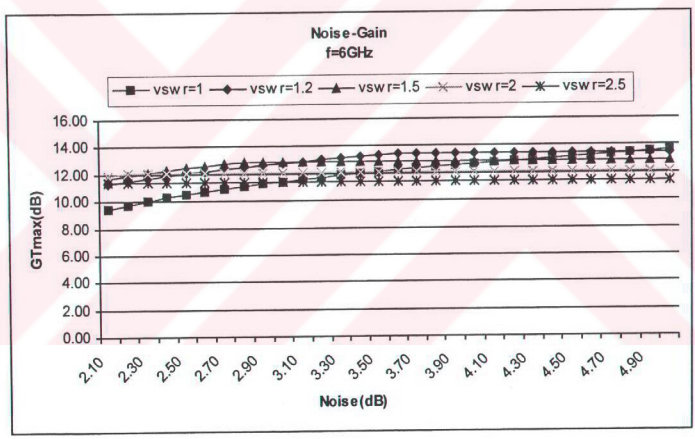
Vireq=1 iken elde edilen VSWR-Gürültü-Kazanç değişimleri aşağıda verilmiştir.

Çizelge 6.2.3 Vireq=1 için VSWR-Gürültü- Kazanç değişimleri

Vireq	Freq	f(Hz)	Gtmax	Real	Imag	Real	Imag	Gtreq	Real	Imag	Real	Imag
	(dB)		(dB)	(ZL)	(ZL)	(ZS)	(ZS)	(dB)	(ZLreq)	(ZLreq)	(ZSreq)	(ZSreq)
1.00	2.10	6.00E+09	9.43	46.22	16.98	11.62	19.61	5.00	26.38	101.21	19.27	9.64
1.00	2.20	6.00E+09	9.77	41.87	19.80	10.92	19.26	5.00	20.96	95.06	18.60	8.48
1.00	2.30	6.00E+09	10.06	38.27	21.85	10.29	18.98	5.00	17.22	90.14	17.89	7.51
1.00	2.40	6.00E+09	10.31	35.25	23.39	9.72	18.73	5.00	14.50	86.13	17.16	6.69
1.00	2.50	6.00E+09	10.54	32.67	24.58	9.20	18.52	5.00	12.45	82.79	16.44	6.00
1.00	2.60	6.00E+09	10.75	30.46	25.51	8.72	18.34	5.00	10.86	79.96	15.73	5.42
1.00	2.70	6.00E+09	10.94	28.54	26.26	8.29	18.19	5.00	9.59	77.54	15.04	4.94
1.00	2.80	6.00E+09	11.11	26.86	26.87	7.88	18.05	5.00	8.55	75.43	14.38	4.54
1.00	2.90	6.00E+09	11.28	25.38	27.38	7.51	17.93	5.00	7.69	73.58	13.75	4.20
1.00	3.00	6.00E+09	11.43	24.06	27.80	7.16	17.83	5.00	6.97	71.95	13.14	3.92
1.00	3.10	6.00E+09	11.58	22.87	28.16	6.84	17.74	5.00	6.35	70.49	12.56	3.69
1.00	3.20	6.00E+09	11.72	21.81	28.47	6.54	17.65	5.00	5.82	69.18	12.01	3.51
1.00	3.30	6.00E+09	11.85	20.84	28.73	6.26	17.58	5.00	5.36	68.00	11.49	3.36
1.00	3.40	6.00E+09	11.98	19.97	28.96	5.99	17.52	5.00	4.96	66.92	10.99	3.23
1.00	3.50	6.00E+09	12.10	19.17	29.16	5.75	17.46	5.00	4.61	65.94	10.53	3.14
1.00	3.60	6.00E+09	12.22	18.43	29.33	5.51	17.40	5.00	4.29	65.04	10.08	3.07
1.00	3.70	6.00E+09	12.34	17.76	29.49	5.29	17.36	5.00	4.01	64.21	9.66	3.01
1.00	3.80	6.00E+09	12.45	17.14	29.62	5.09	17.31	5.00	3.75	63.44	9.26	2.98
1.00	3.90	6.00E+09	12.56	16.56	29.74	4.89	17.27	5.00	3.52	62.73	8.88	2.95
1.00	4.00	6.00E+09	12.67	16.03	29.85	4.71	17.24	5.00	3.32	62.07	8.52	2.94
1.00	4.10	6.00E+09	12.77	15.53	29.95	4.53	17.21	5.00	3.13	61.45	8.18	2.94
1.00	4.20	6.00E+09	12.88	15.07	30.04	4.36	17.18	5.00	2.96	60.87	7.86	2.95
1.00	4.30	6.00E+09	12.98	14.64	30.12	4.21	17.15	5.00	2.80	60.33	7.55	2.97
1.00	4.40	6.00E+09	13.08	14.23	30.20	4.06	17.13	5.00	2.65	59.83	7.26	2.99
1.00	4.50	6.00E+09	13.18	13.85	30.26	3.91	17.10	5.00	2.52	59.35	6.98	3.01
1.00	4.60	6.00E+09	13.27	13.50	30.32	3.78	17.08	5.00	2.39	58.90	6.72	3.05

1.00	4.70	6.00E+09	13.37	13.16	30.38	3.65	17.06	5.00	2.28	58.47	6.47	3.08
1.00	4.80	6.00E+09	13.47	12.84	30.43	3.52	17.04	5.00	2.17	58.07	6.23	3.12
1.00	4.90	6.00E+09	13.56	12.54	30.48	3.41	17.03	5.00	2.07	57.69	6.00	3.16
1.00	5.00	6.00E+09	13.66	12.26	30.52	3.29	17.01	5.00	1.98	57.33	5.79	3.21

Bu durumun grafiksel ifadesi diğer VSWR değerlerindeki değişimlerde göz önüne alındığında aşağıdaki şekilde olmaktadır.



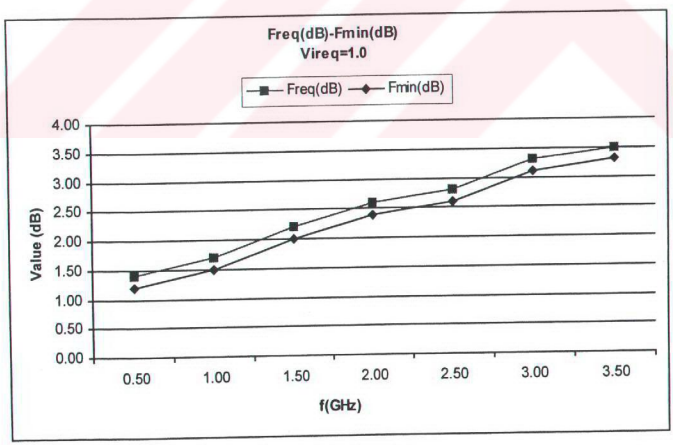
Şekil 6.2.10 f=6GHz için VSWR-Gürültü- Kazanç değişimleri

6.2.2.10 Gürültünün Minimum Gürültüyü İzlemsi Durumundaki Değişimler

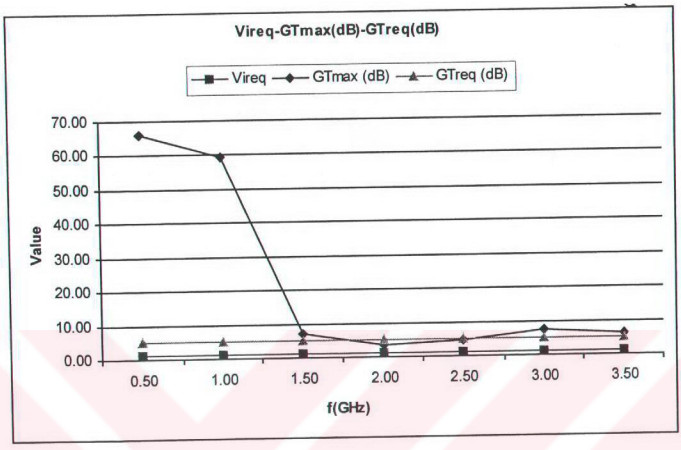
Aşağıda Freq=Fmin+DeltaF olması durumundaki değişimler görülmektedir.

Çizelge 6.2.4 Vireq=1 için Freq=Fmin+DeltaF değişimleri

Vireq	Freq (dB)	f (GHz)	Gtmax (dB)	Real (ZL)	Imag (ZL)	Real (ZS)	Imag (ZS)	Gtreq (dB)	Real (ZLreq)	Imag (ZLreq)	Real (ZSreq)	Imag (ZSreq)
1.00	1.40	0.50	66.02	NaN	NaN	NaN	NaN	5.00	NaN	NaN	NaN	NaN
1.00	1.70	1.00	59.33	NaN	NaN	NaN	NaN	5.00	NaN	NaN	NaN	NaN
1.00	2.20	1.50	6.93	12.50	-15.09	14.36	3.63	5.00	10.26	-27.98	16.50	2.24
1.00	2.60	2.00	3.63	11.09	-21.45	15.74	-3.84	5.00	NaN	NaN	NaN	NaN
1.00	2.80	2.50	4.82	20.59	-13.25	13.03	-9.45	5.00	NaN	NaN	NaN	NaN
1.00	3.30	3.00	7.55	29.80	33.86	7.31	-16.80	5.00	110.96	61.89	10.56	-21.03
1.00	3.50	3.50	6.38	18.93	25.80	7.05	-20.54	5.00	43.62	9.15	11.05	-22.31



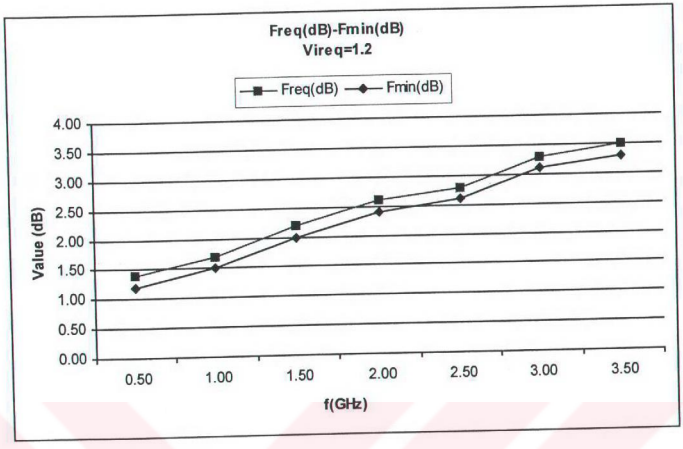
Şekil 6.2.11 Vireq=1 için Freq-Fmin değişimi



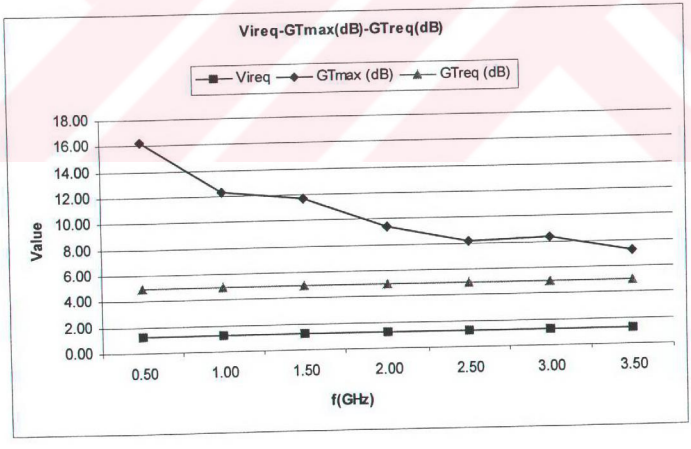
Şekil 6.2.12 Vireq=1 için Freq-Fmin'e karşılık gelen Kazanç değişimleri

Çizelge 6.2.5 Vireq=1.2 için Freq=Fmin+DeltaF değişimleri

Vireq	Freq (dB)	f (GHz)	Gtmax (dB)	Real (ZL)	Imag (ZL)	Real(ZS)	Imag (ZS)	GTreq (dB)	Real (ZLreq)	Imag (ZLreq)	Real (ZSreq)	Imag (ZSreq)
1.20	1.40	0.50	16.28	31.99	-22.91	36.15	31.61	5.00	8.05	-126.11	44.05	22.56
1.20	1.70	1.00	12.29	33.26	-17.55	24.44	13.26	5.00	23.74	-128.44	28.26	9.48
1.20	2.20	1.50	11.74	37.36	18.50	14.36	3.63	5.00	17.97	-49.38	15.78	9.20
1.20	2.60	2.00	9.39	40.02	18.95	15.74	-3.84	5.00	26.92	-36.00	21.61	2.63
1.20	2.80	2.50	8.17	31.61	29.93	13.03	-9.45	5.00	39.59	-23.81	17.61	-3.45
1.20	3.30	3.00	8.35	13.84	41.19	7.80	-15.02	5.00	93.54	19.85	8.98	-13.66
1.20	3.50	3.50	7.25	11.02	37.85	7.64	-21.21	5.00	23.83	13.15	8.49	-14.34



Şekil 6.2.13 Vireq=1.2 için Freq-Fmin değişimi



Şekil 6.2.14 Vireq=1.2 için Freq-Fmin'e karşılık gelen Kazanç değişimleri

6.2.3 Empedans Verisi Uydurma Sonuçları

Önceki kısımlarda bulunan empedans verileri ile uydurma devrelerini gerçekleştirmek üzere (Bazan Tolga, 2000) ve (Güneş Filiz, Yarman B. Sıddık, 1999) tarafından yapılan çalışma sonuçları aşağıda verilmiştir. Aynı analiz Bölüm 6.1'de de yapıldığından burada açıklamalara girilmeyecektir.

Çalışma Bandı: 0.5-3 GHz, GTreq=5 dB, Vireq=1.5, Freq=4.5 dB

Daha önceki kısımda bulunmuş olan verilerden bazıları aşağıda tablo halinde gösterilmiştir.

Çizelge 6.2.6 Uydurma Devresi için kullanılan eleman değerleri

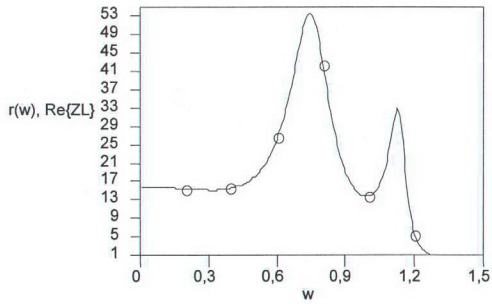
Frekans (GHz)	Z_L (Reel - İmajiner)	Z_S (Reel –İmajiner)
0.5	15.352-192.245j	27.783-24.15j
1	15.463-80.20j	24.394-19.45j
1.5	26.397-62.11j	39.478+9.25j
2	42.159-46.7j	34.69-18.88j
2.5	13.659+87.54j	8.05-13.199j
3	5.26+26.7719j	7.483-7.8549j

R_0=1, $w_0=2\pi f_{max}/1.2$=15.7e9 alınarak 4. Bölümdeki formüllerle elemanların gerçek değerleri F, H, Ohm olarak bulunabilir.

ZL Analizi: Minimum reaktans fonksiyonunun reel kısmı aşağıdaki şekildedir.

$$r(w) := \frac{1.25257}{.0808961 - 3.741410^{-3}\cdot w^2 + .773991\,w^4 - 6.73569\,w^6 + 14.8442\,w^8 - 12.4472\,w^{10} + 3.57922\,w^{12}}$$

r(w)'nın w ile değişimi sürekli eğri ile aşağıda gösterilmektedir.



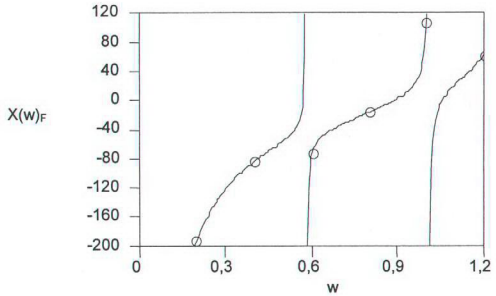
Şekil 6.2.15 r(w)/Re{ZL}-w değişimi

Minimum reaktans fonksiyonunun reel kısmı yardımıyla Gewertz Prosedürü kullanılarak minimum reaktans fonksiyonunun kendisi kolaylıkla elde edilir.

$$z(s)_{MR} = \frac{4.404 + 19.26s + 41.995s^2 + 58.696s^3 + 35.275s^4 + 32.653s^5}{.28442 + 1.21662s + 2.60864s^2 + 3.50573s^3 + 4.39364s^4 + 2.04384s^5 + 1.89188s^6}$$

$$z(w)_{MR} = \frac{\left(4.404 + 19.26iw - 41.995w^2 - 58.696iw^3 + 35.275w^4 + 32.653iw^5\right)}{\left(.28442 + 1.21662iw - 2.60864w^2 - 3.50573iw^3 + 4.39364w^4 + 2.04384iw^5 - 1.89188w^6\right)}$$

Sırada Foster fonksiyonunu bulmak var. $x_F(w_i) = X_D(w_i) - x_{MR}(w_i)$ eşitliğinden gidilerek Foster fonksiyonu bulunur.

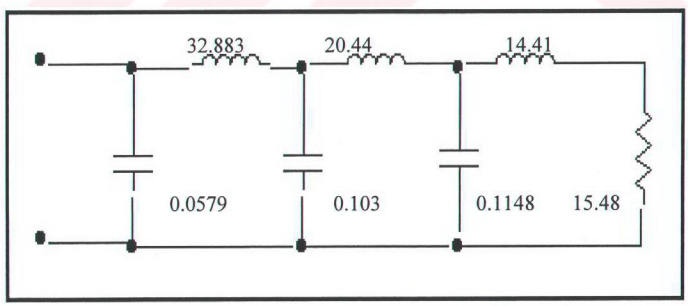


Şekil 6.2.16 ZL için Foster verileri-w değişimi

$$X(w)_F = \frac{39.9807}{w} + 5.24808w + \frac{1.18159w}{(.58^2 - w^2)} + \frac{1.93229w}{(1.01^2 - w^2)} + \frac{51.0013w}{(1.442^2 - w^2)}$$

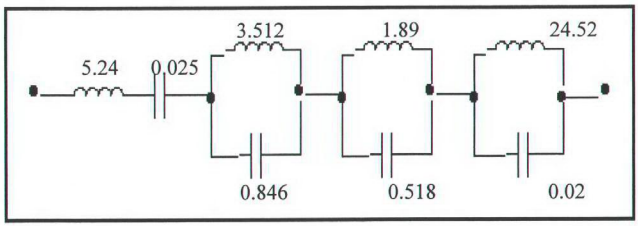
Sonuç olarak minimum reaktans fonksiyonunun ve Foster fonksiyonunun toplamı bize giriş empedansını verir. Devreyi çizerken Foster ve minimum reaktans fonksiyonunu ayrı ayrı çizmek ve sentez etmek bize kolaylık sağlayacaktır.

Aşağıda minimum reaktans fonksiyonunun devresi vardır



Şekil 6.2.17 ZL için minimum reaktans fonksiyonunun devresi

Aşağıda Foster Fonksiyonunun devresi görülmektedir.

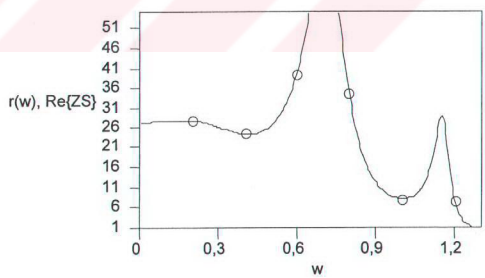


Şekil 6.2.18 ZL için Foster Fonksiyonunun devresi

ZS Analizi: Minimum reaktans fonksiyonunun reel kısmı aşağıdaki şekildedir

$$r(w) := \frac{4.40717}{.161944 - .317141w^2 + 7.32084w^4 - 39.6625w^6 + 77.2838w^8 - 61.0326w^{10} + 16.7928w^{12}}$$

r(w)'nın w ile değişimi aşağıdaki sürekli eğri şeklindedir.



Şekil 6.2.19 r(w)/Re{ZS}-w değişimi

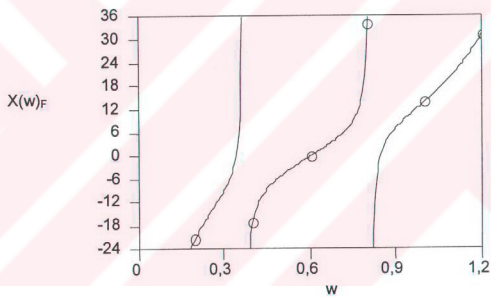
Minimum reaktans fonksiyonunun reel kısmı yardımıyla Gewertz Prosedürü kullanılarak minimum reaktans fonksiyonunun kendisi kolaylıkla elde edilir.

$$z(s)=\frac{10.952+48.021\cdot s+94.318s^2+155.433s^3+74.175s^4+88.721\cdot s^5}{.40242+1.84878s+4.64081\cdot s^2+5.77745s^3+8.87896s^4+3.42602s^5+4.0979s^6}$$

$$z(w)=\frac{\left(10.952+48.021iw-94.318w^2-155.433iw^3+74.175w^4+88.721iw^5\right)}{\left(.40242+1.84878iw-4.64081\cdot w^2-5.77745iw^3+8.87896w^4+3.42602iw^5-4.0979w^6\right)}$$

$x_F(w_i)=X_D(w_i)-x_{MR}(w_i)$ eşitliğinden gidilerek Foster fonksiyonu bulunur.

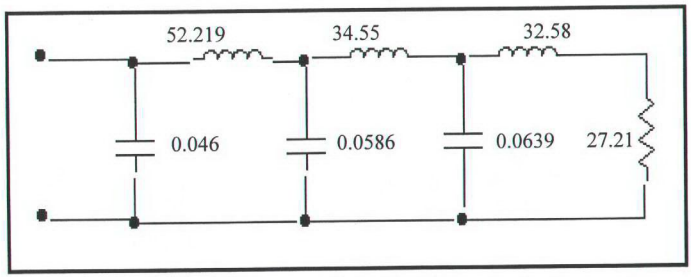
$$X(w)_F=\frac{-5.05837}{w}+\frac{.574429w}{.371^2-w^2}+\frac{.559366w}{.81^2-w^2}+\frac{29.7553w}{1.552^2-w^2}$$



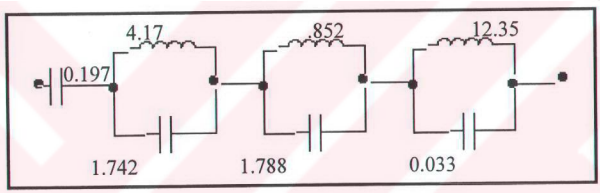
Şekil 6.2.20 ZS için Foster verileri-w değişimi

Sonuç olarak minimum reaktans fonksiyonunun ve Foster fonksiyonunun toplamı bize giriş empedansını verir. Devreyi çizerken Foster ve minimum reaktans fonksiyonunu ayrı ayrı çizmek ve sentez etmek bize kolaylık sağlayacaktır.

Aşağıda minimum reaktans fonksiyonunun devresi vardır



Şekil 6.2.21 ZS için minimum reaktans fonksiyonunun devresi



Şekil 6.2.22 ZS için Foster Fonksiyonunun devresi

SONUÇLAR ve ÖNERİLER

Yapılan bu doktora tez çalışmasında, bir mikrodalga transistorun Yapay Sinir Ağı (YSA) ile performans analiz ve giriş/çıkış uydurman devrelerinin tasarımına yönelik yeni bir yöntem sunulmuştur. Dolayısıyla bu tez; YSA, performans analizi ve uydurma devre tasarımı olmak üzere üç ana bölümden oluşmuştur.

Aktif devre elemanı (transistor) üretici verileriyle, YSA optimum sürede eğitilmektedir. Yapay sinir ağı eğitildikten sonra, klasik kestirim yöntemleri yerine kullanılarak, istenilen veriler kolay bir şekilde ve hızlıca elde edilebilmektedir. Böylelikle önerilen YSA modeli, optimizasyon prosedürü esnasında elemanın fizik denklemlerinin tekrar tekrar çözülmesini gerektirmeyecek ve klasik optimizasyon tekniklerinin ortak sorunu olan "başlangıç değer kestirimi" problemini ortadan kaldırarak tercih edilir duruma gelmektedir. Modelleme esnasında bir döngü içerisinde defalarca optimizasyona ihtiyaç duyulduğu düşünüldüğünde, önerilen yöntemin başarısı kolaylıkla ortaya çıkmaktadır.

Önerilen modelde, herhangi bir mikrodalga transistorun S ve gürültü parametreleri yapay sinir ağı çıkışı, frekans, kutuplama akımı ve gerilimi ile konfigürasyon tipi yapay sinir ağı girişini oluşturmaktadırlar. Böylelikle YSA, frekans, kutuplama ve konfigürasyon tipi parametreleriyle, S ve gürültü parametreleri arasında bir eşleşme yaratarak; eleman denklemleri çözülmeden optimizasyon işlemlerini yapma olanağı sağlamaktadır. Bu çalışmada, YSA üretici tarafından verilmeyen gürültü parametrelerini kestirebilmek için kullanılmıştır.

Transistorun performans analizinde kullanılan ve GÜNEŞ metoduna dayanan optimizasyon problemi, $F_{req} - F(R_S, X_S) = 0$ ve $V_{ireq} - V_i(R_S, X_S, R_L, X_L) = 0$ olmak üzere ($G_{T\max}$, maksimum kazanç ve G_{Treq} istenilen kazanç), $G_{T\max} - G_T(R_S, X_S, R_L, X_L) = 0$ ve ayrıca $G_{Treq} - G_T(R_S, X_S, R_L, X_L) = 0$ olacak şekilde iki ayrı durum için kararlı bölgede kalmak koşuluyla mümkün tüm $Z_S = R_S + jX_S$ ve $Z_L = R_L + jX_L$ kompleks sonlandırmalarını geometriksel ve matematiksel olarak belirlemek şeklinde ifade edilebilir. Bu durumda bu işlemler sonucunda; $\{F_{req}, V_{ireq}, G_{T\max}[f, V_{CE}, I_C]\} \Leftrightarrow Z_{L\max}[f, V_{CE}, I_C]$, $Z_{S\max}[f, V_{CE}, I_C]$ ve $\{F_{req}, V_{ireq}, G_{T\min} \le G_{Treq} \le G_{T\max}[f, V_{CE}, I_C]\} \Leftrightarrow Z_{Lreq}[f, V_{CE}, I_C]$, $Z_{Sreq}[f, V_{CE}, I_C]$ fonksiyonlarına ulaşılmış olacaktır. Burada f; frekansı ve V_{CE} ile I_C (veya V_{DS} ile I_{DS}) kutuplama noktasını dolayısıyla transistorun gürültü ve S-parametrelerini ifade etmektedir.

Bu tür yüksek dereceli polinomların içerildiği optimizasyon problemlerin çözümüne yönelik

bir takım yöntemler literatürde sıkça yer almaktadır ancak tanımladığımız problemin hem reel hem de imajiner kısma sahip olması ve ayrıca çok hesaplama zamanına ihtiyaç duymaları önerilen geometriksel yöntemi daha avantajlı duruma getirmektedir. Bunun yanında bu tezde tüm çözümler hem analitik hem`de geometriksel olarak ifade edilmişlerdir. Kullanılan yöntem adımları ilgili bölümlerde açıklanmıştır.

Performans analizi sonucunda elde edilen sonlandırmalara ilişkin giriş/çıkış uydurma devrelerin tasarımına yönelik birçok yöntem kullanılmıştır. Biz burada önceki bölümlerde açıklanan empedans yaklaşımı yöntemini kullandık. Bu yöntemde veriler, pozitif reel bir z(s) empedans fonksiyonu ile modellendiği zaman bu z(s) empedans fonksiyonu sentez edilebilir olmaktadır. Ölçülen verilerdeki reel ve sanal kısımları ayrı ayrı modellemek, Empedans Yaklaşımı ile Modelleme yöntemimizin esasını oluşturur. Reel ve sanal kısımları ayrı ayrı modellemek için, uygulanan empedans yaklaşımına ilişkin adımlar önceki bölümlerde anlatılmıştır.

Bir mikrodalga transistorun YSA ile performans analizi ve modellenmesi konusunda ilk defa bu tezde ortaya konulan noktalar aşağıda sıralanmıştır.

a) Transistorun, klasik küçük-işaret eşdeğer devresi yerine YSA eşdeğeriyle, çalışma koşullarının (kofigürasyon tipi (CT), kutuplama akım (I_C) ve gerilimi (V_{CE}), frekans (f)) fonksiyonu olarak davranışının anında tayini ve bunun uygulamada kullanılması.

b) $F_{req} \geq F_{\min}$, $V_{ireq} \geq 1$ ve $G_{T\min} \leq G_{Treq} \leq G_{T\max}$ koşulları altında herhangi bir $(F_{req},\ V_{ireq},\ G_{Treq})$ uyumlu performans üçlüsü ile birlikte, $G_{T\max}$'ye karşılık gelen $Z_{L\max} = Z_L = R_L + jX_L$, $Z_{S\max} = Z_S = R_S + jX_S$ ve G_{Treq}'a karşılık gelen $Z_{Lreq} = R_{Lreq} + jX_{Lreq}$, $Z_{Sreq} = R_{Sreq} + jX_{Sreq}$ sonlandırmalarının, transistorun çalışma koşullarının (kofigürasyon tipi (CT), kutuplama akım (I) ve gerilimi (V), frekans (f)) fonksiyonu olarak tayin edilip hesaplanması.

c) Uyumlu performans üçlüsünü gerçekleyen kararlı bölgelerde istenilen adım aralığıyla mümkün tüm sonlandırmaların sistematik olarak tayin edilip hesaplanması.

d) Bulunan sonlandırmaların sistematik bir yöntemle (empedans parametreleri yaklaşımı) giriş/çıkış uydurma devreleri tasarımında kullanılması.

e) Bir mikrodalga transistorun için YSA, performans analizi ve giriş/çıkış uydurma devrelerinin elde edilmesi şeklinde bir blok yapının Aktif Mikrodalga devre sentezinde

kullanılmak üzere sunulması.

Tüm çalışmalar bilgisayar programlarıyla yapılmıştır ve programlarla çıktıları ilgili bölümlerde verilmiştir.

Bölüm 6 incelendiğinde ne02135c(V_{CE}=10V, I_C=20mA) ve ne02135(V_{CE}=10V, I_C=5mA) transistorları için yapılan analiz sonuçları görülecektir. Burada transistorlara yönelik YSA çıktıları, performans analiz sonuçları (çalışma aralığı, kararlı bölgeler, gerekli sonlandırmalar,...) verilmiş ve elde edilen sonlandırmalara yönelik uydurma devreleri çizilmiştir.

Bundan sonra yapılacak çalışmalarda, yaratılacak yeniliklerle YSA'nın eğitim sürecini kısaltma ve hatasını azaltma yönünde adımlar atmaya çalışılabilir. Bunun yanında çeşitli veri modelleme yöntemleri geliştirilerek veya kullanılarak daha geniş bant genişliğinde uydurma devreleri tasarlanması hedeflenebilir.

KAYNAKLAR

Arthur E. Bryons, Jr. Mu-Chi-Ho, (1975) "Applied Optimal Control Optimization, Estimation and Control", Hemisphere Publishing Corporation.

Bandler John, Shauhuacher W., (1998) "Circuit Optimization: The State of the Art, "IEEE Trans.. Microwave Theory Tech, Vol.36. No.2, pp. 424-443.

Bazan Tolga, (2000) "Veri Modellenmesi: Empedans Yaklaşımı Modeli ve Performans Karekteristikleri", YTÜ Bitirme Tezi, 2000 (Danışman: Prof. Dr. Filiz Güneş).

Björn Albinson, (1990) "On Microwave Low Noise Feedback Amplifiers and Control Circuits", Gothenburg.

Blaavw, Gerrit A. ve Brodes, Frederic P. "Computer Architecture Cuncepts and Evolution" Addison Wesley, California.

Carlin H.J. and Yarman S.B., (1983) "The Double Matching Problem Analytic and Real Frequency Solution", IEEE Trans. Circuits and Systems, Vol. CAS-30, No.1, pp.15-28.

Cichocki, A ve Unbehauen R., (1993) "Neural Networks for Optimization and Signal Processing", John Wiley & Sons, New York.

Dobrowolski Janusz A., (1991) "Introduction to Computer Methods for Microwave Circuit Analysis and Design", Artech House, Boston London.

Güneş F., Çetiner B.A., (1998) "Smith chart formulation of performance characterisation for a microwave transistor", IEE Proc. Circuits Devices Syst. Vol.145 No. December 1998.

Güneş Filiz, Güneş M., Fidan M., (1994) "Performance characterization of a microwave transistor", IEE proc- Circuit Devices Syst. Vol.141, No.5, pp.337-344.

Güneş Filiz, Yarman B. Sıddık (1999) "Potential BroadBand Characteristics of a Microwave Transistor and Realisation Conditions", submitted to Piers-2000.

Güneş Macit, (1980) "The Design of Low-Noise Microwave Bipolar Transistor Amplifiers", Ph.D Thesis University of Bradford.

Güneş, F., Gürgen, F., Torpi, H., (1996) "Signal-Noise Neural Network Model for Active Microwave Devices", IEE Circuits Devices Systems Vol.143, No.1. pp.227-234.

Hykin Simon, (1994) "Neural Networks, A Comprehensive Foundation", Macmillan Collage

Publishing, New York.

Kenneth W. Cattermale, (1985) "Mathematical Foundations for Communication Engineering", Pentch Press, London.

Kenneth W. Cattermule, Sohn &.O'Resilly, (1984) "Optimization Methods in Electronics and Communications", Pentech Press, London.

Kılınç Ali, (1995) "Özgün Veri Modelleme Yöntemleri: Empedans ve Saçılma Parametreli Yaklaşımlar", Ph.D Thesis in İÜ (Danışman: Prof. Dr. B. Sıddık Yarman).

McCard Marilyn Nelson and Illigworth W. T., (1991) "A Practical Guide to Neural Nets", Addison-Wesley Publishing Company.

Stephen A. Maas, (1988) "Nonlinear Microwave Circuits", Artech House.

Tepe Cemal, (1991) "Bikuadratik Alt Devrelerin Bilgisayar Yardımıyla Parametre ve Duyarlık Analizi", İTÜ Elektronik ve Haberleşme Bölümü, Bitirme Tezi (Danışman: Prof. Dr. Ali Nur Gönüleren).

Tepe Cemal, (1994) "Numerik Analiz Yöntemleri ve Programlama", İTÜ Fen Bilimleri Enstitüsü, Elektonik ve Haberleşme Bölümü, Master Tezi (Danışman: Prof. Dr. Ali Nur Gönüleren).

Thrun Sebastian, (1996) "Explanation-Based Neural Network Learning", Kluwer Academic Publishers, Boston.

Torpi Hamid, (1997) "Mikrodalga Transistörlerinin Yapay Sinir Ağı Eşdeğerleri", YTÜ Doktora Tezi (Danışman: Prof. Dr. Filiz Güneş).

Veluswami Anand, Nakhla Michel S. and Zhang Qi-Jul, (1997) "The Application of Neural Networks to EM Based Simulation and Optimization of Interconnects. in High-Speed VLSI Circuits. "IEEE Trans. Microwave Theory Tech, Vol.45, No.5, pp.713-723.

Wamg Fang, Zhang Qi-Jun, (1997) "Knowledge-Based Neural Models For Microwave Design", IEEE Trans. Microwave Theory Tech, Vol.43, No.12, pp. 233-2343.

Werbos, P. John, (1994) "The Roots of Back-propagation", John Wiley & Sons. Inc. New York.

Yarman S.B., Akşen A., (1992) "An Integrated Design Tool to Construct Lossless Matching Network with Mixed Lumped and Distributed Elements", IEEE Trans. on Circuits and Syst.–

I: Fundamental Theory and Applications, Vol.39, No.9, pp.713-723.

Yarman S.B., Feltuvies A., (1990) "Computer-Aidede Double Matching via Parametric Representation of Brune Functions", IEEE Trans. Circuits and Systems, Vol. 37, No.2, pp.212-222.

Zaabab A. Hafid, Zhong Qi-Jun, Nakhla Michel S., (1995) "Device and Circuit - Level Modeling Using Neural Networks with Faster Training Based on Network Sparsity", IEEE Trans. Microwave Theory Tech, Vol.43, No.6, pp. 1349-1358.

Zaabab A.Hafid, Zhang Qi-Jun, Nakhla Michel, (1995) "A Neural Network Modeling Approach to Circuit Optimization and Statistical Design", IEEE Trans. Microwave Theory Tech, Vol.43, No.6, pp. 1349-1358.

EKLER

Ek 1 Yapay Sinir Ağı Programı (nnpsn.pas)

```
{/*************************************************************/
/*                                                           */
/*  File: nnpsn.pas                                          */
/*                                                           */
/*  Compiler: Turbo Pascal 6.0                               */
/*                                                           */
/*  Description:Neural Network Program to estimate           */
/*           S and Noise parameters of a Transistor          */
/*                                                           */
/*  Created: Cemal Tepe 1998 Ph.D. Thesis                    */
/*                                                           */
/*  Version: 1.0                                             */
/*                                                           */
/*  Notes:                                                   */
/*                                                           */
/*************************************************************/
}

PROGRAM Multi_Layer_NN;
{$N+}
uses crt,printer,Dos;
CONST
  Layers_Max    = 3;      { Maximum number of layers }
  Nodes_Max     = 15;     { Maximum number of node in any layer }
  Outputs_Max   = 10;     { Number of input combinations (the line number in input.dat) }
  Input_Nodes   = 4;      { Number of input layer nodes }
  Output_Nodes  = 12;     { Number of output layer nodes }
  Hidden_Nodes  = 12;     { Number of hidden layer nodes }
  Epsilon       = 0.5;    { Learning Rate; nu }
  Alpha         = 0.1;
  Iterative_Max = 100000; { Maximum iteration number }
  Every         = 50000;  { After every iteration print results }

  Degree        = 360;  { !!! do not change this value; normalization for angle }


TYPE
  adizi   = array [ 0..outputs_Max-1 ] of real;

VAR
  j:longint;
  f:adizi;
  count:longint;
  samplecount,i,ii,tt,isylla, num,dcount:byte;
  fmax,fmin,step,eps, alp, err, error, terr, terror, total_rate: double;
  Layers, Inputs, Outputs : byte;

  Node     : array [0..Layers_Max-1] of integer;
```

```
  matrix          :  array  [0..Outputs_Max-1,    0..(Input_Nodes+Output_Nodes-1)] of double;
  matrix_tmp: array [0..trunc(Iterative_Max/Every), 0..Outputs_Max-1,
              0..(Input_Nodes+Output_Nodes-1)] of double;
  matrix_tmp_out: array [0..trunc(Iterative_Max/Every), 0..Outputs_Max-1,
              0..(Input_Nodes+Output_Nodes-1)] of double;

  des   : double;
  out   : array [0..Layers_Max-1, 0..Nodes_Max-1] of double;
  der   : array [0..Layers_Max-1, 0..Nodes_Max-1] of double;
  theta  : array [0..Layers_Max-1, 0..Nodes_Max-1] of double;
  dtheta  : array [0..Layers_Max-1, 0..Nodes_Max-1] of double;
  weight  : array [0..Layers_Max-1, 0..Nodes_Max-1, 0..Nodes_Max-1] of double;
  dweight : array [0..Layers_Max-1, 0..Nodes_Max-1, 0..Nodes_Max-1] of double;
  dess   : array [0..Outputs_Max-1,0..output_nodes-1] of double;
  dessout : array [0..Outputs_Max-1,0..output_nodes-1] of double;
  dessin  : array [0..Outputs_Max-1,0..input_nodes-1] of double;
  desstest: array [0..Outputs_Max-1,0..input_nodes-1] of double;

  st1: string[7]; st5,st6: string[6];
  d,st2, st3, st4: string[12];
  idatafile,datafile,graphfile,outfile,outfile_t:text;
  weightfile:text;
  tercih,tercih_test:char;
  beta:real;
  ad,hiz:string[8];
  TransistorName :string[100];
  pass:char;
  gettime_h, gettime_m, gettime_s, gettime_hund : Word;
  S21max,Fminmax:real;

{ CT: Used by gettime }
Function LeadingZero(time_w : Word) : String;
var
   time_s : String;
begin
   Str(time_w:0,time_s);
   if Length(time_s) = 1 then
     time_s := '0' + time_s;
   LeadingZero := time_s;
end;


Procedure KARTEZYEN_TO_POLAR(a1,b1:real;var gk3,ak3:real);
var
   t:real;
begin
    t := b1/a1;
    gk3 := sqrt(sqr(a1)+sqr(b1));
       if a1<=0 then
            begin
              if b1=0 then ak3 := 180;
              if b1<0 then
```

```
                begin
                  if a1=0 then ak3 := -90
                        else ak3 := 180+arctan(t)*180/pi;
                end;
              if b1>0 then
                begin
                  if a1=0 then ak3 := 90
                        else ak3 := 180+arctan(t)*180/pi;
                end
            end
            else ak3 := arctan(t)*180/pi;
end;


{ Reading Normalization Values }
Procedure READ_NORMALIZE;
var
ndosya:text;
k,l:integer;
s:real;
begin
  Assign(ndosya,'c:\cemal\phd\thesis\yeni-tez\nnpas\norm.dat');
  reset(ndosya);

  read(ndosya,num);    { number of line in input.dat }
  read(ndosya,S21max);
  readln(ndosya,Fminmax);
  readln(ndosya,TransistorName);

  close(ndosya);
end;



{ CT: hedef_belle_yeni }
Procedure HEDEF_BELLE_YENI;
var
odosya:text;
k,l:integer;
s:real;
begin
  Assign(odosya,'c:\cemal\phd\thesis\yeni-tez\nnpas\desired.dat');
  reset(odosya);
  for k:=0 to outputs_max-1 do
    for l:= 0 to output_nodes-1 do
      begin
        read(odosya,s);
        dess[k,l]:=s;
      end;
  close(odosya);

  { CT: Normalization }
```

```
  for k:=0 to outputs_max-1 do
   for l:= 0 to output_nodes-1 do
   begin
     case (l) of
          1,3,5,7,10:begin  { degrees }
             if dess[k,l] < 0 then
             begin
               dess[k,l]:=dess[k,l]+Degree;
               dess[k,l]:=dess[k,l]/Degree;
             end
             else
             begin
               dess[k,l]:=dess[k,l]/Degree;
             end;
           end;
          2:dess[k,l]:=dess[k,l]/S21max;
          8:dess[k,l]:=dess[k,l]/Fminmax;
     end;
   end;

end;

{Reading test data; f(GHz) VCE(V) IC(mA) CT  (CT=0.1 --> CE, CT=0.9 --> CC) }
Procedure READ_TEST;
var
tdosya:text;
k,l:integer;
s:real;
begin
  Assign(tdosya,'c:\cemal\phd\thesis\yeni-tez\nnpas\input_t.dat');
  reset(tdosya);
  for k:=0 to Outputs_max-1 do
    for l:= 0 to input_nodes-1 do
      begin
        read(tdosya,s);
        desstest[k,l]:=s;
      end;
  close(tdosya);
end;

{Reading input data for learning; f(GHz) VCE(V) IC(mA) CT  (CT=0.1 --> CE, CT=0.9 --> CC) }
Procedure READ_INPUT;
var
idosya:text;
k,l:integer;
s:real;
begin
  Assign(idosya,'c:\cemal\phd\thesis\yeni-tez\nnpas\input.dat');
  reset(idosya);
  for k:=0 to Outputs_max-1 do
    for l:= 0 to input_nodes-1 do
      begin
```

```
          read(idosya,s);
          dessin[k,l]:=s;
        end;
  close(idosya);
end;


Procedure  WRAND;
{ Initializes node offsets & weights at the beginning of the program }
var
  li, ni, ni_end, no, no_end : integer;
  drand48, power: double;
begin
  { Randomize offsets }
  for li:=1 to Layers_Max-1 do
  begin
    ni_end:=node[li];
    for ni:=0 to ni_end-1 do
    begin
      drand48:=Random(30)/100;
      theta[li,ni]:= drand48;
      dtheta[li,ni]:=0;
    end;
  end;

  { Randomize weights }
  for li:=Layers_Max-1 downto 1 do
  begin
    ni_end:=node[li];
    no_end:=node[li-1];
    for ni:=0 to ni_end-1 do
      for no:=0 to no_end-1 do
      begin
        drand48:=Random(31)/100;
        weight[li,ni,no]:=drand48;
        dweight[li,ni,no]:=0;
      end;
  end;
end;  { procedure WRAND }

Procedure INITIAL;
{ sets zero the output of each node before every iteration }
VAR
  l, n, n_end: integer;
begin
  for l:= 1 to Layers-1 do {Dikkat l:=0 idi}
  begin
    n_end:= node[l];
    for n:= 0 to n_end-1 do
      out[l,n]:= 0;
  end;
```

```
end;   { procedure INITIAL }

Procedure YREAD_TEST ( isylla: integer );
{ read input pattern ( for several different input combination ) }
{   f(GHz) VCE(V) IC(mA) CT  (CT=0.1 --> CE, CT=0.9 --> CC)    }
var
l:integer;
begin
 for l:= 0 to input_nodes-1 do
 begin
  out[0,l]:=desstest[isylla,l];
 end;
end;



Procedure YREAD ( isylla: integer );
{ read input pattern ( for several different input combination ) }
{   f(GHz) VCE(V) IC(mA) CT  (CT=0.1 --> CE, CT=0.9 --> CC)    }
var
l:integer;
begin
 for l:= 0 to input_nodes-1 do
 begin
  out[0,l]:=dessin[isylla,l];
 end;
end; { procedure YREAD }


Function SIGMOID ( x: extended ): extended;

{ sigmoid function }
var
  y: extended;
begin
  Beta:=1;
  if x >= 50000 then y:=1
                else
                begin
                  if x <= -50000 then y:=0
                                 else y:=1/(1+exp(-beta*(x)));
                end;
  sigmoid:=y;
end;  { function SIGMOID }

{ CT: ************************************
FUNCTION TANHIPER ( x: double ): double;

var
  y: double;
begin
  Beta:=1;
```

```
  if x >= 1000000000 then y:=1
                else
                begin
                  if x <= -1000000000 then y:=-1
                              else y:=(exp(beta*(x))-exp(-beta*(x)))/(exp(beta*(x))+exp(-beta*(x)));
                end;
  tanhiper:=y;
end;
***************************** }
{ function TANHIPER }

{ ***************************
FUNCTION SIGMOID ( x: double ): double;

 sigmoid function
var
  y: double;
begin
  y:=1/(1+exp(-(x)));
  sigmoid:=y;
end;   function SIGMOID
******************************** }

Procedure WFORWARD ;
{ wforward propagation }
var
  li, ni, ni_end, no, no_end: integer;
  x, th: double;
begin
  for li:=1 to Layers-1 do
  begin
    ni_end:= node[li];
    no_end:= node[li-1];
    for ni:= 0 to ni_end-1 do
    begin
      x:= 0;
      for no:= 0 to no_end-1 do
      x:= x + weight[li,ni,no] * out[li-1,no];
      out[li,ni]:= sigmoid( x-theta[li,ni] );
    end;
  end;
end;   { procedure FORWARD }

Function BACK: double;
{ back propagation }
var
  li, ni, ni_end, no, no_end: integer;
  e, err, x, d, w, y: double;
begin
  err:=0;
  for ni:=0 to outputs-1 do
  begin
```

```
    y:= out[2,ni];
    des:=dess[isylla,ni];
    e:= des - y;
    der[2,ni]:= e*y*(1-y);
    err:=err+ e*e;
  end;
  for li:=2 downto 1 do
  begin
    no_end:= node[li-1];
    ni_end:= node[li];
    for no:= 0 to no_end-1 do
    begin
      x:= 0;
      y:= out[li-1,no];
      for ni:= 0 to ni_end-1 do
      begin
        d:= der[li,ni];
        w:= weight[li,ni,no];
        x:= x + d*w;
      end;
      der[li-1,no]:=y*(1-y)*x;
    end;
  end;
  back:= 0.5*err;
end; { function BACK }

Procedure LEARNING (alp, eps: double );
{ update offsets & weights }
var
  li, lo, ni, ni_end, no, no_end: integer;
  di, yo, ew, et: double;
begin
  for li:= 1 to layers-1 do
  begin
    ni_end:= node[li];
    for ni:= 0 to ni_end-1 do
    begin
      et:= der[li,ni];
      dtheta[li,ni]:= -eps*et + alp*dtheta[li,ni];
      theta[li,ni]:= theta[li,ni] + dtheta[li,ni];
    end;
  end;
  for li:=1 to layers-1 do
  begin
    lo:= li-1;
    ni_end:= node[li];
    no_end:= node[lo];
    for ni:= 0 to ni_end-1 do
    begin
      di:= der[li,ni];
      for no:= 0 to no_end-1 do
      begin
```

```
          yo:= out[lo,no];
          ew:= di*yo;
          dweight[li,ni,no]:= eps*ew + alp*dweight[li,ni,no];
          weight[li,ni,no]:= weight[li,ni,no] + dweight[li,ni,no];
        end;
      end;
    end;
end;  { procedure LEARNING }

Procedure MAKE_MATRIX( it, isylla: integer );
var
  i,j: integer;
begin
  for i:= 0 to inputs-1 do
  matrix_tmp[it,isylla,i]:= out[0,i];
  for i:= 0 to outputs-1 do
    matrix_tmp[it,isylla,inputs+i]:= out[Layers_Max-1,i];
end;  { procedure MAKE_MATRIX }


Procedure READ_WEIGHTS;
var
  data:string[100];
  li,ni,no,no_end,ni_end: byte;
begin

  assign(weightfile,'c:\cemal\phd\thesis\yeni-tez\nnpas\weight.txt');
  reset(weightfile);
  readln(weightfile,data);
  readln(weightfile,data);
  for li:=Layers_Max-1 downto 1 do
  begin
    ni_end:=node[li];
    no_end:=node[li-1];
    for no:=0 to no_end-1 do
      begin
        for ni:=0 to ni_end-1 do
        begin
          read(weightfile,weight[li,ni,no]);
          read(weightfile,dweight[li,ni,no]);{dweight[li,ni,no]:=0;}
        end;
      end;
  end;
  for li:=1 to layers-1 do
  begin
    ni_end:=node[li];
    for ni:=0 to ni_end-1 do
    begin
      readln(weightfile,theta[li,ni]);
      readln(weightfile,dtheta[li,ni]);
    end;
  end;
```

```
  read(weightfile,alp);read(weightfile,eps);
  close(weightfile);
end;


Procedure WRITE_WEIGHTS;
var
  li,ni,no,no_end,ni_end:byte;
begin
  assign(weightfile,'weight.txt');
  rewrite(weightfile);
  writeln(weightfile,d,' Weights and offsets.',j,' iteration ');
  writeln(weightfile,' Total error ',terror,'.');
  for li:=Layers_Max-1 downto 1 do
  begin
    ni_end:=node[li];
    no_end:=node[li-1];
    for no:=0 to no_end-1 do
      begin
        for ni:=0 to ni_end-1 do
        begin
          write(weightfile,' ',weight[li,ni,no]);
          write(weightfile,' ',dweight[li,ni,no]);
        end;
        writeln(weightfile,' ');
      end;
  end;
  for li:=1 to layers-1 do
  begin
    ni_end:=node[li];
    for ni:=0 to ni_end-1 do
    begin
      writeln(weightfile,theta[li,ni]);
      writeln(weightfile,dtheta[li,ni]);
    end;
  end;
  writeln(weightfile,alp,eps);
  close(weightfile);
end;

BEGIN { main }
    clrscr;
{ CT:
  arrange;
  hedef_belle;
}
    read_normalize;          {read normalization data }
    hedef_belle_yeni;        {read desired data }
    read_input;              {read input values (f(GHz) VCE(V) IC(mA) CT  (CT=0.1 --> CE, CT=0.9 --> CC)) }
    read_test;               {read test velues }

    { print first header }
```

```
    node[2]:= Output_Nodes;
    Outputs:= Output_Nodes;
    node[1]:= Hidden_Nodes;
    node[0]:= Input_Nodes;
    Inputs:= Input_Nodes;
    Layers:= Layers_Max;
{CT: number of input line was read form norm.dat
    num:= Outputs_Max;
}
    alp:=alpha;
    eps:=epsilon;

    write('Stop?: [Y/N] ');
    readln(tercih);tercih:=Upcase(tercih);
    if tercih='Y' then Exit;

    write('Use different data for test (input_t.dat)? [Y/N] ');
    readln(tercih_test);tercih_test:=Upcase(tercih_test);

    { **********************
    INITIALIZE MATRIX
    ********************** }
    for i:= 0 to num-1 do
    for j:= 0 to ( inputs+Outputs-1 ) do
    matrix[i,j]:= 0;
    for ii:= 0 to ( Trunc(Iterative_Max/Every)-1 ) do
    for i:= 0 to num-1 do
    for j:= 0 to ( inputs+Outputs-1 ) do
    matrix_tmp[ii,i,j]:=0;

{ CT: for i:=1 to 24 do
    writeln;
}
    str(epsilon:4:2,st5);str(alpha:4:3,st6);
    assign(outfile,'c:\cemal\phd\thesis\yeni-tez\nnpas\output.txt');
    assign(outfile_t,'c:\cemal\phd\thesis\yeni-tez\nnpas\output_t.txt');
{
    writeln;append(outfile);
}
    rewrite(outfile);
    rewrite(outfile_t);
    writeln(outfile,TransistorName);
    writeln(outfile_t,TransistorName);


    write('Use Available Weights? [Y/N] ');
    readln(tercih);tercih:=Upcase(tercih);

    GetTime(gettime_h,gettime_m,gettime_s,gettime_hund);
    WriteLn('Program Started... ',LeadingZero(gettime_h),':',
     LeadingZero(gettime_m),':',LeadingZero(gettime_s),
     '.',LeadingZero(gettime_hund));
```

```
if tercih='Y' then read_weights else wrand;


j:=0;dcount:=trunc(iterative_max/every)+1;
{**********************************
LEARNING PHASE
********************************** }
while ( j < Iterative_Max ) do
begin
  str ( j:7, st1 );
  writeln('iteration=',st1);
  GetTime(gettime_h,gettime_m,gettime_s,gettime_hund);
   WriteLn('Training Started... ',LeadingZero(gettime_h),':',
   LeadingZero(gettime_m),':',LeadingZero(gettime_s),
   '.',LeadingZero(gettime_hund));

  for count:= 0 to Every-1 do
  begin
    error:= 0;
    for isylla:=0 to num-1 do
    begin
      initial;
      yread(isylla);    { reade input.dat file }
      wforward;
      err:= back;
      learning( alp, eps );
      error:= error+err;
    end;
    error:= error / outputs;
    eps:= 0.5 * sqrt( error );

    Inc( j );
  end;
  GetTime(gettime_h,gettime_m,gettime_s,gettime_hund);
   WriteLn('Training Finished! ',LeadingZero(gettime_h),':',
   LeadingZero(gettime_m),':',LeadingZero(gettime_s),
   '.',LeadingZero(gettime_hund));

  { ***********************************************
  TESTING PHASE
  *********************************************** }
  { CT: test of learning data }
  terror:= 0;
  GetTime(gettime_h,gettime_m,gettime_s,gettime_hund);
   WriteLn('Test Started... ',LeadingZero(gettime_h),':',
   LeadingZero(gettime_m),':',LeadingZero(gettime_s),
   '.',LeadingZero(gettime_hund));
  for isylla:= 0 to num-1 do
  begin                  { read input.dat }
    yread(isylla);
    wforward;
```

```
        terr:= back;
        terror:= terror + terr;
        make_matrix( trunc( j/Every ) - 1, isylla );
      end;
      GetTime(gettime_h,gettime_m,gettime_s,gettime_hund);
       WriteLn('Test Finished! ',LeadingZero(gettime_h),':',
       LeadingZero(gettime_m),':',LeadingZero(gettime_s),
       '.',LeadingZero(gettime_hund));

      write(outfile, '       Input Values                                    Calculated Values ');
      writeln(outfile,'                                                     Desired Values');
      write(outfile, 'Num f(GHz)  VCE(V) IC(mA)  CT    ');
      write(outfile, '|S11|   <S11(D) |S21|   <S21(D) |S12|   <S12(D) |S22|   <S22(D) Fmin(dB) |Gopt| <Gopt(D)
Rn/50   ');
      writeln(outfile,'|S11|   <S11(D) |S21|   <S21(D) |S12|   <S12(D) |S22|   <S22(D) Fmin(dB) |Gopt| <Gopt(D)
Rn/50 ');
      write(outfile, '--- ------  ------ ------  ---   ');
      write(outfile, '-------------------------------------------------------------------------------   ');
      writeln(outfile,'-------------------------------------------------------------------------------  ' );

      for i:= 0 to num-1 do
      begin
        str( i:2, st1 );
        write(outfile,st1+'  ');
        for ii:= 0 to inputs-1 do
        begin
          str( matrix_tmp[trunc(j/Every)-1,i,ii]:6:2, st1 );
          write(outfile,st1+' ');
        end;
        write(outfile,' ');
        for ii:= 0 to outputs-1 do
        begin
                                                          { CT: Denormalization of calculated values  }
                          matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:=matrix_tmp[trunc(j/Every)-1,i,inputs+ii];
          str( matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:5:5, st1);
          case (ii) of
             1,3,5,7,10:
              begin  { degrees }
                if matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii] > 0.5 then
                begin
                  matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:=matrix_tmp_out[trunc(j/Every)-
1,i,inputs+ii]*Degree;
                  matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:=matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]-
Degree;
                  str( matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:5:5, st1);
                end
                else
                begin
                  matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:=matrix_tmp_out[trunc(j/Every)-
1,i,inputs+ii]*Degree;
                  str( matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:5:5, st1);
                end;
              end;
```

```
            2:begin
                matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:=matrix_tmp_out[trunc(j/Every)-
1,i,inputs+ii]*S21max;
                str( matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:5:5, st1);
              end;
            8:begin
                matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:=matrix_tmp_out[trunc(j/Every)-
1,i,inputs+ii]*Fminmax;
                str( matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:5:5, st1);
              end;
          end;
          write( outfile,st1+' ');
        end;
        write(outfile,' ');
        for ii:=0 to outputs-1 do
        begin
                                                    { CT: Denormalization of desired values }
          dessout[i,ii]:=dess[i,ii];
          str( dessout[i,ii]:5:5, st1 );
          case (ii) of
            1,3,5,7,10:
              begin  { degrees }
                if dessout[i,ii] > 0.5 then
                begin
                  dessout[i,ii]:=dessout[i,ii]*Degree-Degree;
                  str( dessout[i,ii]:5:5, st1);
                end
                else
                begin
                  dessout[i,ii]:=dessout[i,ii]*Degree;
                  str( dessout[i,ii]:5:5, st1);
                end;
              end;
            2:begin
                dessout[i,ii]:=dessout[i,ii]*S21max;
                str( dessout[i,ii]:5:5, st1);
              end;
            8:begin
                dessout[i,ii]:=dessout[i,ii]*Fminmax;
                str( dessout[i,ii]:5:5, st1);
              end;
          end;
          write( outfile, st1+' ');
        end;
        Writeln(outfile,' ');
      end;
      terror:= terror/outputs;
      str ( j:7, st1 ); str ( eps:9:6, st2 );          { eps=nu }
      str ( error:9:6, st3 ); str ( terror:9:6, st4 );
      writeln( outfile, ' ');
      writeln( outfile, 'iteration=',st1,'   nu =',st2,
      '   error_learning=', st3, '   error_test=', st4 );
      writeln( outfile, ' ');
```

```
if ( j = Iterative_Max ) then
begin
  writeln( outfile, 'learning and test finished!  (CT=0.1 --> CE, CT=0.9 --> CC)');
end;

if tercih_test='Y' then  { CT: test of data that are not learnin data }
begin

  terror:= 0;
  GetTime(gettime_h,gettime_m,gettime_s,gettime_hund);
   WriteLn('Test Started (not learning data)... ',LeadingZero(gettime_h),':',
   LeadingZero(gettime_m),':',LeadingZero(gettime_s),
   '.',LeadingZero(gettime_hund));
  for isylla:= 0 to num-1 do
  begin                    { read input_t.dat }
    yread_test(isylla);
    wforward;
    terr:= back;
    terror:= terror + terr;
    make_matrix( trunc( j/Every ) - 1, isylla );
  end;
  GetTime(gettime_h,gettime_m,gettime_s,gettime_hund);
   WriteLn('Test Finished (not learning data)! ',LeadingZero(gettime_h),':',
   LeadingZero(gettime_m),':',LeadingZero(gettime_s),
   '.',LeadingZero(gettime_hund));

  writeln(outfile_t,'        Input Values                                    Calculated Values');
  write(outfile_t,  'Num f(GHz)  VCE(V) IC(mA)  CT    ');
  write(outfile_t,'|S11|    <S11(D)  |S21|    <S21(D)  |S12|    <S12(D)  |S22|    <S22(D)  Fmin(dB)  |Gopt|
<Gopt(D) Rn/50 ');
  writeln(outfile_t,' ');
  write(outfile_t,  '--- ------  ------ ------  ---   ');
  writeln(outfile_t,'-------------------------------------------------------------------------------');

  for i:= 0 to num-1 do
  begin
    str( i:2, st1 );
    write(outfile_t,st1+'  ');
    for ii:= 0 to inputs-1 do
    begin
      str( matrix_tmp[trunc(j/Every)-1,i,ii]:6:2, st1 );
      write(outfile_t,st1+' ');
    end;
    write(outfile_t,' ');
    for ii:= 0 to outputs-1 do
    begin
                                                    { CT: Denormalization of calculated values  }
                    matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:=matrix_tmp[trunc(j/Every)-1,i,inputs+ii];
      str( matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:5:5, st1);
      case (ii) of
        1,3,5,7,10:
          begin  { degrees }
```

```
                if      matrix_tmp_out[trunc(j/Every)-      1,i,inputs+ii] > 0.5 then
                begin
                  matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:=matrix_tmp_out[trunc(j/Every)-
1,i,inputs+ii]*Degree;
                  matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:=matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]-
Degree;
                  str( matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:5:5, st1);
                end
                else
                begin
                  matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:=matrix_tmp_out[trunc(j/Every)-
1,i,inputs+ii]*Degree;
                  str( matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:5:5, st1);
                end;
              end;
             2:begin
                matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:=matrix_tmp_out[trunc(j/Every)-
1,i,inputs+ii]*S21max;
                str( matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:5:5, st1);
              end;
             8:begin
                matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:=matrix_tmp_out[trunc(j/Every)-
1,i,inputs+ii]*Fminmax;
                str( matrix_tmp_out[trunc(j/Every)-1,i,inputs+ii]:5:5, st1);
              end;
            end;
            write( outfile_t,st1+' ');
          end;

{ Ct: There is no desired values for test
          write(outfile_t,' ');
          for ii:=0 to outputs-1 do
          begin
            str( dess[i,ii]:5:5, st1 );
            write( outfile_t, st1+' ');
          end;
}

          Writeln(outfile_t,' ');
        end;
        terror:= terror/outputs;
        str ( j:7, st1 ); str ( eps:9:6, st2 );              { eps=nu }
        str ( error:9:6, st3 ); str ( terror:9:6, st4 );
        writeln( outfile_t, ' ');
        writeln( outfile_t, 'iteration=',st1,'    nu =',st2,
        '   error_learning=', st3, '    error_test=', st4 );
        writeln( outfile_t, ' ');
        if ( j = Iterative_Max ) then
        begin
          writeln( outfile_t, 'test finished (not learning data)!  (CT=0.1 --> CE, CT=0.9 --> CC)');
        end;

      end;
```

```
      write_weights;

    end;  {main loop}
    close( outfile );
    close( outfile_t );
    GetTime(gettime_h,gettime_m,gettime_s,gettime_hund);
    WriteLn('Program Finished! ',LeadingZero(gettime_h),':',
     LeadingZero(gettime_m),':',LeadingZero(gettime_s),
     '.',LeadingZero(gettime_hund));
    write('Ok!');
end.
```

Ek 2 Performans Analizi Programı (PerAna.m)

```
%/*********************************************************** */
%/*                                                          */
%/* File: PerAna.m (Performan Analizi Programı)              */
%/*                                                          */
%/* Compiler: Matlab v5.0                                    */
%/*                                                          */
%/* Description:Takes the output of Neural Network Program   */
%/*          nnpsn.pas (S and Noise parameters) and          */
%/*          calculates GTmax,ZL,ZS and stability            */
%/*          condition according to a given Vireq, Freq      */
%/*          pairs geometrically. Additionally calculates    */
%/*          all possible ZL,ZL terminations that            */
%/*          correspond to a given GTreq.                    */
%/*                                                          */
%/* Created: Cemal Tepe 1999 Ph.D. Thesis                    */
%/*                                                          */
%/* Version:1.0                                              */
%/*                                                          */
%/* Notes: to run this program for a different transistor, quit Matlab than  */
%/*          run program again. (to discard previous values) */
%/*********************************************************** */
%
%

% **** Arrange the required one: SP *****************%

% fid_r_s=fopen('in_s.ins','r');  % read S matrix from a file  N_Num=1 %
% fid_r_n=fopen('in_s.inn','r');  % read Noise matrix from a file %
% fid_r_p=fopen('in_s.inp','r');  % read input file, GTdBreq,Vireq,FdBreq %

% fid_r_s=fopen('n72089aa.ins','r');  % read S matrix from a file  N_Num=6, n72089aa(Vds=3V, Ids=10mA) %
% fid_r_n=fopen('n72089aa.inn','r');  % read Noise matrix from a file %
% fid_r_p=fopen('n72089aa.inp','r');  % read input file, GTdBreq,Vireq,FdBreq %

% fid_r_s=fopen('n72089aa.ins','r');     % read Neural Network SN matrix from a file  N_Num=11,
n72089aa(Vds=3V, Ids=10mA) from Neural Network %
% fid_r_n=fopen('n72089aa.inn','r');  % read Noise matrix from a file %
% fid_r_p=fopen('n72089aa.inp','r');  % read input file, GTdBreq,Vireq,FdBreq %

% fid_r_s=fopen('ne329s01.ins','r');  % read S matrix from a file  N_Num=9, ne329s01(Vds=2V, Id=10mA) %
% fid_r_n=fopen('ne329s01.inn','r');  % read Noise matrix from a file %
% fid_r_p=fopen('ne329s01.inp','r');  % read input file, GTdBreq,Vireq,FdBreq %

% fid_r_s=fopen('ne329s01.ins','r');     % read Neural Network SN matrix from a file  N_Num=17,
ne329s01(Vds=2V, Id=10mA) from Neural Network %
% fid_r_n=fopen('ne329s01.inn','r');  % read Noise matrix from a file %
% fid_r_p=fopen('ne329s01.inp','r');  % read input file, GTdBreq,Vireq,FdBreq %

% fid_r_s=fopen('ne02135c.ins','r');  % read S matrix from a file  N_Num=7, ne02135c(VCE=10V, IC=20mA)
%
% fid_r_n=fopen('ne02135c.inn','r');  % read Noise matrix from a file %
```

```
% fid_r_p=fopen('ne02135c.inp','r');    % read input     file, GTdBreq,Vireq,FdBreq %

% fid_r_s=fopen('ne02135c.ins','r');      % read Neural Network SN matrix from a file  N_Num=14,
ne02135c(VCE=10V, IC=20mA) from Neural Network %
% fid_r_n=fopen('ne02135c.inn','r');  % read Noise matrix from a file %
% fid_r_p=fopen('ne02135c.inp','r');  % read input file, GTdBreq,Vireq,FdBreq %

 fid_r_s=fopen('ne02135.ins','r');  % read S matrix from a file  N_Num=7, ne02135(VCE=10V, IC=5mA) %
 fid_r_n=fopen('ne02135.inn','r');  % read Noise matrix from a file %
 fid_r_p=fopen('ne02135.inp','r');  % read input file, GTdBreq,Vireq,FdBreq %

% fid_r_s=fopen('ne3210s1.ins','r');  % read S matrix from a file  N_Num=9, ne3210s1(Vds=2V, Id=10mA) %
% fid_r_n=fopen('ne3210s1.inn','r');  % read Noise matrix from a file %
% fid_r_p=fopen('ne3210s1.inp','r');  % read input file, GTdBreq,Vireq,FdBreq %

% fid_r_s=fopen('ne38018.ins','r');  % read S matrix from a file  N_Num=6, ne38018(Vds=2V, Id=10mA) %
% fid_r_n=fopen('ne38018.inn','r');  % read Noise matrix from a file %
% fid_r_p=fopen('ne38018.inp','r');  % read input file, GTdBreq,Vireq,FdBreq %

% fid_r_s=fopen('ne38018x.ins','r');  % read S matrix from a file  N_Num=6, ne38018(Vds=3V, Id=5mA) %
% fid_r_n=fopen('ne38018x.inn','r');  % read Noise matrix from a file %
% fid_r_p=fopen('ne38018x.inp','r');  % read input file, GTdBreq,Vireq,FdBreq %

% fid_r_s=fopen('ne38018y.ins','r');  % read S matrix from a file  N_Num=6, ne38018(Vds=2V, Id=5mA) %
% fid_r_n=fopen('ne38018y.inn','r');  % read Noise matrix from a file %
% fid_r_p=fopen('ne38018y.inp','r');  % read input file, GTdBreq,Vireq,FdBreq %


Name_of_transistor='ne02135(VCE=10V, IC=5mA) ';
N_Num=7;     % Number of frequencies (that have Noise parameters) %
P_Num=1;     % Number of rows in input file (*.inp) %

step_for_sample_on_circle=0.01;     % step of test circle's radius
switch_for_plot=0;                  % 1 --> Plot circles, 0 --> Do not plot circles %


% ****  Arrange the required one: EP ********************%

fid_w_g = fopen('out_gra.txt','wt+'); % write output to a file for a graph%
fid_w = fopen('out_int.txt','wt+'); % write intersection points to a file %
fid_w_o = fopen('out_oth.txt','wt+'); % write other points  to a file %
fprintf(fid_w,'Transistor: %s\n',Name_of_transistor);
fprintf(fid_w,'\nDetailed information,Terminations for GTmax and all of intersection points on T1/T2 for
GTreq:\n');
fprintf(fid_w_o,'Transistor: %s\n',Name_of_transistor);
fprintf(fid_w_o,'\nTerminations that give GTreq in Region-3:\n');
fprintf(fid_w_g,'Transistor: %s\n',Name_of_transistor);
fprintf(fid_w_g,'\nTerminations for GTmax and One of Intersection points on T1/T2 for GTreq:\n');
% fprintf(fid_w_g,'\nf(Hz)          GTmax (dB)      GTmin (dB)      Real(ZL)          Imag(ZL)
        Real(ZS)          Imag(ZS)\n'); %
fprintf(fid_w_g,'\nVireq              Freq(dB)          f(Hz)              GTmax (dB)      Real(ZL)
        Imag(ZL)          Real(ZS)          Imag(ZS)          GTreq (dB)      Real(ZLreq)      Imag(ZLreq)
        Real(ZSreq)       Imag(ZSreq)\n');
```

```
fclose(fid_w);
fclose(fid_w_o);
fclose(fid_w_g);



Smatrix=fscanf(fid_r_s,'%f',[9,inf]);    % read whole S-par file %
Nmatrix=fscanf(fid_r_n,'%f',[5,inf]);    % read whole N-par file %
Pmatrix=fscanf(fid_r_p,'%f',[3,inf]);    % read whole Input File file %

for input_counts=1:P_Num   % main loop %

 GTDBreq=Pmatrix(1,input_counts);
 Vireq =Pmatrix(2,input_counts);
 FdB =Pmatrix(3,input_counts);

 fid_w_g = fopen('out_gra.txt','at+'); % write output to a file for a graph%
 fid_w = fopen('out_int.txt','at+'); % write intersection points to a file %
 fid_w_o = fopen('out_oth.txt','at+'); % write other points  to a file %

 fprintf(fid_w,'\n*** %d. Given Values:\n',input_counts);
 fprintf(fid_w_o,'\n*** %d. Given Values:\n',input_counts);
 %fprintf(fid_w_g,'\n*** %d. Given Values:\n',input_counts);

 % GTDBreq = input('Required GT(dB) value?: ') ;
 fprintf(fid_w,'Required GT(dB) value: %f\n',GTDBreq);
 fprintf(fid_w_o,'Required GT(dB) value: %f\n',GTDBreq);
 %fprintf(fid_w_g,'Required GT(dB) value: %f\n',GTDBreq);

 % Vireq = input('Required Vi value?: ') ;
 fprintf(fid_w,'Required Vi value: %f\n',Vireq);
 fprintf(fid_w_o,'Required Vi value: %f\n',Vireq);
 %fprintf(fid_w_g,'Required Vi value: %f\n',Vireq);

 % FdB = input ('Required F(dB) value?: ');
 fprintf(fid_w,'Required F(dB) value: %f\n',FdB);
 fprintf(fid_w_o,'Required F(dB) value: %f\n',FdB);
 fclose(fid_w_o);
 %fprintf(fid_w_g,'Required F(dB) value: %f\n',FdB);


 % Input Reflection Coefficient and Required Noice Figure (not in dB) %
 Rmod2 = (Vireq - 1)/(Vireq + 1);
 Rmod=sqrt(Rmod2);
 C = 1-(Rmod^2);

 Freq = 10^(FdB*0.1);

 teta = 0:2*pi/100:2*pi;

 for counts=1:N_Num   % loop under main loop %
```

```
f=Smatrix(1,counts);
s11 =Smatrix(2,counts);
o11 =Smatrix(3,counts);
s21 =Smatrix(4,counts);
o21 =Smatrix(5,counts);
s12 =Smatrix(6,counts);
o12 =Smatrix(7,counts);
s22 =Smatrix(8,counts);
o22 =Smatrix(9,counts);

fprintf('f:  %f\n',f);
fprintf('s11 and o11:  %f %f \n',s11,o11);
fprintf('s21 and o21:  %f %f \n',s21,o21);
fprintf('s12 and o12:  %f %f \n',s12,o12);
fprintf('s22 and o22:  %f %f \n',s22,o22);


O11 = o11*(pi/180);
S11 = s11*exp(i*O11);
O21 = o21*(pi/180);
S21 = s21*exp(i*O21);
O12 = o12*(pi/180);
S12 = s12*exp(i*O12);
O22 = o22*(pi/180);
S22 = s22*exp(i*O22);


% Transistor Noise Parameters %

FirstCol=Nmatrix(1,counts);  % waste reading for first column %
FmindB=Nmatrix(2,counts);
Fmin=10^(FmindB*0.1);
GamaOpt_m=Nmatrix(3,counts);
GamaOpt_a_deg=Nmatrix(4,counts);
GamaOpt_a=GamaOpt_a_deg*(pi/180);
Gamaopt=GamaOpt_m*exp(i*GamaOpt_a);
RnBolu50=Nmatrix(5,counts);
Rn=RnBolu50;              % Zo is 50ohm %
%  Rn=RnBolu50*50;
omega = 2*pi*f ;
Z01 = 50;
Z02 = 50 ;
R01 = real(Z01);
R02 = real(Z02);

fprintf('\nCircuit Definition:\n');
fprintf(fid_w,'\nCircuit Definition:\n');

fprintf('f(Hz):  %f\n',f);
fprintf('S11 Module and Angle(Rad): %f %f \n',abs(S11),angle(S11));
```

```
fprintf('S21 Module and Angle(Rad): %f %f \n',abs(S21),angle(S21));
fprintf('S12 Module and Angle(Rad): %f %f \n',abs(S12),angle(S12));
fprintf('S22 Module and Angle(Rad): %f %f \n',abs(S22),angle(S22));
fprintf('Fmin %f, GamaOpt Module and Angle(Rad) %f %f, Rn %f \n',Fmin, GamaOpt_m,GamaOpt_a,Rn);

fprintf(fid_w,'f(Hz): %f\n',f);
fprintf(fid_w,'S11 Module and Angle(Rad): %f %f \n',abs(S11),angle(S11));
fprintf(fid_w,'S21 Module and Angle(Rad): %f %f \n',abs(S21),angle(S21));
fprintf(fid_w,'S12 Module and Angle(Rad): %f %f \n',abs(S12),angle(S12));
fprintf(fid_w,'S22 Module and Angle(Rad): %f %f \n',abs(S22),angle(S22));
fprintf(fid_w,'Fmin %f, GamaOpt Module and Angle(Rad) %f %f, Rn %f \n',Fmin,
GamaOpt_m,GamaOpt_a,Rn);


fid_w_o = fopen('out_oth.txt','at+'); % append to a file %
fprintf(fid_w_o,'\nf(Hz): %f\n',f);
fclose(fid_w_o);

deltas = ((1-S11)*(1-S22)-S12*S21)


% Converting S to Z %

Z11 =((1+S11)*(1-S22)+S12*S21)/deltas;
Z21 =(2*S21)/deltas;
Z12 =(2*S12)/deltas;
Z22 =((1+S22)*(1-S11)+S12*S21)/deltas;

R11 = real(Z11); X11 = imag(Z11);
R21 = real(Z21); X21 = imag(Z21);
R12 = real(Z12); X12 = imag(Z12);
R22 = real(Z22); X22 = imag(Z22);

deltaz =Z11*Z22 -Z12*Z21;

fprintf(fid_w,'Z11 Real and Imaginer parts: %f %f \n',real(Z11),imag(Z11));
fprintf(fid_w,'Z21 Real and Imaginer parts: %f %f \n',real(Z21),imag(Z21));
fprintf(fid_w,'Z12 Real and Imaginer parts: %f %f \n',real(Z12),imag(Z12));
fprintf(fid_w,'Z22 Real and Imaginer parts: %f %f \n',real(Z22),imag(Z22));


% Zopt Calculation %
Zopt = ((1+Gamaopt))/(1-Gamaopt);
Zoptm = abs(Zopt);
Zopta = angle(Zopt);
Ropt = real(Zopt);
Xopt = imag(Zopt);

% Theoretical Noice Parameters of Transistor %
Yopt = 1/Zopt;
Ycor = ((Fmin - 1)/(2*Rn)) - Yopt ;
Gu = ((Fmin - 1)*real(Yopt))-(((Fmin - 1)^2)/(4*Rn)) ;
```

```
% Center phasor and radious of the noise circles are calculated %
N = ((Freq - Fmin)*(Zoptm^2)) / (2*Rn);
Rcn = Ropt + N ;
Xcn = Xopt;
Zcn=Rcn+i*Xcn;
rn = sqrt(N*(N + 2*Ropt));
U = (1 + Rmod^2) / C;
V = 2*Rmod / C;

% T1 and T2 Circle Definitions %
Rct1 = Rcn*U + rn*V;
Xct1 = -Xopt;
Zct1 = Rct1 + i*Xct1;
Zct1m = abs(Zct1);
Zct1a = angle(Zct1);
rt1 = sqrt((Zct1m^2) - (Zoptm^2));

Rct2 = Rcn*U - rn*V;
Xct2 = -Xopt;
Zct2 = Rct2 + i*Xct2;
Zct2m = abs(Zct2);
Zct2a = angle(Zct2);
rt2 = sqrt( (Zct2m^2)-(Zoptm^2));

fprintf('Freq,Fmin,Ropt,Xopt,Rn,Rmod,N,Rcn,Xcn,rn,Rct1,Xct1,rt1,Rct2,Xct2,rt2: %f %f %f %f %f %f %f
%f %f %f %f %f %f %f %f %f %f
%f\n',Freq,Fmin,Ropt,Xopt,Rn,Rmod,N,Rcn,Xcn,rn,Rct1,Xct1,rt1,Rct2,Xct2,rt2);
fprintf(fid_w,'Freq,Fmin,Ropt,Xopt,Rn,Rmod,N,Rcn,Xcn,rn,Rct1,Xct1,rt1,Rct2,Xct2,rt2: %f %f %f %f %f %f
%f %f %f %f %f %f %f %f %f %f
%f\n\n',Freq,Fmin,Ropt,Xopt,Rn,Rmod,N,Rcn,Xcn,rn,Rct1,Xct1,rt1,Rct2,Xct2,rt2);

% Noise Figure checking
if Fmin > Freq
  fprintf('Error: Fmin > Freq, skip to the next input value\n');
  fprintf(fid_w,'\nWarning: Fmin > Freq, skip to the next input value\n');

  % exit loop (matlab stops here)
  break;
end

% T1 and T2 Circle Drawing %
Zi1 = Zct1 + rt1*exp(i*teta);
Zi2 = Zct2 + rt2*exp(i*teta);
Ri1 = real(Zi1);
Xi1 = imag(Zi1);
Ri2 = real(Zi2);
Xi2 = imag(Zi2);

%  Pre-Defined Values %
Z = Z12*Z21;
R = real(Z);
X = imag(Z);
```

```
Q = 2*R11*R22 - R
COEFS = (abs(Z12)^2)/(2*R22*C);
ita = Q/abs(Z);
GT_Negative=0 % assume that GT always pozitive (1 --> Negative) %


if R11>0 & R22>0 & ita>1   % Stability: if %

  %   Unconditional stability %

  fprintf('Unconditional stability: R11>0 & R22>0 & ita>1\n');
  fprintf(fid_w,'\nUnconditional stability: R11>0 & R22>0 & ita>1\n');
  fprintf('R11,R22,ita = %f,%f,%f \n ',R11,R22,ita);
  fprintf(fid_w,'R11,R22,ita = %f,%f,%f\n',R11,R22,ita);


  % Calculating values for Required GT value in Unconditional stability case

  P=(abs(Z12)^2/C)*(10^(GTDBreq*0.1));          % GTDBreq converted to dimentionless %
  RCGreq = (Q - P)/(2*R22);
  XCGreq = (2*X11*R22-X)/(2*R22);
  ZCGreq = RCGreq + i*XCGreq;
  rgreq = (sqrt(P^2-2*Q*P+abs(Z)^2))/(2*R22);

  % GTDBreq Circle %
  ZCGTreq= ZCGreq+rgreq*exp(i*teta);
  RCGTreq=real(ZCGTreq);
  XCGTreq=imag(ZCGTreq);

  % Find out the intersection of two circles
  InstersectionExist=1;
  fprintf('GTDBreq, T1 and T2 circles:\n');
  fprintf('RCGreq= %f, XCGreq= %f, rgreq= %f\n',RCGreq,XCGreq,rgreq);
  fprintf('Rct1= %f, Xct1= %f, rt1= %f\n',Rct1,Xct1,rt1);
  fprintf('Rct2= %f, Xct2= %f, rt2= %f\n',Rct2,Xct2,rt2);
  fprintf(fid_w,'GTDBreq, T1 and T2 circles:\n');
  fprintf(fid_w,'RCGreq= %f, XCGreq= %f, rgreq= %f\n',RCGreq,XCGreq,rgreq);
  fprintf(fid_w,'Rct1= %f, Xct1= %f, rt1= %f\n',Rct1,Xct1,rt1);
  fprintf(fid_w,'Rct2= %f, Xct2= %f, rt2= %f\n',Rct2,Xct2,rt2);


  [RintT1,XintT1]=circlint(RCGreq,XCGreq,rgreq,Rct1,Xct1,rt1);
  ZireqT11=RintT1(1)+i*XintT1(1);
  ZireqT12=RintT1(2)+i*XintT1(2);

  fprintf('1. Intersection point on T1; RintT1= %f: XintT1= %f\n',RintT1(1),XintT1(1));
  fprintf(fid_w,'1. Intersection point on T1; RintT1= %f: XintT1= %f\n',RintT1(1),XintT1(1));
  fprintf('2. Intersection point on T1; RintT1= %f: XintT1= %f\n',RintT1(2),XintT1(2));
  fprintf(fid_w,'2. Intersection point on T1; RintT1= %f: XintT1= %f\n',RintT1(2),XintT1(2));

  fprintf('ZireqT11 Value: real and imaginer part =   %f %f\n',RintT1(1)*50,XintT1(1)*50);
  fprintf(fid_w,'ZireqT11 Value: real and imaginer part =   %f %f\n',RintT1(1)*50,XintT1(1)*50);
```

```
    fprintf('ZireqT12 Value: real and imaginer part =      %f %f\n',RintT1(2)*50,XintT1(2)*50);
    fprintf(fid_w,'ZireqT12 Value: real and imaginer part =   %f %f\n',RintT1(2)*50,XintT1(2)*50);

    [RintT2,XintT2]=circlint(RCGreq,XCGreq,rgreq,Rct2,Xct2,rt2);
    ZireqT21=RintT2(1)+i*XintT2(1);  % two intersection points on T2 %
    ZireqT22=RintT2(2)+i*XintT2(2);

    fprintf('1. Intersection point on T2; RintT2= %f: XintT2= %f\n',RintT2(1),XintT2(1));
    fprintf(fid_w,'1. Intersection point on T2; RintT2= %f: XintT2= %f\n',RintT2(1),XintT2(1));
    fprintf('2. Intersection point on T2; RintT2= %f: XintT2= %f\n',RintT2(2),XintT2(2));
    fprintf(fid_w,'2. Intersection point on T2; RintT2= %f: XintT2= %f\n',RintT2(2),XintT2(2));

    fprintf('ZireqT21 Value: real and imaginer part =   %f %f\n',RintT2(1)*50,XintT2(1)*50);
    fprintf(fid_w,'ZireqT21 Value: real and imaginer part =   %f %f\n',RintT2(1)*50,XintT2(1)*50);
    fprintf('ZireqT22 Value: real and imaginer part =   %f %f\n',RintT2(2)*50,XintT2(2)*50);
    fprintf(fid_w,'ZireqT22 Value: real and imaginer part =   %f %f\n',RintT2(2)*50,XintT2(2)*50);

    if ~isnan(ZireqT11) & ~isnan(ZireqT12) & ~isnan(ZireqT21) & ~isnan(ZireqT22)   % GTreq cuts both T1
and T2
      fprintf('GTreq cuts both T1 and T2\n');
      fprintf(fid_w,'GTreq cuts both T1 and T2\n');
    elseif ~isnan(ZireqT11) & ~isnan(ZireqT12)                    % GTreq cuts only T1
      fprintf('GTreq cuts only T1\n');
      fprintf(fid_w,'GTreq cuts only T1\n');
    elseif ~isnan(ZireqT21) & ~isnan(ZireqT22)                    % GTreq cuts only T2
      fprintf('GTreq cuts only T2\n');
      fprintf(fid_w,'GTreq cuts only T2\n');
    else                                                % GTreq does not cut T1 or T2
      fprintf('GTreq does not cut T1 or T2\n');
      fprintf(fid_w,'GTreq does not cut T1 or T2\n');
      InstersectionExist=0;
      ZSreq= NaN+i*NaN;
      ZLreq= NaN+i*NaN;
%       break;
    end

    RCS = -(2*R11*R22-R)/(2*R22);
    XCS = -(2*X11*R22-X)/(2*R22);
    ZCS = RCS + i*XCS;
    rs = abs(Z)/(2*R22);
    ZCSC = conj(ZCS)

    % Conjugate Source Stability Circle (CSSC) %
    Zi3 = ZCSC +  rs * exp(i*teta);
    Ri3 = real(Zi3);
    Xi3 = imag(Zi3);

    % Input Stability Circle (ISC) (ZCINPUT = -RCS - i*XCS; rINPUT=rs) %
    ZCINPUT = -RCS - i*XCS;
    Zi5 = ZCINPUT + rs * exp(i*teta);
    Ri5 = real(Zi5);
    Xi5 = imag(Zi5);
```

```
% GT Circles %
XCG = (2*X11*R22-X)/(2*R22);
GTMAX=(Q-sqrt(Q^2-abs(Z)^2)*C)/(abs(Z12)^2);

for k=1:6
      y(k)=(k-1)*(GTMAX/5);
      P(k)=(abs(Z12)^2*y(k))/C;
      RCG(k) = (Q - P(k))/(2*R22);
 ZCG(k) = RCG(k) + i*XCG;
 rg(k) = (sqrt(P(k)^2-2*Q*P(k)+abs(Z)^2))/(2*R22);
end

ZCG1 = ZCG(1);rg1 = rg(1);
ZCG2 = ZCG(2);rg2 = rg(2);
ZCG3 = ZCG(3);rg3 = rg(3);
ZCG4 = ZCG(4);rg4 = rg(4);
ZCG5 = ZCG(5);rg5 = rg(5);
ZCG6 = ZCG(6);rg6 = rg(6);

Zi41 = ZCG1 + rg1 * exp(i*teta);
  Ri41 = real(Zi41);Xi41 = imag(Zi41);
Zi42 = ZCG2 + rg2 * exp(i*teta);
  Ri42 = real(Zi42);Xi42 = imag(Zi42);
Zi43 = ZCG3 + rg3 * exp(i*teta);
  Ri43 = real(Zi43);Xi43 = imag(Zi43);
Zi44 = ZCG4 + rg4 * exp(i*teta);
  Ri44 = real(Zi44);Xi44 = imag(Zi44);
Zi45 = ZCG5 + rg5 * exp(i*teta);
  Ri45 = real(Zi45);Xi45 = imag(Zi45);
Zi46 = ZCG6 + rg6 * exp(i*teta);
  Ri46 = real(Zi46);Xi46 = imag(Zi46);


%  Conserning Solution %
RCGmax = sqrt(Q^2-abs(Z)^2)/(2*R22);
XCGmax = XCG;
ZCGmax = RCGmax + i*XCGmax;

fprintf('ZCGmax(%f %f),Zct1(%f %f), Zct2(%f %f)\n',RCGmax,XCGmax,Rct1, Xct1, Rct2, Xct2);
fprintf('rt1 %f, rt2 %f\n',rt1, rt2);
fprintf(fid_w,'ZCGmax(%f %f),Zct1(%f %f), Zct2(%f %f)\n',RCGmax,XCGmax,Rct1, Xct1, Rct2, Xct2);
fprintf(fid_w,'rt1 %f, rt2 %f\n',rt1, rt2);

if abs(ZCGmax-Zct1) > rt1  % Conserning Solution %

 fprintf('abs(ZCGmax-Zct1) > rt1: ZCGmax in Region-1 (tangent case --> GTmax)\n');
     fprintf(fid_w,'abs(ZCGmax-Zct1) > rt1: ZCGmax in Region-1 (tangent case --> GTmax)\n');

 %  Zcgmax in Region-1 %
 %  GTMAX value; absolute stability and Zcgmax in Region-1 (tangent case --> GTmax) %
```

```
    if (Vireq == 1) & (Freq == Fmin)  % VSWR and Noise match, rn=rv=rt1=tr2=0,
Zs=Zopt=Zcn,Rmod=0 Zi=conj(Zopt), Zcv=conj(Zi)
      GTMC=(Q-sqrt(Q^2-abs(Z)^2)*C)/(abs(Z12)^2);
    else
        D=(R22*abs(Z11)^2-R*R11-X*X11)/R22+abs(Zopt)^2;
        E=(D*R22-Rct1*(2*R11*R22-R)-2*Xct1*XCG*R22)/rt1;
        F=Rct1/rt1;

        GTpayn=C*(Q+E*F-sqrt(((Q+E*F)^2-(1-F^2)*(abs(Z)^2-E^2))));
        GTpayda=(abs(Z12)^2*(1-F^2));
        GTMC=GTpayn/GTpayda;
    end

      P=(abs(Z12)^2/C)*GTMC;

      RCGmax = (Q - P)/(2*R22);
    XCGmax = XCG;
      ZCGmax = RCGmax + i*XCGmax;
      rgmax = (sqrt(P^2-2*Q*P+abs(Z)^2))/(2*R22);

    % GTmax Circle %
      ZCGTM= ZCGmax+rgmax*exp(i*teta);
      RCGTM=real(ZCGTM);
      XCGTM=imag(ZCGTM);


    %  Load and Source impedance calculation; Zcgmax in Region-1 %
      % Zi value %
    if (Vireq == 1) & (Freq == Fmin)  % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
      Zi=conj(Zopt);
    else
        Zi=(((rgmax/(rt1+rgmax))*Zct1)+((rt1/(rt1+rgmax))*ZCGmax));
      end

      fprintf('Zi Value: real and imaginer part =  %f %f\n',real(Zi)*50,imag(Zi)*50);
      fprintf(fid_w,'Zi Value: real and imaginer part =  %f %f\n',real(Zi)*50,imag(Zi)*50);

      %        fprintf(fid_w,'Q  %f,P  %f,abs(Z)  %f,rgmax  %f,rt1
%f,Zct1(%f,%f),ZCGmax(%f,%f)\n',Q,P,abs(Z),rgmax,rt1,real(Zct1),imag(Zct1),real(ZCGmax),imag(ZCGmax)
);

      % ZL value %
      ZL=((Z12*Z21)/(Z11-Zi)-Z22);

      fprintf('ZL Value: real and imaginer part =  %f %f\n',real(ZL)*50,imag(ZL)*50);
      fprintf(fid_w,'ZL Value: real and imaginer part =  %f %f\n',real(ZL)*50,imag(ZL)*50);

      % ZS values %
      Rcv=(1+Rmod^2)*real(Zi)/C;
      Xcv=-imag(Zi);
      Zcv=Rcv+i*Xcv;
      rv=2*Rmod*real(Zi)/C;
```

```
    if (Vireq == 1) & (Freq == Fmin)   % VSWR   and   Noise   match,   rn=rv=rt1=tr2=0,
Zs=Zopt=Zcn,Rmod=0 Zi=conj(Zopt), Zcv=conj(Zi)
      ZS=Zopt;
    else
        ZS=(((rn/(rn+rv))*Zcv)+((rv/(rv+rn))*Zcn));
      end

      fprintf('ZS Value: real and imaginer part =  %f %f\n',real(ZS)*50,imag(ZS)*50);
      fprintf(fid_w,'ZS Value: real and imaginer part =  %f %f\n',real(ZS)*50,imag(ZS)*50);


    % MAG value; absolute stability and Zcgmax in Region-1 (Rmod=0--> C=1) %
      MAGndB=(Q-sqrt(Q^2-abs(Z)^2))/(abs(Z12)^2);
      MAG=10*log10(MAGndB);  % MAG in dB %

      fprintf('MAG Value (dB) = %f\n',MAG);
      %fprintf(fid_w,'MAG Value (dB) = %f\n',MAG);

    % GTMAX value; absolute stability and Zcgmax in Region-1 %
    GTMAX=10*log10(GTMC);
    GTMIN=0;  % in dB %

%     if GTMAX > MAG     % MAG is the available maximum gain %
%       GTMAX=MAG;
%     end;

    fprintf('GTMAX Value (dB) = %f\n',GTMAX);
      fprintf(fid_w,'GTMAX Value (dB) = %f\n',GTMAX);
      fprintf('GTMIN Value (dB) = %f\n',GTMIN);
      fprintf(fid_w,'GTMIN Value (dB) = %f\n',GTMIN);

    % Define values to plot f-GTmax %
    x_f(counts)=f;
    y_gtmax(counts)=GTMAX;
    y_gtreq(counts)=GTDBreq;
    z_gtmin(counts)=GTMIN;


      % fid_w will be open in empcal.m
      fclose(fid_w);

      % Caluclate empedances on T1 and T2 intersection points and other points in Region-3
      if (InstersectionExist ~= 0) & (Vireq ~= 1) & (Freq ~= Fmin)  % go to claculation in Region-3


[ZLreq,ZSreq]=empcal(ZireqT11,ZireqT12,ZireqT21,ZireqT22,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn
,RCGreq,XCGreq,rgreq,Rct2,Xct2,rt2,Rct1,Xct1,rt1,Ri1,Xi1,Ri2,Xi2,step_for_sample_on_circle,switch_for_plo
t)

    elseif (InstersectionExist ~= 0) & (Vireq == 1) & (Freq == Fmin)  % T1,T2,Vi.F  are a point

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %
```

```
        % Zi value %
            fprintf('ZireqT11=ZireqT12=ZireqT21=ZireqT22= Value: real and imaginer part =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);
          fprintf(fid_w,'ZireqT11=ZireqT12=ZireqT21=ZireqT22= Value: real and imaginer part =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);

    % ZLreq value %
    ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);

        % ZLreq value %
        fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);

    % ZSreq value %
    ZSreq=Zopt;

        % ZSreq value %
        fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
        fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

        % fid_w will be open in empcal.m
        fclose(fid_w);

        ZLreq=ZLreq1;  % print the first one in the intersection %


  elseif (InstersectionExist ~= 0) & (Freq == Fmin)  % T1 and T1 identical, Zs=Zopt

    % open fid_w for append
    fid_w = fopen('out_int.txt','at+'); % write output to a file %

        % Zi value %
        fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real
and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
         fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22
Value: real and imaginer part = %f
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

    % ZLreq value %
    ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
    ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

        % ZLreq value %
        fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
        fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

    % ZSreq value %
    ZSreq=Zopt;

        % ZSreq value %
```

```
        fprintf('ZSreq Value: real and imaginer part      =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
        fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

        % fid_w will be open in empcal.m
        fclose(fid_w);

        ZLreq=ZLreq1;  % print the first one in the intersection %

    elseif (InstersectionExist ~= 0) & (Vireq == 1)   % T1 and T1 identical, Zs=conj(Zi)

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

        % Zi value %
        fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
        fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

      % ZLreq value %
      ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
      ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

        % ZLreq value %
        fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
        fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

      % ZSreq value %
      ZSreq1=conj(ZireqT11);
      ZSreq2=conj(ZireqT12);

        % ZSreq value %
        fprintf('ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
        fprintf(fid_w,'ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
        fprintf('ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);
        fprintf(fid_w,'ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);

        % fid_w will be open in empcal.m
        fclose(fid_w);

        ZLreq=ZLreq1;  % print the first one in the intersection %
        ZSreq=ZSreq1;  % print the first one in the intersection %

      else

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

      fprintf('No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');
      fprintf(fid_w,'No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');
```

```
            % fid_w will be open in empcal.m
            fclose(fid_w);

        ZSreq= NaN+i*NaN;
        ZLreq= NaN+i*NaN;

      end;

%       fprintf(fid_w_g,'%f      %f      %f      %f      %f      %f
        %f\n',f,GTMAX,GTMIN,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50); %
        fprintf(fid_w_g,'%f      %f      %f      %f      %f      %f      %f      %f      %f      %f
        %f      %f
        %f\n',Vireq,FdB,f,GTMAX,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50,GTDBreq,real(ZLreq)
*50,imag(ZLreq)*50,real(ZSreq)*50,imag(ZSreq)*50);

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

    elseif abs(ZCGmax-Zct1) == rt1

        fprintf('abs(ZCGmax-Zct1) == rt1: ZCGmax in Region-2\n');
        fprintf(fid_w,'abs(ZCGmax-Zct1) == rt1: ZCGmax in Region-2\n');

      % Zcgmax in Region-2 %
      % GTMAX value; Zcgmax in Region-2 %
        GTMC=(Q-sqrt(Q^2-abs(Z)^2)*C)/(abs(Z12)^2);

      % GTmax Circle (a point )%
      rgmax=0;
        ZCGTM= ZCGmax+rgmax*exp(i*teta);
        RCGTM=real(ZCGTM);
        XCGTM=imag(ZCGTM);

      % Load and Source impedance calculation; Zcgmax in Region-2 %
        % Zi value %
      if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
        Zi=conj(Zopt);
      else
          Zi=ZCGmax;
        end

        fprintf('Zi Value: real and imaginer part =   %f %f\n',real(Zi)*50,imag(Zi)*50);
        fprintf(fid_w,'Zi Value: real and imaginer part =   %f %f\n',real(Zi)*50,imag(Zi)*50);

        % ZL value %
        ZL=((Z12*Z21)/(Z11-Zi)-Z22);

        fprintf('ZL Value: real and imaginer part =   %f %f\n',real(ZL)*50,imag(ZL)*50);
        fprintf(fid_w,'ZL Value: real and imaginer part =   %f %f\n',real(ZL)*50,imag(ZL)*50);

        % ZS values %
```

```
        Rcv=(1+Rmod^2)*real(Zi)/C;
        Xcv=-imag(Zi);
        Zcv=Rcv+i*Xcv;
        rv=2*Rmod*real(Zi)/C;
    if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
      ZS=Zopt;
    else
        ZS=(((rn/(rn+rv))*Zcv)+((rv/(rv+rn))*Zcn));
        end

        fprintf('ZS Value: real and imaginer part =   %f %f\n',real(ZS)*50,imag(ZS)*50);
        fprintf(fid_w,'ZS Value: real and imaginer part =   %f %f\n',real(ZS)*50,imag(ZS)*50);

    %  MAG value; absolute stability and Zcgmax in Region-2 (Rmod=0--> C=1) %
        MAGndB=(Q-sqrt(Q^2-abs(Z)^2))/(abs(Z12)^2);
        MAG=10*log10(MAGndB);  % MAG in dB %

        fprintf('MAG Value (dB) = %f\n',MAG);
        %fprintf(fid_w,'MAG Value (dB) = %f\n',MAG);

    %  GTMAX value; absolute stability and Zcgmax in Region-2 %
    GTMAX=10*log10(GTMC);
    GTMIN=0; % in dB %

%     if GTMAX > MAG     % MAG is the available maximum gain %
%       GTMAX=MAG;
%     end;

    fprintf('GTMAX Value (dB) = %f\n',GTMAX);
        fprintf(fid_w,'GTMAX Value (dB) = %f\n',GTMAX);
        fprintf('GTMIN Value (dB) = %f\n',GTMIN);
        fprintf(fid_w,'GTMIN Value (dB) = %f\n',GTMIN);

    %  Define values to plot f-GTmax %
    x_f(counts)=f;
    y_gtmax(counts)=GTMAX;
    y_gtreq(counts)=GTDBreq;
    z_gtmin(counts)=GTMIN;


        % fid_w will be open in empcal.m
        fclose(fid_w);

        % Caluclate empedances on T1 and T2 intersection points and other points in Region-3
        if (InstersectionExist ~= 0) & (Vireq ~= 1) & (Freq ~= Fmin)   % go to claculation in Region-3


[ZLreq,ZSreq]=empcal(ZireqT11,ZireqT12,ZireqT21,ZireqT22,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn
,RCGreq,XCGreq,rgreq,Rct2,Xct2,rt2,Rct1,Xct1,rt1,Ri1,Xi1,Ri2,Xi2,step_for_sample_on_circle,switch_for_plo
t)

    elseif (InstersectionExist ~= 0) & (Vireq == 1) & (Freq == Fmin)  % T1,T2,Vi.F  are a point
```

```
% open fid_w for append
fid_w = fopen('out_int.txt','at+'); % write output to a file %

    % Zi value %
        fprintf('ZireqT11=ZireqT12=ZireqT21=ZireqT22= Value: real and imaginer part = %f\n',real(ZireqT11)*50,imag(ZireqT11)*50);
        fprintf(fid_w,'ZireqT11=ZireqT12=ZireqT21=ZireqT22= Value: real and imaginer part = %f\n',real(ZireqT11)*50,imag(ZireqT11)*50);

% ZLreq value %
ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);

    % ZLreq value %
    fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);

% ZSreq value %
ZSreq=Zopt;

    % ZSreq value %
    fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
    fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

    % fid_w will be open in empcal.m
    fclose(fid_w);

    ZLreq=ZLreq1;  % print the first one in the intersection %


elseif (InstersectionExist ~= 0) & (Freq == Fmin)   % T1 and T1 identical, Zs=Zopt

% open fid_w for append
fid_w = fopen('out_int.txt','at+'); % write output to a file %

    % Zi value %
        fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
        fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

% ZLreq value %
ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

    % ZLreq value %
    fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
    fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

% ZSreq value %
```

```
ZSreq=Zopt;

    % ZSreq value %
    fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
    fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

    % fid_w will be open in empcal.m
    fclose(fid_w);

    ZLreq=ZLreq1;  % print the first one in the intersection %

elseif (InstersectionExist ~= 0) & (Vireq == 1)  % T1 and T1 identical, Zs=conj(Zi)

  % open fid_w for append
  fid_w = fopen('out_int.txt','at+'); % write output to a file %

    % Zi value %
     fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real
and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
      fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22
Value: real and imaginer part = %f
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

  % ZLreq value %
  ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
  ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

    % ZLreq value %
    fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
    fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

  % ZSreq value %
  ZSreq1=conj(ZireqT11);
  ZSreq2=conj(ZireqT12);

    % ZSreq value %
    fprintf('ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
    fprintf(fid_w,'ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
    fprintf('ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);
    fprintf(fid_w,'ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);

    % fid_w will be open in empcal.m
    fclose(fid_w);

    ZLreq=ZLreq1;  % print the first one in the intersection %
    ZSreq=ZSreq1;  % print the first one in the intersection %

  else

  % open fid_w for append
  fid_w = fopen('out_int.txt','at+'); % write output to a file %
```

```
        fprintf('No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');
        fprintf(fid_w,'No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');

            % fid_w will be open in empcal.m
            fclose(fid_w);

        ZSreq= NaN+i*NaN;
        ZLreq= NaN+i*NaN;

      end;

%         fprintf(fid_w_g,'%f       %f       %f       %f       %f       %f
          %f\n',f,GTMAX,GTMIN,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50); %
          fprintf(fid_w_g,'%f       %f       %f       %f       %f       %f       %f       %f       %f
          %f       %f
          %f\n',Vireq,FdB,f,GTMAX,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50,GTDBreq,real(ZLreq)
*50,imag(ZLreq)*50,real(ZSreq)*50,imag(ZSreq)*50);

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

    elseif (abs(ZCGmax-Zct1) < rt1) & (rt2 < abs(ZCGmax-Zct2))

          fprintf('(abs(ZCGmax-Zct1) < rt1) & (rt2 < abs(ZCGmax-Zct2)): ZCGmax in Region-3\n');
          fprintf(fid_w,'(abs(ZCGmax-Zct1) < rt1) & (rt2 < abs(ZCGmax-Zct2)): ZCGmax in Region-3\n');

      %  Zcgmax in Region-3; GTmax=GTMAX %
      %  GTMAX value; absolute stability and Zcgmax in Region-3 %
          GTMC=(Q-sqrt(Q^2-abs(Z)^2)*C)/(abs(Z12)^2);

      % GTmax Circle (a point )%
      rgmax=0;
          ZCGTM= ZCGmax+rgmax*exp(i*teta);
          RCGTM=real(ZCGTM);
          XCGTM=imag(ZCGTM);

      %  Load and Source impedance calculation; Zcgmax in Region-3 %
          % Zi value %
      if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
        Zi=conj(Zopt);
      else
            Zi=ZCGmax;
          end

          fprintf('Zi Value: real and imaginer part =   %f %f\n',real(Zi)*50,imag(Zi)*50);
          fprintf(fid_w,'Zi Value: real and imaginer part =   %f %f\n',real(Zi)*50,imag(Zi)*50);

          % ZL value %
          ZL=((Z12*Z21)/(Z11-Zi)-Z22);

          fprintf('ZL Value: real and imaginer part =   %f %f\n',real(ZL)*50,imag(ZL)*50);
```

```
        fprintf(fid_w,'ZL Value: real and imaginer      part =   %f %f\n',real(ZL)*50,imag(ZL)*50);

        % ZS values %
        Rcv=(1+Rmod^2)*real(Zi)/C;
        Xcv=-imag(Zi);
        Zcv=Rcv+i*Xcv;
        rv=2*Rmod*real(Zi)/C;

        A=(Rcn-Rcv)/(Xcv-Xcn);
        B=(abs(Zi)^2-abs(Zopt)^2)/(2*(Xcv-Xcn));
        K=(Rcn+A*Xcn-A*B)/(1+A^2);
        D=(B^2+abs(Zopt)^2-2*Xcn*B)/(1+A^2);

        Rs1=K+sqrt(K^2-D);
        Xs1=A*Rs1+B;
        Rs2=K-sqrt(K^2-D);
        Xs2=A*Rs2+B;

    if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
      Zs1=Zopt;
      Zs2=Zopt;
    else
          Zs1=Rs1+i*Xs1;
          Zs2=Rs2+i*Xs2;
        end

        ZS=Zs1;

        fprintf('Zs1 Value: real and imaginer part =   %f %f\n',real(Zs1)*50,imag(Zs1)*50);
        fprintf(fid_w,'Zs1 Value: real and imaginer part =   %f %f\n',real(Zs1)*50,imag(Zs1)*50);

        fprintf('Zs2 Value: real and imaginer part =   %f %f\n',real(Zs2)*50,imag(Zs2)*50);
        fprintf(fid_w,'Zs2 Value: real and imaginer part =   %f %f\n',real(Zs2)*50,imag(Zs2)*50);

    %  MAG value; absolute stability and Zcgmax in Region-3 (Rmod=0--> C=1) %
        MAGndB=(Q-sqrt(Q^2-abs(Z)^2))/(abs(Z12)^2);
        MAG=10*log10(MAGndB);  % MAG in dB %

        fprintf('MAG Value (dB) = %f\n',MAG);
        %fprintf(fid_w,'MAG Value (dB) = %f\n',MAG);

    %  GTMAX value; absolute stability and Zcgmax in Region-3 %
    GTMAX=10*log10(GTMC);
    GTMIN=0; % in dB %

%     if GTMAX > MAG     % MAG is the available maximum gain %
%       GTMAX=MAG;
%     end;

    fprintf('GTMAX Value (dB) = %f\n',GTMAX);
        fprintf(fid_w,'GTMAX Value (dB) = %f\n',GTMAX);
```

```
        fprintf('GTMIN      Value      (dB)      =      %f\n',GTMIN);
        fprintf(fid_w,'GTMIN Value (dB) = %f\n',GTMIN);

    %  Define values to plot f-GTmax %
    x_f(counts)=f;
    y_gtmax(counts)=GTMAX;
    y_gtreq(counts)=GTDBreq;
    z_gtmin(counts)=GTMIN;


        % fid_w will be open in empcal.m
        fclose(fid_w);

        % Caluclate empedances on T1 and T2 intersection points and other points in Region-3
        if (InstersectionExist ~= 0) & (Vireq ~= 1) & (Freq ~= Fmin)   % go to claculation in Region-3


[ZLreq,ZSreq]=empcal(ZireqT11,ZireqT12,ZireqT21,ZireqT22,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn
,RCGreq,XCGreq,rgreq,Rct2,Xct2,rt2,Rct1,Xct1,rt1,Ri1,Xi1,Ri2,Xi2,step_for_sample_on_circle,switch_for_plo
t)

    elseif (InstersectionExist ~= 0) & (Vireq == 1) & (Freq == Fmin)  % T1,T2,Vi.F  are a point

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

          % Zi value %
                fprintf('ZireqT11=ZireqT12=ZireqT21=ZireqT22=  Value:  real  and  imaginer  part  =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);
              fprintf(fid_w,'ZireqT11=ZireqT12=ZireqT21=ZireqT22=  Value:  real  and  imaginer  part  =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);

      % ZLreq value %
      ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);

          % ZLreq value %
          fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
          fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);

      % ZSreq value %
      ZSreq=Zopt;

          % ZSreq value %
          fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
          fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

          % fid_w will be open in empcal.m
          fclose(fid_w);

          ZLreq=ZLreq1;  % print the first one in the intersection %


    elseif (InstersectionExist ~= 0) & (Freq == Fmin)   % T1 and T1 identical, Zs=Zopt
```

```
% open fid_w for append
fid_w = fopen('out_int.txt','at+'); % write output to a file %

    % Zi value %
    fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
    fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

% ZLreq value %
ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

    % ZLreq value %
    fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
    fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

% ZSreq value %
ZSreq=Zopt;

    % ZSreq value %
    fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
    fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

    % fid_w will be open in empcal.m
    fclose(fid_w);

    ZLreq=ZLreq1;  % print the first one in the intersection %

elseif (InstersectionExist ~= 0) & (Vireq == 1)  % T1 and T1 identical, Zs=conj(Zi)

% open fid_w for append
fid_w = fopen('out_int.txt','at+'); % write output to a file %

    % Zi value %
    fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
    fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

% ZLreq value %
ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

    % ZLreq value %
    fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
```

```
            fprintf(fid_w,'ZLreq2 Value: real and imaginer part = %f
%f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

        % ZSreq value %
        ZSreq1=conj(ZireqT11);
        ZSreq2=conj(ZireqT12);

            % ZSreq value %
            fprintf('ZSreq1 Value: real and imaginer part = %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
            fprintf(fid_w,'ZSreq1 Value: real and imaginer part = %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
            fprintf('ZSreq2 Value: real and imaginer part = %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);
            fprintf(fid_w,'ZSreq2 Value: real and imaginer part = %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);

            % fid_w will be open in empcal.m
            fclose(fid_w);

            ZLreq=ZLreq1; % print the first one in the intersection %
            ZSreq=ZSreq1; % print the first one in the intersection %

          else

        % open fid_w for append
        fid_w = fopen('out_int.txt','at+'); % write output to a file %

        fprintf('No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');
        fprintf(fid_w,'No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');

            % fid_w will be open in empcal.m
            fclose(fid_w);

        ZSreq= NaN+i*NaN;
        ZLreq= NaN+i*NaN;

      end;

%       fprintf(fid_w_g,'%f %f %f %f %f %f
        %f\n',f,GTMAX,GTMIN,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50); %
        fprintf(fid_w_g,'%f %f %f %f %f %f %f %f %f %f
        %f %f
        %f\n',Vireq,FdB,f,GTMAX,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50,GTDBreq,real(ZLreq)
*50,imag(ZLreq)*50,real(ZSreq)*50,imag(ZSreq)*50);

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

    elseif abs(ZCGmax-Zct2) == rt2

        fprintf('abs(ZCGmax-Zct2) == rt2: ZCGmax in Region-4\n');
        fprintf(fid_w,'abs(ZCGmax-Zct2) == rt2: ZCGmax in Region-4\n');

      % Zcgmax in Region-4; GTmax-->T2 %
      % GTMAX value; absolute stability and Zcgmax in Region-4 %
        GTMC=(Q-sqrt(Q^2-abs(Z)^2)*C)/(abs(Z12)^2);
```

```
% GTmax Circle (a point )%
rgmax=0;
    ZCGTM= ZCGmax+rgmax*exp(i*teta);
    RCGTM=real(ZCGTM);
    XCGTM=imag(ZCGTM);

% Load and Source impedance calculation; Zcgmax in Region-4 %
    % Zi value %
if (Vireq == 1) & (Freq == Fmin)  % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
  Zi=conj(Zopt);
else
     Zi=ZCGmax;
    end

    fprintf('Zi Value: real and imaginer part =  %f %f\n',real(Zi)*50,imag(Zi)*50);
    fprintf(fid_w,'Zi Value: real and imaginer part =  %f %f\n',real(Zi)*50,imag(Zi)*50);

    % ZL value %
    ZL=((Z12*Z21)/(Z11-Zi)-Z22);

    fprintf('ZL Value: real and imaginer part =  %f %f\n',real(ZL)*50,imag(ZL)*50);
    fprintf(fid_w,'ZL Value: real and imaginer part =  %f %f\n',real(ZL)*50,imag(ZL)*50);

    % ZS values %
    Rcv=(1+Rmod^2)*real(Zi)/C;
    Xcv=-imag(Zi);
    Zcv=Rcv+i*Xcv;
    rv=2*Rmod*real(Zi)/C;
if (Vireq == 1) & (Freq == Fmin)  % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
  ZS=Zopt;
else
     if rn > rv
       ZS=(((rn/(rn-rv))*Zcv)-((rv/(rn-rv))*Zcn));
     else
       ZS=(((rv/(rv-rn))*Zcn)-((rn/(rv-rn))*Zcv));  % rn <= rv %
     end;
end

    fprintf('ZS Value: real and imaginer part =  %f %f\n',real(ZS)*50,imag(ZS)*50);
    fprintf(fid_w,'ZS Value: real and imaginer part =  %f %f\n',real(ZS)*50,imag(ZS)*50);

% MAG value; absolute stability and Zcgmax in Region-4 (Rmod=0--> C=1) %
    MAGndB=(Q-sqrt(Q^2-abs(Z)^2))/(abs(Z12)^2);
    MAG=10*log10(MAGndB);  % MAG in dB %

    fprintf('MAG Value (dB) = %f\n',MAG);
    %fprintf(fid_w,'MAG Value (dB) = %f\n',MAG);


% GTMAX value; absolute stability and Zcgmax in Region-4 %
```

```
GTMAX=10*log10(GTMC);
GTMIN=0;  % in dB %

%    if GTMAX > MAG     % MAG is the available maximum gain %
%      GTMAX=MAG;
%    end;

fprintf('GTMAX Value (dB) = %f\n',GTMAX);
    fprintf(fid_w,'GTMAX Value (dB) = %f\n',GTMAX);
    fprintf('GTMIN Value (dB) = %f\n',GTMIN);
    fprintf(fid_w,'GTMIN Value (dB) = %f\n',GTMIN);

% Define values to plot f-GTmax %
x_f(counts)=f;
y_gtmax(counts)=GTMAX;
y_gtreq(counts)=GTDBreq;
z_gtmin(counts)=GTMIN;


    % fid_w will be open in empcal.m
    fclose(fid_w);

    % Caluclate empedances on T1 and T2 intersection points and other points in Region-3
    if (InstersectionExist ~= 0) & (Vireq ~= 1) & (Freq ~= Fmin)   % go to claculation in Region-3


[ZLreq,ZSreq]=empcal(ZireqT11,ZireqT12,ZireqT21,ZireqT22,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn
,RCGreq,XCGreq,rgreq,Rct2,Xct2,rt2,Rct1,Xct1,rt1,Ri1,Xi1,Ri2,Xi2,step_for_sample_on_circle,switch_for_plo
t)

elseif (InstersectionExist ~= 0) & (Vireq == 1) & (Freq == Fmin)  % T1,T2,Vi.F  are a point

  % open fid_w for append
  fid_w = fopen('out_int.txt','at+'); % write output to a file %

    % Zi value %
        fprintf('ZireqT11=ZireqT12=ZireqT21=ZireqT22=  Value: real and imaginer part =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);
       fprintf(fid_w,'ZireqT11=ZireqT12=ZireqT21=ZireqT22=  Value: real and imaginer part =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);

  % ZLreq value %
  ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);

    % ZLreq value %
    fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);

  % ZSreq value %
  ZSreq=Zopt;

    % ZSreq value %
    fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
```

```
            fprintf(fid_w,'ZSreq Value: real and imaginer part = %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

            % fid_w will be open in empcal.m
            fclose(fid_w);

            ZLreq=ZLreq1;  % print the first one in the intersection %


      elseif (InstersectionExist ~= 0) & (Freq == Fmin)   % T1 and T1 identical, Zs=Zopt

        % open fid_w for append
        fid_w = fopen('out_int.txt','at+'); % write output to a file %

            % Zi value %
            fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
            fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

        % ZLreq value %
        ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
        ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

            % ZLreq value %
            fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
            fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
            fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
            fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

        % ZSreq value %
        ZSreq=Zopt;

            % ZSreq value %
            fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
            fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

            % fid_w will be open in empcal.m
            fclose(fid_w);

            ZLreq=ZLreq1;  % print the first one in the intersection %

      elseif (InstersectionExist ~= 0) & (Vireq == 1)   % T1 and T1 identical, Zs=conj(Zi)

        % open fid_w for append
        fid_w = fopen('out_int.txt','at+'); % write output to a file %

            % Zi value %
            fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
            fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
```

```
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

        % ZLreq value %
        ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
        ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

            % ZLreq value %
            fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
            fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
            fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
            fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

        % ZSreq value %
        ZSreq1=conj(ZireqT11);
        ZSreq2=conj(ZireqT12);

            % ZSreq value %
            fprintf('ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
            fprintf(fid_w,'ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
            fprintf('ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);
            fprintf(fid_w,'ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);

            % fid_w will be open in empcal.m
            fclose(fid_w);

            ZLreq=ZLreq1;  % print the first one in the intersection %
            ZSreq=ZSreq1;  % print the first one in the intersection %

          else

        % open fid_w for append
        fid_w = fopen('out_int.txt','at+'); % write output to a file %

        fprintf('No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');
        fprintf(fid_w,'No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');

            % fid_w will be open in empcal.m
            fclose(fid_w);

        ZSreq= NaN+i*NaN;
        ZLreq= NaN+i*NaN;

      end;

%       fprintf(fid_w_g,'%f      %f      %f      %f      %f      %f
        %f\n',f,GTMAX,GTMIN,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50); %
        fprintf(fid_w_g,'%f      %f      %f      %f      %f      %f      %f      %f      %f
        %f      %f
        %f\n',Vireq,FdB,f,GTMAX,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50,GTDBreq,real(ZLreq)
*50,imag(ZLreq)*50,real(ZSreq)*50,imag(ZSreq)*50);

      % open fid_w for append
```

```
    fid_w = fopen('out_int.txt','at+'); % write output      to a file %


  elseif abs(ZCGmax-Zct2) < rt2

      fprintf('abs(ZCGmax-Zct2) < rt2: ZCGmax in Region-5 (tangent case --> GTmax)\n');
      fprintf(fid_w,'abs(ZCGmax-Zct2) < rt2: ZCGmax in Region-5 (tangent case --> GTmax)\n');

    % Zcgmax in Region-5 %
    % GTMAX value; absolute stability and Zcgmax in Region-5 (tangent case --> GTmax) %

    if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
      GTMC=(Q-sqrt(Q^2-abs(Z)^2)*C)/(abs(Z12)^2);
    else
        D=(R22*abs(Z11)^2-R*R11-X*X11)/R22+abs(Zopt)^2;
        E=(D*R22-Rct2*(2*R11*R22-R)-2*Xct2*XCG*R22)/(-rt2);
        F=Rct2/(-rt2);

        GTpayn=C*(Q+E*F-sqrt(((Q+E*F)^2-(1-F^2)*(abs(Z)^2-E^2))));
        GTpayda=(abs(Z12)^2*(1-F^2));
        GTMCn=GTpayn/GTpayda;

        GTpayp=C*(Q+E*F+sqrt(((Q+E*F)^2-(1-F^2)*(abs(Z)^2-E^2))));
        GTpayda=(abs(Z12)^2*(1-F^2));
        GTMCp=GTpayp/GTpayda;

      if GTMCn > GTMCp
        GTMC=GTMCn;
        GTmC=GTMCp;
      else
        GTMC=GTMCp;
        GTmC=GTMCn;
      end;

      % Checking negative sign for GTmC %
      if GTmC < 0
        GT_Negative=1;
        GTmC=abs(GTmC);
      end
    end

      P=abs(Z12)^2/C*GTMC;

      RCGmax = (Q - P)/(2*R22);
  XCGmax = XCG;
  ZCGmax = RCGmax + i*XCGmax;
      rgmax = (sqrt(P^2-2*Q*P+abs(Z)^2))/(2*R22);

  % GTmax Circle %
      ZCGTM= ZCGmax+rgmax*exp(i*teta);
      RCGTM=real(ZCGTM);
```

```
        XCGTM=imag(ZCGTM);

    % Load and Source impedance calculation; Zcgmax in Region-5 %
        % Zi value %
    if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
      Zi=conj(Zopt);
    else
         Zi=(((rgmax/(rgmax-rt2))*Zct2)-((rt2/(rgmax-rt2))*ZCGmax));
    end

        fprintf('Zi Value: real and imaginer part =  %f %f\n',real(Zi)*50,imag(Zi)*50);
        fprintf(fid_w,'Zi Value: real and imaginer part =  %f %f\n',real(Zi)*50,imag(Zi)*50);

        % ZL value %
        ZL=((Z12*Z21)/(Z11-Zi)-Z22);

        fprintf('ZL Value: real and imaginer part =  %f %f\n',real(ZL),imag(ZL));
        fprintf(fid_w,'ZL Value: real and imaginer part =  %f %f\n',real(ZL),imag(ZL));

        % ZS values %
        Rcv=(1+Rmod^2)*real(Zi)/C;
        Xcv=-imag(Zi);
        Zcv=Rcv+i*Xcv;
        rv=2*Rmod*real(Zi)/C;
    if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
      ZS=Zopt;
    else
         if rn > rv
           ZS=(((rn/(rn-rv))*Zcv)-((rv/(rn-rv))*Zcn));
         else
           ZS=(((rv/(rv-rn))*Zcn)-((rn/(rv-rn))*Zcv));  % rn <= rv %
         end;
    end

        fprintf('ZS Value: real and imaginer part =  %f %f\n',real(ZS)*50,imag(ZS)*50);
        fprintf(fid_w,'ZS Value: real and imaginer part =  %f %f\n',real(ZS)*50,imag(ZS)*50);

    % MAG value; absolute stability and Zcgmax in Region-5 (Rmod=0--> C=1) %
        MAGndB=(Q-sqrt(Q^2-abs(Z)^2))/(abs(Z12)^2);
        MAG=10*log10(MAGndB);  % MAG in dB %

        fprintf('MAG Value (dB) = %f\n',MAG);
        %fprintf(fid_w,'MAG Value (dB) = %f\n',MAG);

    % GTMAX value; absolute stability and Zcgmax in Region-5 %
    GTMAX=10*log10(GTMC);
    if GT_Negative == 1   % Negative GT %
      GTMIN=-10*log10(GTmC);
    else
      GTMIN=10*log10(GTmC);
    end
```

```
%     if GTMAX > MAG     % MAG is the available maximum gain %
%       GTMAX=MAG;
%     end;

    fprintf('GTMAX Value (dB) = %f\n',GTMAX);
       fprintf(fid_w,'GTMAX Value (dB) = %f\n',GTMAX);
       fprintf('GTMIN Value (dB) = %f\n',GTMIN);
       fprintf(fid_w,'GTMIN Value (dB) = %f\n',GTMIN);

    %  Define values to plot f-GTmax %
    x_f(counts)=f;
    y_gtmax(counts)=GTMAX;
    y_gtreq(counts)=GTDBreq;
    z_gtmin(counts)=GTMIN;


       % fid_w will be open in empcal.m
       fclose(fid_w);

       % Caluclate empedances on T1 and T2 intersection points and other points in Region-3
       if (InstersectionExist ~= 0) & (Vireq ~= 1) & (Freq ~= Fmin)   % go to claculation in Region-3


[ZLreq,ZSreq]=empcal(ZireqT11,ZireqT12,ZireqT21,ZireqT22,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn
,RCGreq,XCGreq,rgreq,Rct2,Xct2,rt2,Rct1,Xct1,rt1,Ri1,Xi1,Ri2,Xi2,step_for_sample_on_circle,switch_for_plo
t)

    elseif (InstersectionExist ~= 0) & (Vireq == 1) & (Freq == Fmin)  % T1,T2,Vi.F  are a point

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

         % Zi value %
              fprintf('ZireqT11=ZireqT12=ZireqT21=ZireqT22=  Value:  real  and  imaginer  part  =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);
            fprintf(fid_w,'ZireqT11=ZireqT12=ZireqT21=ZireqT22=  Value:  real  and  imaginer  part  =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);

      % ZLreq value %
      ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);

         % ZLreq value %
         fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
         fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);

      % ZSreq value %
      ZSreq=Zopt;

         % ZSreq value %
         fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
         fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
```

```
        % fid_w will be open in empcal.m
        fclose(fid_w);

        ZLreq=ZLreq1;  % print the first one in the intersection %


    elseif (InstersectionExist ~= 0) & (Freq == Fmin)   % T1 and T1 identical, Zs=Zopt

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

          % Zi value %
          fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real
and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
          fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22
Value:  real  and  imaginer  part  =  %f
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

      % ZLreq value %
      ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
      ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

          % ZLreq value %
          fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
          fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
          fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
          fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

      % ZSreq value %
      ZSreq=Zopt;

          % ZSreq value %
          fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
          fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

          % fid_w will be open in empcal.m
          fclose(fid_w);

          ZLreq=ZLreq1;  % print the first one in the intersection %

    elseif (InstersectionExist ~= 0) & (Vireq == 1)   % T1 and T1 identical, Zs=conj(Zi)

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

          % Zi value %
          fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real
and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
          fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22
Value:  real  and  imaginer  part  =  %f
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

      % ZLreq value %
```

```
        ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
        ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

            % ZLreq value %
            fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
            fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
            fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
            fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

        % ZSreq value %
        ZSreq1=conj(ZireqT11);
        ZSreq2=conj(ZireqT12);

            % ZSreq value %
            fprintf('ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
            fprintf(fid_w,'ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
            fprintf('ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);
            fprintf(fid_w,'ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);

            % fid_w will be open in empcal.m
            fclose(fid_w);

            ZLreq=ZLreq1;  % print the first one in the intersection %
            ZSreq=ZSreq1;  % print the first one in the intersection %

          else

        % open fid_w for append
        fid_w = fopen('out_int.txt','at+'); % write output to a file %

        fprintf('No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');
        fprintf(fid_w,'No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');

            % fid_w will be open in empcal.m
            fclose(fid_w);

        ZSreq= NaN+i*NaN;
        ZLreq= NaN+i*NaN;

      end;

%         fprintf(fid_w_g,'%f       %f       %f       %f       %f       %f
          %f\n',f,GTMAX,GTMIN,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50); %
          fprintf(fid_w_g,'%f       %f       %f       %f       %f       %f       %f       %f       %f       %f
          %f       %f
          %f\n',Vireq,FdB,f,GTMAX,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50,GTDBreq,real(ZLreq)
*50,imag(ZLreq)*50,real(ZSreq)*50,imag(ZSreq)*50);

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

    else
```

```
    fprintf('Not defined\n');
    fprintf(fid_w,'Not defined\n');

  end; % Conserning Solution %

  % Drawing Circles % % Ri41 --> GTmin so not drawn %
  if switch_for_plot == 1
    figure,plot(RCGTreq,XCGTreq,'k-',        Ri1,Xi1,'m-',Ri2,Xi2,'m-.',RCGTM,XCGTM,'b-',Ri5,Xi5,'g-
',Ri3,Xi3,'c-',Ri42,Xi42,'r:',Ri43,Xi43,'r:',Ri44,Xi44,'r:',Ri45,Xi45,'r:')
        axis([-1 1 -1 1])
        title('Zi-Plane')
        xlabel('real{Zi}'),ylabel('imag{Zi}'),grid
        gtext('GTreq'),gtext('T1'),gtext('T2'),gtext('GTmax'),gtext('ISC'),gtext('CSSC'),gtext('Gain
Circles'),gtext('Unconditional Stability')
  end

elseif R11>0 & R22>0 & ita>0 & ita<1   % Stability: elseif %

  %  Conditionally stable %

  fprintf('Conditionally stable: R11>0 & R22>0 & ita>0 & ita<1\n');
  fprintf(fid_w,'\nConditionally stable: R11>0 & R22>0 & ita>0 & ita<1\n');
  fprintf('R11,R22,ita = %f,%f,%f \n ',R11,R22,ita);
  fprintf(fid_w,'R11,R22,ita = %f,%f,%f\n',R11,R22,ita);

  % Calculating values for Required GT value in Conditional stability case

  S=COEFS*(10^(GTDBreq*0.1));          % GTDBreq converted to dimentionless %
  RCS = -(2*R11*R22-R)/(2*R22);
  XCS = -(2*X11*R22-X)/(2*R22);
  ZCS = RCS + i*XCS;
  rs = abs(Z)/(2*R22);
  RCGreq = -(RCS + S) ;
  XCGreq = -XCS;
  rgreq = sqrt(rs^2 + 2*S*RCS + S^2);
  ZCGreq = RCGreq + i*XCGreq;

  % GTDBreq Circle %
  ZCGTreq= ZCGreq+rgreq*exp(i*teta);
  RCGTreq=real(ZCGTreq);
  XCGTreq=imag(ZCGTreq);


  % Find out the intersection of two circles
  InstersectionExist=1;
  fprintf('GTDBreq, T1 and T2 circles:\n');
  fprintf('RCGreq= %f, XCGreq= %f, rgreq= %f\n',RCGreq,XCGreq,rgreq);
  fprintf('Rct1= %f, Xct1= %f, rt1= %f\n',Rct1,Xct1,rt1);
  fprintf('Rct2= %f, Xct2= %f, rt2= %f\n',Rct2,Xct2,rt2);
  fprintf(fid_w,'GTDBreq, T1 and T2 circles:\n');
  fprintf(fid_w,'RCGreq= %f, XCGreq= %f, rgreq= %f\n',RCGreq,XCGreq,rgreq);
  fprintf(fid_w,'Rct1= %f, Xct1= %f, rt1= %f\n',Rct1,Xct1,rt1);
  fprintf(fid_w,'Rct2= %f, Xct2= %f, rt2= %f\n',Rct2,Xct2,rt2);
```

```
    [RintT1,XintT1]=circlint(RCGreq,XCGreq,rgreq,Rct1,Xct1,rt1);
    ZireqT11=RintT1(1)+i*XintT1(1);
    ZireqT12=RintT1(2)+i*XintT1(2);
%     AlfaT11=atan(XintT1(1)/RintT1(1))*180/pi;
%     AlfaT12=atan(XintT1(2)/RintT1(2))*180/pi;

    fprintf('1. Intersection point on T1; RintT1= %f: XintT1= %f\n',RintT1(1),XintT1(1));
    fprintf(fid_w,'1. Intersection point on T1; RintT1= %f: XintT1= %f\n',RintT1(1),XintT1(1));
%     fprintf('Line Slope for 1. Intersection point on T1; AlfaT11(deg)= %f\n',AlfaT11);
%     fprintf(fid_w,'Line Slope for 1. Intersection point on T1; AlfaT11(deg)= %f\n',AlfaT11);
    fprintf('2. Intersection point on T1; RintT1= %f: XintT1= %f\n',RintT1(2),XintT1(2));
    fprintf(fid_w,'2. Intersection point on T1; RintT1= %f: XintT1= %f\n',RintT1(2),XintT1(2));
%     fprintf('Line Slope for 2. Intersection point on T1; AlfaT12(deg)= %f\n',AlfaT12);
%     fprintf(fid_w,'Line Slope for 2. Intersection point on T1; AlfaT12(deg)= %f\n',AlfaT12);

    fprintf('ZireqT11 Value: real and imaginer part =  %f %f\n',RintT1(1)*50,XintT1(1)*50);
    fprintf(fid_w,'ZireqT11 Value: real and imaginer part =  %f %f\n',RintT1(1)*50,XintT1(1)*50);
    fprintf('ZireqT12 Value: real and imaginer part =  %f %f\n',RintT1(2)*50,XintT1(2)*50);
    fprintf(fid_w,'ZireqT12 Value: real and imaginer part =  %f %f\n',RintT1(2)*50,XintT1(2)*50);



%     [cor_x,cor_y]=linecirc(AlfaT11,XintT1(1),RCGreq,XCGreq,rgreq)

%     fprintf('%f\n',cor_x);
%     fprintf('%f\n',cor_y);


    [RintT2,XintT2]=circlint(RCGreq,XCGreq,rgreq,Rct2,Xct2,rt2);
    ZireqT21=RintT2(1)+i*XintT2(1);  % two intersection points on T2 %
    ZireqT22=RintT2(2)+i*XintT2(2);
%     AlfaT21=atan(XintT2(1)/RintT2(1))*180/pi;
%     AlfaT22=atan(XintT2(2)/RintT2(2))*180/pi;

    fprintf('1. Intersection point on T2; RintT2= %f: XintT2= %f\n',RintT2(1),XintT2(1));
    fprintf(fid_w,'1. Intersection point on T2; RintT1= %f: XintT2= %f\n',RintT2(1),XintT2(1));
%     fprintf('Line Slope for 1. Intersection point on T2; AlfaT21(deg)= %f\n',AlfaT21);
%     fprintf(fid_w,'Line Slope for 1. Intersection point on T2; AlfaT21(deg)= %f\n',AlfaT21);
    fprintf('2. Intersection point on T2; RintT2= %f: XintT2= %f\n',RintT2(2),XintT2(2));
    fprintf(fid_w,'2. Intersection point on T2; RintT2= %f: XintT2= %f\n',RintT2(2),XintT2(2));
%     fprintf('Line Slope for 2. Intersection point on T2; AlfaT22(deg)= %f\n',AlfaT22);
%     fprintf(fid_w,'Line Slope for 2. Intersection point on T2; AlfaT22(deg)= %f\n',AlfaT22);

    fprintf('ZireqT21 Value: real and imaginer part =  %f %f\n',RintT2(1)*50,XintT2(1)*50);
    fprintf(fid_w,'ZireqT21 Value: real and imaginer part =  %f %f\n',RintT2(1)*50,XintT2(1)*50);
    fprintf('ZireqT22 Value: real and imaginer part =  %f %f\n',RintT2(2)*50,XintT2(2)*50);
    fprintf(fid_w,'ZireqT22 Value: real and imaginer part =  %f %f\n',RintT2(2)*50,XintT2(2)*50);


    if ~isnan(ZireqT11) & ~isnan(ZireqT12) & ~isnan(ZireqT21) & ~isnan(ZireqT22)   % GTreq cuts both T1
```

```
and T2
      fprintf('GTreq cuts both T1 and T2\n');
      fprintf(fid_w,'GTreq cuts both T1 and T2\n');
    elseif ~isnan(ZireqT11) & ~isnan(ZireqT12)                    % GTreq cuts only T1
      fprintf('GTreq cuts only T1\n');
      fprintf(fid_w,'GTreq cuts only T1\n');
    elseif ~isnan(ZireqT21) & ~isnan(ZireqT22)                    % GTreq cuts only T2
      fprintf('GTreq cuts only T2\n');
      fprintf(fid_w,'GTreq cuts only T2\n');
    else                                                  % GTreq does not cut T1 or T2
      fprintf('GTreq does not cut T1 or T2\n');
      fprintf(fid_w,'GTreq does not cut T1 or T2\n');
      InstersectionExist=0;
      ZSreq= NaN+i*NaN;
      ZLreq= NaN+i*NaN;

%       break;
    end


    % Conjugate Source Stability Circle (GTmax Circle)%
    RCS = -(2*R11*R22-R)/(2*R22);
    XCS = -(2*X11*R22-X)/(2*R22);
    ZCS = RCS + i*XCS;
    rs = abs(Z)/(2*R22);
    ZCSC = conj(ZCS)

    Zi3 = ZCSC +  rs * exp(i*teta);
    Ri3 = real(Zi3);
    Xi3 = imag(Zi3);

    % Input Stability Circle (ZCINPUT = -RCS - i*XCS; rINPUT=rs) (GTmin Circle) %
    ZCINPUT = -RCS - i*XCS;
    Zi5 = ZCINPUT + rs * exp(i*teta);
    Ri5 = real(Zi5);
    Xi5 = imag(Zi5);

    % GT Circles %
    GTMAX = 2*C*Q/(abs(Z12)^2);

    for k = 1:6
      y(k) = (k-1)*(GTMAX/5);
      S(k) = y(k)*COEFS;
      RCG(k) = -(RCS + S(k)) ;
      XCG = -XCS;
      rg(k) = sqrt(rs^2 + 2*S(k)*RCS + S(k)^2);
      ZCG(k) = RCG(k) + i*XCG;
    end

    ZCG1 = ZCG(1);rg1 = rg(1);
    ZCG2 = ZCG(2);rg2 = rg(2);
    ZCG3 = ZCG(3);rg3 = rg(3);
```

```
ZCG4 = ZCG(4);rg4 = rg(4);
ZCG5 = ZCG(5);rg5 = rg(5);
ZCG6 = ZCG(6);rg6 = rg(6);

Zi41 = ZCG1 + rg1 * exp(i*teta);
  Ri41 = real(Zi41);Xi41 = imag(Zi41);
Zi42 = ZCG2 + rg2 * exp(i*teta);
  Ri42 = real(Zi42);Xi42 = imag(Zi42);
Zi43 = ZCG3 + rg3 * exp(i*teta);
  Ri43 = real(Zi43);Xi43 = imag(Zi43);
Zi44 = ZCG4 + rg4 * exp(i*teta);
  Ri44 = real(Zi44);Xi44 = imag(Zi44);
Zi45 = ZCG5 + rg5 * exp(i*teta);
  Ri45 = real(Zi45);Xi45 = imag(Zi45);
Zi46 = ZCG6 + rg6 * exp(i*teta);
  Ri46 = real(Zi46);Xi46 = imag(Zi46);


% Conserning Solution %
ZCGmin=ZCINPUT;  % rs=rINPUT=rgmin=rgmax  and  ZCGmin=ZCINPUT=-RCS-iXCS  and
ZCGmax=ZCSC=RCS-iXCS %
ZCGmax=ZCSC;
ZCGTM=0;  % to draw GTmax for tangential case b) %

fprintf('ZCGmax(%f %f), ZCGmin(%f %f), Zct1(%f %f), Zct2(%f %f)\n',real(ZCGmax),imag(ZCGmax),
real(ZCGmin), imag(ZCGmin),Rct1, Xct1, Rct2, Xct2);
fprintf('rs %f, rt1 %f, rt2 %f\n',rs,rt1, rt2);
fprintf(fid_w,'ZCGmax(%f  %f),  ZCGmin(%f  %f),  Zct1(%f  %f),  Zct2(%f
%f)\n',real(ZCGmax),imag(ZCGmax),real(ZCGmin), imag(ZCGmin), Rct1, Xct1, Rct2, Xct2);
fprintf(fid_w,'rs %f, rt1 %f, rt2 %f\n',rs,rt1, rt2);


if abs(ZCGmin-Zct1) > (rs+rt1)  % Conserning Solution %
% ?? eşitlik %
 fprintf('abs(ZCGmin-Zct1) > (rs+rt1): Case a) GT=0 is completely in Region-1; No solution\n');
 fprintf(fid_w,'abs(ZCGmin-Zct1) > (rs+rt1): Case a) GT=0 is completely in Region-1; No solution\n');


 % Case a) GT=0 is completely in Region-1; No solution %

 ZS= NaN+i*NaN;
 ZL= NaN+i*NaN;

    % fid_w will be open in empcal.m
    fclose(fid_w);

    % Caluclate empedances on T1 and T2 intersection points and other points in Region-3
    if (InstersectionExist ~= 0) & (Vireq ~= 1) & (Freq ~= Fmin)  % go to claculation in Region-3


[ZLreq,ZSreq]=empcal(ZireqT11,ZireqT12,ZireqT21,ZireqT22,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn
,RCGreq,XCGreq,rgreq,Rct2,Xct2,rt2,Rct1,Xct1,rt1,Ri1,Xi1,Ri2,Xi2,step_for_sample_on_circle,switch_for_plo
t)
```

```
elseif (InstersectionExist ~= 0) & (Vireq == 1) & (Freq == Fmin)  % T1,T2,Vi.F  are a point

    % open fid_w for append
    fid_w = fopen('out_int.txt','at+'); % write output to a file %

        % Zi value %
            fprintf('ZireqT11=ZireqT12=ZireqT21=ZireqT22=  Value:  real  and  imaginer  part  =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);
          fprintf(fid_w,'ZireqT11=ZireqT12=ZireqT21=ZireqT22=  Value:  real  and  imaginer  part  =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);

    % ZLreq value %
    ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);

        % ZLreq value %
        fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);

    % ZSreq value %
    ZSreq=Zopt;

        % ZSreq value %
        fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
        fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

        % fid_w will be open in empcal.m
        fclose(fid_w);

        ZLreq=ZLreq1;  % print the first one in the intersection %


elseif (InstersectionExist ~= 0) & (Freq == Fmin)   % T1 and T1 identical, Zs=Zopt

    % open fid_w for append
    fid_w = fopen('out_int.txt','at+'); % write output to a file %

        % Zi value %
         fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real
and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
          fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22
Value:  real  and  imaginer  part  =  %f
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

    % ZLreq value %
    ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
    ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

        % ZLreq value %
        fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
        fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
```

```
% ZSreq value %
ZSreq=Zopt;

    % ZSreq value %
    fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
    fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

    % fid_w will be open in empcal.m
    fclose(fid_w);

    ZLreq=ZLreq1;  % print the first one in the intersection %

elseif (InstersectionExist ~= 0) & (Vireq == 1)  % T1 and T1 identical, Zs=conj(Zi)

% open fid_w for append
fid_w = fopen('out_int.txt','at+'); % write output to a file %

    % Zi value %
    fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real
and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
     fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22
Value:  real  and  imaginer  part  =  %f
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

% ZLreq value %
ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

    % ZLreq value %
    fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
    fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

% ZSreq value %
ZSreq1=conj(ZireqT11);
ZSreq2=conj(ZireqT12);

    % ZSreq value %
    fprintf('ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
    fprintf(fid_w,'ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
    fprintf('ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);
    fprintf(fid_w,'ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);

    % fid_w will be open in empcal.m
    fclose(fid_w);

    ZLreq=ZLreq1;  % print the first one in the intersection %
    ZSreq=ZSreq1;  % print the first one in the intersection %

  else
```

```
        % open fid_w for append
        fid_w = fopen('out_int.txt','at+'); % write output to a file %

        fprintf('No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');
        fprintf(fid_w,'No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');

            % fid_w will be open in empcal.m
            fclose(fid_w);

        ZSreq= NaN+i*NaN;
        ZLreq= NaN+i*NaN;

      end;

%         fprintf(fid_w_g,'%f      %f      %f      %f      %f      %f
          %f\n',f,GTMAX,GTMIN,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50); %
          fprintf(fid_w_g,'%f      %f      %f      %f      %f      %f      %f      %f      %f
          %f      %f
          %f\n',Vireq,FdB,f,GTMAX,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50,GTDBreq,real(ZLreq)
*50,imag(ZLreq)*50,real(ZSreq)*50,imag(ZSreq)*50);

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

     elseif (abs(ZCGmax-Zct1) > (rs+rt1)) & ((rs+rt1) > abs(ZCGmin-Zct1))

          fprintf('(abs(ZCGmax-Zct1) > (rs+rt1)) & ((rs+rt1) > abs(ZCGmin-Zct1)): Case b) GTmax is external
tangential to T1\n');
          fprintf(fid_w,'(abs(ZCGmax-Zct1) > (rs+rt1)) & ((rs+rt1) > abs(ZCGmin-Zct1)): Case b) GTmax is
external tangential to T1\n');

      % Case b) GTmax is external tangential to T1 %
      % GTMAX value; Case b) GTmax is external tangential to T1 %

      if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
        GTMC=2*C*Q/(abs(Z12)^2);
      else
           D=(R22*abs(Z11)^2-R*R11-X*X11)/R22+abs(Zopt)^2;
           E=(D*R22-Rct1*(2*R11*R22-R)-2*Xct1*XCG*R22)/rt1;
           F=Rct1/rt1;

           GTpayn=C*(Q+E*F-sqrt(((Q+E*F)^2-(1-F^2)*(abs(Z)^2-E^2))));
           GTpayda=(abs(Z12)^2*(1-F^2));
           GTMC=GTpayn/GTpayda;
      end

      S=COEFS*GTMC;

      RCGmax=-(RCS+S);
      XCGmax=-XCS;
          ZCGmax = RCGmax + i*XCGmax;
      rgmax = sqrt(S^2+ 2*S*RCS + rs^2);
```

```
% GTmax Circle %
    ZCGTM= ZCGmax+rgmax*exp(i*teta);
    RCGTM=real(ZCGTM);
    XCGTM=imag(ZCGTM);


% Load and Source impedance calculation; Case b) GTmax is external tangential to T1 %
    % Zi value %
if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
  Zi=conj(Zopt);
else
     Zi=(((rgmax/(rt1+rgmax))*Zct1)+((rt1/(rt1+rgmax))*ZCGmax));
end

    fprintf('Zi Value: real and imaginer part =  %f %f\n',real(Zi)*50,imag(Zi)*50);
    fprintf(fid_w,'Zi Value: real and imaginer part =  %f %f\n',real(Zi)*50,imag(Zi)*50);

    % ZL value %
    ZL=((Z12*Z21)/(Z11-Zi)-Z22);

    fprintf('ZL Value: real and imaginer part =  %f %f\n',real(ZL)*50,imag(ZL)*50);
    fprintf(fid_w,'ZL Value: real and imaginer part =  %f %f\n',real(ZL)*50,imag(ZL)*50);

    % ZS values %
    Rcv=(1+Rmod^2)*real(Zi)/C;
    Xcv=-imag(Zi);
    Zcv=Rcv+i*Xcv;
    rv=2*Rmod*real(Zi)/C;

if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
  ZS=Zopt;
else
     ZS=(((rn/(rn+rv))*Zcv)+((rv/(rv+rn))*Zcn));
    end

    fprintf('ZS Value: real and imaginer part =  %f %f\n',real(ZS)*50,imag(ZS)*50);
    fprintf(fid_w,'ZS Value: real and imaginer part =  %f %f\n',real(ZS)*50,imag(ZS)*50);

% MSG value; Case b) GTmax is external tangential to T1 %
    MSGndB=2*(abs(Z21))/(abs(Z12))*ita;
    MSG=10*log10(MSGndB);  % MSG in dB %

    fprintf('MSG Value (dB) = %f\n',MSG);
    %fprintf(fid_w,'MSG Value (dB) = %f\n',MSG);


% GTMAX value; Case b) GTmax is external tangential to T1 %
GTMAX=10*log10(GTMC);
GTMIN=0; % in dB %
```

```
%       if GTMAX > MSG      % MSG is maximum      stable gain %
%        GTMAX=MSG;
%      end;

      fprintf('GTMAX Value (dB) = %f\n',GTMAX);
          fprintf(fid_w,'GTMAX Value (dB) = %f\n',GTMAX);
          fprintf('GTMIN Value (dB) = %f\n',GTMIN);
          fprintf(fid_w,'GTMIN Value (dB) = %f\n',GTMIN);

      %  Define values to plot f-GTmax %
      x_f(counts)=f;
      y_gtmax(counts)=GTMAX;
      y_gtreq(counts)=GTDBreq;
      z_gtmin(counts)=GTMIN;

          % fid_w will be open in empcal.m
          fclose(fid_w);

          % Caluclate empedances on T1 and T2 intersection points and other points in Region-3
          if (InstersectionExist ~= 0) & (Vireq ~= 1) & (Freq ~= Fmin)   % go to claculation in Region-3


[ZLreq,ZSreq]=empcal(ZireqT11,ZireqT12,ZireqT21,ZireqT22,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn
,RCGreq,XCGreq,rgreq,Rct2,Xct2,rt2,Rct1,Xct1,rt1,Ri1,Xi1,Ri2,Xi2,step_for_sample_on_circle,switch_for_plo
t)

      elseif (InstersectionExist ~= 0) & (Vireq == 1) & (Freq == Fmin)  % T1,T2,Vi.F  are a point

        % open fid_w for append
        fid_w = fopen('out_int.txt','at+'); % write output to a file %

            % Zi value %
                  fprintf('ZireqT11=ZireqT12=ZireqT21=ZireqT22=  Value:  real  and  imaginer  part  =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);
                fprintf(fid_w,'ZireqT11=ZireqT12=ZireqT21=ZireqT22=  Value:  real  and  imaginer  part  =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);

        % ZLreq value %
        ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);

            % ZLreq value %
            fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
            fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);

        % ZSreq value %
        ZSreq=Zopt;

            % ZSreq value %
            fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
            fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

            % fid_w will be open in empcal.m
            fclose(fid_w);
```

```
        ZLreq=ZLreq1;  % print the first one in the intersection %


    elseif (InstersectionExist ~= 0) & (Freq == Fmin)   % T1 and T1 identical, Zs=Zopt

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

          % Zi value %
          fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real
and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
          fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22
Value:  real  and  imaginer  part  =  %f
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

      % ZLreq value %
      ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
      ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

          % ZLreq value %
          fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
          fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
          fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
          fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

      % ZSreq value %
      ZSreq=Zopt;

          % ZSreq value %
          fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
          fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

          % fid_w will be open in empcal.m
          fclose(fid_w);

          ZLreq=ZLreq1;  % print the first one in the intersection %

    elseif (InstersectionExist ~= 0) & (Vireq == 1)   % T1 and T1 identical, Zs=conj(Zi)

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

          % Zi value %
          fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real
and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
          fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22
Value:  real  and  imaginer  part  =  %f
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

      % ZLreq value %
      ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
      ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);
```

```
            % ZLreq value %
            fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
            fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
            fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
            fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

        % ZSreq value %
        ZSreq1=conj(ZireqT11);
        ZSreq2=conj(ZireqT12);

            % ZSreq value %
            fprintf('ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
            fprintf(fid_w,'ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
            fprintf('ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);
            fprintf(fid_w,'ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);

            % fid_w will be open in empcal.m
            fclose(fid_w);

            ZLreq=ZLreq1;  % print the first one in the intersection %
            ZSreq=ZSreq1;  % print the first one in the intersection %

          else

        % open fid_w for append
        fid_w = fopen('out_int.txt','at+'); % write output to a file %

        fprintf('No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');
        fprintf(fid_w,'No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');

            % fid_w will be open in empcal.m
            fclose(fid_w);

        ZSreq= NaN+i*NaN;
        ZLreq= NaN+i*NaN;

      end;

%         fprintf(fid_w_g,'%f       %f       %f       %f       %f       %f
          %f\n',f,GTMAX,GTMIN,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50); %
          fprintf(fid_w_g,'%f       %f       %f       %f       %f       %f       %f       %f       %f       %f
          %f       %f
          %f\n',Vireq,FdB,f,GTMAX,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50,GTDBreq,real(ZLreq)
*50,imag(ZLreq)*50,real(ZSreq)*50,imag(ZSreq)*50);

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

    elseif (abs(Zct1-ZCGmax) > (rs+rt1)) & (abs(rs-rt1) > abs(Zct1-ZCGmin))

          fprintf('(abs(ZCGmax-Zct1) > (rs+rt1)) & ((rs+rt1) > abs(ZCGmin-Zct1)): Case b) GTmax is internal
tangential to T1\n');
```

```
        fprintf(fid_w,'(abs(ZCGmax-Zct1)          >     (rs+rt1)) & ((rs+rt1) > abs(ZCGmin-Zct1)): Case b)
GTmax is internal tangential to T1\n');

    % Case b) GTmax is internal tangential to T1 %
    % GTMAX value; Case b) GTmax is internal tangential to T1 %

    if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0,Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
      GTMC=2*C*Q/(abs(Z12)^2);
    else
        D=(R22*abs(Z11)^2-R*R11-X*X11)/R22+abs(Zopt)^2;
        E=(D*R22-Rct1*(2*R11*R22-R)-2*Xct1*XCG*R22)/rt1;
        F=Rct1/rt1;

        GTpayn=C*(Q+E*F-sqrt(((Q+E*F)^2-(1-F^2)*(abs(Z)^2-E^2))));
        GTpayda=(abs(Z12)^2*(1-F^2));
        GTMCn=GTpayn/GTpayda;
        GTpayp=C*(Q+E*F+sqrt(((Q+E*F)^2-(1-F^2)*(abs(Z)^2-E^2))));
        GTpayda=(abs(Z12)^2*(1-F^2));
        GTMCp=GTpayp/GTpayda;

      if GTMCn > GTMCp
        GTMC=GTMCn;
        GTmC=GTMCp;
      else
        GTMC=GTMCp;
        GTmC=GTMCn;
      end;

      % Checking negative sign for GTmC %
      if GTmC < 0
        GT_Negative=1;
        GTmC=abs(GTmC);
      end
    end

    S=COEFS*GTMC;

    RCGmax=-(RCS+S);
    XCGmax=-XCS;
        ZCGmax = RCGmax + i*XCGmax;
    rgmax = sqrt(S^2+ 2*S*RCS + rs^2);

    % GTmax Circle %
        ZCGTM= ZCGmax+rgmax*exp(i*teta);
        RCGTM=real(ZCGTM);
        XCGTM=imag(ZCGTM);

    % Load and Source impedance calculation; Case b) GTmax is internal tangential to T1 %
        % Zi value %
    if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
      Zi=conj(Zopt);
```

```
    else
        Zi=(((rgmax/(rt1+rgmax))*Zct1)+((rt1/(rt1+rgmax))*ZCGmax));
    end

        fprintf('Zi Value: real and imaginer part =  %f %f\n',real(Zi)*50,imag(Zi)*50);
        fprintf(fid_w,'Zi Value: real and imaginer part =  %f %f\n',real(Zi)*50,imag(Zi)*50);

        % ZL value %
        ZL=((Z12*Z21)/(Z11-Zi)-Z22);

        fprintf('ZL Value: real and imaginer part =  %f %f\n',real(ZL)*50,imag(ZL)*50);
        fprintf(fid_w,'ZL Value: real and imaginer part =  %f %f\n',real(ZL)*50,imag(ZL)*50);

        % ZS values %
        Rcv=(1+Rmod^2)*real(Zi)/C;
        Xcv=-imag(Zi);
        Zcv=Rcv+i*Xcv;
        rv=2*Rmod*real(Zi)/C;

    if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0,Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
      ZS=Zopt;
    else
        ZS=(((rn/(rn+rv))*Zcv)+((rv/(rv+rn))*Zcn));
      end

        fprintf('ZS Value: real and imaginer part =  %f %f\n',real(ZS)*50,imag(ZS)*50);
        fprintf(fid_w,'ZS Value: real and imaginer part =  %f %f\n',real(ZS)*50,imag(ZS)*50);

    %  MSG value; Case b) GTmax is internal tangential to T1 %
        MSGndB=2*(abs(Z21))/(abs(Z12))*ita;
        MSG=10*log10(MSGndB);  % MSG in dB %

        fprintf('MSG Value (dB) = %f\n',MSG);
        %fprintf(fid_w,'MSG Value (dB) = %f\n',MSG);


    %  GTMAX value; Case b) GTmax is internal tangential to T1 %
    GTMAX=10*log10(GTMC);
    if GT_Negative == 1   % Negative GT %
      GTMIN=-10*log10(GTmC);
    else
      GTMIN=10*log10(GTmC);
    end

%     if GTMAX > MSG     % MSG is maximum stable gain %
%       GTMAX=MSG;
%     end;

    fprintf('GTMAX Value (dB) = %f\n',GTMAX);
        fprintf(fid_w,'GTMAX Value (dB) = %f\n',GTMAX);
        fprintf('GTMIN Value (dB) = %f\n',GTMIN);
```

```
        fprintf(fid_w,'GTMIN    Value    (dB)    =    %f\n',GTMIN);

    % Define values to plot f-GTmax %
    x_f(counts)=f;
    y_gtmax(counts)=GTMAX;
    y_gtreq(counts)=GTDBreq;
    z_gtmin(counts)=GTMIN;

        % fid_w will be open in empcal.m
        fclose(fid_w);

        % Caluclate empedances on T1 and T2 intersection points and other points in Region-3
        if (InstersectionExist ~= 0) & (Vireq ~= 1) & (Freq ~= Fmin)   % go to claculation in Region-3


[ZLreq,ZSreq]=empcal(ZireqT11,ZireqT12,ZireqT21,ZireqT22,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn
,RCGreq,XCGreq,rgreq,Rct2,Xct2,rt2,Rct1,Xct1,rt1,Ri1,Xi1,Ri2,Xi2,step_for_sample_on_circle,switch_for_plo
t)

    elseif (InstersectionExist ~= 0) & (Vireq == 1) & (Freq == Fmin)  % T1,T2,Vi.F  are a point

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

          % Zi value %
                fprintf('ZireqT11=ZireqT12=ZireqT21=ZireqT22=  Value:  real  and  imaginer  part  =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);
              fprintf(fid_w,'ZireqT11=ZireqT12=ZireqT21=ZireqT22=  Value:  real  and  imaginer  part  =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);

      % ZLreq value %
      ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);

          % ZLreq value %
          fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
          fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);

      % ZSreq value %
      ZSreq=Zopt;

          % ZSreq value %
          fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
          fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

          % fid_w will be open in empcal.m
          fclose(fid_w);

          ZLreq=ZLreq1;  % print the first one in the intersection %


    elseif (InstersectionExist ~= 0) & (Freq == Fmin)   % T1 and T1 identical, Zs=Zopt

      % open fid_w for append
```

```
fid_w = fopen('out_int.txt','at+'); % write output to a file %

    % Zi value %
    fprintf('ZireqT11=ZireqT21 Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
    fprintf(fid_w,'ZireqT11=ZireqT21 Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

% ZLreq value %
ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

    % ZLreq value %
    fprintf('ZLreq1 Value: real and imaginer part = %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf(fid_w,'ZLreq1 Value: real and imaginer part = %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf('ZLreq2 Value: real and imaginer part = %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
    fprintf(fid_w,'ZLreq2 Value: real and imaginer part = %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

% ZSreq value %
ZSreq=Zopt;

    % ZSreq value %
    fprintf('ZSreq Value: real and imaginer part = %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
    fprintf(fid_w,'ZSreq Value: real and imaginer part = %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

    % fid_w will be open in empcal.m
    fclose(fid_w);

    ZLreq=ZLreq1; % print the first one in the intersection %

elseif (InstersectionExist ~= 0) & (Vireq == 1) % T1 and T1 identical, Zs=conj(Zi)

% open fid_w for append
fid_w = fopen('out_int.txt','at+'); % write output to a file %

    % Zi value %
    fprintf('ZireqT11=ZireqT21 Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
    fprintf(fid_w,'ZireqT11=ZireqT21 Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

% ZLreq value %
ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

    % ZLreq value %
    fprintf('ZLreq1 Value: real and imaginer part = %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf(fid_w,'ZLreq1 Value: real and imaginer part = %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf('ZLreq2 Value: real and imaginer part = %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
    fprintf(fid_w,'ZLreq2 Value: real and imaginer part = %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
```

```
        % ZSreq value %
        ZSreq1=conj(ZireqT11);
        ZSreq2=conj(ZireqT12);

            % ZSreq value %
            fprintf('ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
            fprintf(fid_w,'ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
            fprintf('ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);
            fprintf(fid_w,'ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);

            % fid_w will be open in empcal.m
            fclose(fid_w);

            ZLreq=ZLreq1;  % print the first one in the intersection %
            ZSreq=ZSreq1;  % print the first one in the intersection %

          else

        % open fid_w for append
        fid_w = fopen('out_int.txt','at+'); % write output to a file %

        fprintf('No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');
        fprintf(fid_w,'No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');

            % fid_w will be open in empcal.m
            fclose(fid_w);

        ZSreq= NaN+i*NaN;
        ZLreq= NaN+i*NaN;

      end;

%       fprintf(fid_w_g,'%f      %f      %f      %f      %f      %f
        %f\n',f,GTMAX,GTMIN,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50); %
        fprintf(fid_w_g,'%f      %f      %f      %f      %f      %f      %f      %f      %f      %f
        %f      %f
        %f\n',Vireq,FdB,f,GTMAX,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50,GTDBreq,real(ZLreq)
*50,imag(ZLreq)*50,real(ZSreq)*50,imag(ZSreq)*50);

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

    elseif (abs(ZCGmax-Zct2) > (rs+rt2)) & ((rs+rt1) > abs(ZCGmax-Zct1))

      fprintf('(abs(ZCGmax-Zct2) > (rs+rt2)) & ((rs+rt1) > abs(ZCGmax-Zct1)): Case c) T1 cuts CSSC\n');
      fprintf(fid_w,'(abs(ZCGmax-Zct2) > (rs+rt2)) & ((rs+rt1) > abs(ZCGmax-Zct1)): Case c) T1 cuts CSSC\n');

      %  Case c) T1 cuts CSSC %
      %  GTMAX value; Case c) T1 cuts CSSC %
      GTMC = 2*C*Q/(abs(Z12)^2);

      Zm=abs(ZCSC)^2-rs^2;
```

```
% Zi in intersection of CSSC-T1 points %
A=(RCS-Rct1)/(Xct1+XCS);
B=(abs(Zopt)^2-Zm)/(2*(Xct1+XCS));
K=(Rct1-A*B+A*Xct1)/(1+A^2);
D=(abs(Zopt)^2+B^2-2*B*Xct1)/(1+A^2);
Ri_1=K+sqrt(K^2-D);
Ri_2=K-sqrt(K^2-D);
Xi_1=A*Ri_1+B;
    Xi_2=A*Ri_2+B;

% Load and Source impedance calculation; Case c) T1 cuts CSSC %
    % Zi values %
Zi_1=Ri_1+i*Xi_1;
Zi_2=Ri_2+i*Xi_2;

    Zi=Zi_1;

    fprintf('Zi_1 Value: real and imaginer part =  %f %f\n',real(Zi_1)*50,imag(Zi_1)*50);
    fprintf(fid_w,'Zi_1 Value: real and imaginer part =  %f %f\n',real(Zi_1)*50,imag(Zi_1)*50);
    fprintf('Zi_2 Value: real and imaginer part =  %f %f\n',real(Zi_2)*50,imag(Zi_2)*50);
    fprintf(fid_w,'Zi_2 Value: real and imaginer part =  %f %f\n',real(Zi_2)*50,imag(Zi_2)*50);

    % ZL values %
    ZL_1=((Z12*Z21)/(Z11-Zi_1)-Z22);
    ZL_2=((Z12*Z21)/(Z11-Zi_2)-Z22);

    ZL=ZL_1;  % take just one %

    fprintf('ZL_1 Value: real and imaginer part =  %f %f\n',real(ZL_1)*50,imag(ZL_1)*50);
    fprintf(fid_w,'ZL_1 Value: real and imaginer part =  %f %f\n',real(ZL_1)*50,imag(ZL_1)*50);
    fprintf('ZL_2 Value: real and imaginer part =  %f %f\n',real(ZL_2)*50,imag(ZL_2)*50);
    fprintf(fid_w,'ZL_2 Value: real and imaginer part =  %f %f\n',real(ZL_2)*50,imag(ZL_2)*50);

    % ZS values %
    Rcv=(1+Rmod^2)*real(Zi_1)/C;
    Xcv=-imag(Zi_1);
    Zcv=Rcv+i*Xcv;
    rv=2*Rmod*real(Zi_1)/C;

  if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
    ZS_1=Zopt;
  else
      ZS_1=(((rn/(rn+rv))*Zcv)+((rv/(rv+rn))*Zcn));
    end

    Rcv=(1+Rmod^2)*real(Zi_2)/C;
    Xcv=-imag(Zi_2);
    Zcv=Rcv+i*Xcv;
    rv=2*Rmod*real(Zi_2)/C;

  if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
```

```
  ZS_2=Zopt;
else
     ZS_2=(((rn/(rn+rv))*Zcv)+((rv/(rv+rn))*Zcn));
   end

   ZS=ZS_1;  % take just one %

   fprintf('ZS_1 Value: real and imaginer part =  %f %f\n',real(ZS_1)*50,imag(ZS_1)*50);
   fprintf(fid_w,'ZS_1 Value: real and imaginer part =  %f %f\n',real(ZS_1)*50,imag(ZS_1)*50);
   fprintf('ZS_2 Value: real and imaginer part =  %f %f\n',real(ZS_2)*50,imag(ZS_2)*50);
   fprintf(fid_w,'ZS_2 Value: real and imaginer part =  %f %f\n',real(ZS_2)*50,imag(ZS_2)*50);

% MSG value; Case c) T1 cuts CSSC (Rmod=0--> C=1) %
   MSGndB=2*(abs(Z21))/(abs(Z12))*ita;
   MSG=10*log10(MSGndB);  % MSG in dB %

   fprintf('MSG Value (dB) = %f\n',MSG);
   %fprintf(fid_w,'MSG Value (dB) = %f\n',MSG);

% GTMAX value; Case c) T1 cuts CSSC %
GTMAX=10*log10(GTMC);
GTMIN=0; % in dB %

%    if GTMAX > MSG     % MSG is maximum stable gain %
%      GTMAX=MSG;
%    end;

fprintf('GTMAX Value (dB) = %f\n',GTMAX);
   fprintf(fid_w,'GTMAX Value (dB) = %f\n',GTMAX);
   fprintf('GTMIN Value (dB) = %f\n',GTMIN);
   fprintf(fid_w,'GTMIN Value (dB) = %f\n',GTMIN);

% Define values to plot f-GTmax %
x_f(counts)=f;
y_gtmax(counts)=GTMAX;
y_gtreq(counts)=GTDBreq;
z_gtmin(counts)=GTMIN;

   % fid_w will be open in empcal.m
   fclose(fid_w);

   % Caluclate empedances on T1 and T2 intersection points and other points in Region-3
   if (InstersectionExist ~= 0) & (Vireq ~= 1) & (Freq ~= Fmin)   % go to claculation in Region-3


[ZLreq,ZSreq]=empcal(ZireqT11,ZireqT12,ZireqT21,ZireqT22,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn
,RCGreq,XCGreq,rgreq,Rct2,Xct2,rt2,Rct1,Xct1,rt1,Ri1,Xi1,Ri2,Xi2,step_for_sample_on_circle,switch_for_plo
t)

elseif (InstersectionExist ~= 0) & (Vireq == 1) & (Freq == Fmin)  % T1,T2,Vi.F  are a point

  % open fid_w for append
  fid_w = fopen('out_int.txt','at+'); % write output to a file %
```

```
        % Zi value %
            fprintf('ZireqT11=ZireqT12=ZireqT21=ZireqT22=  Value: real and imaginer part =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);
            fprintf(fid_w,'ZireqT11=ZireqT12=ZireqT21=ZireqT22=  Value: real and imaginer part =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);

    % ZLreq value %
    ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);

        % ZLreq value %
        fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);

    % ZSreq value %
    ZSreq=Zopt;

        % ZSreq value %
        fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
        fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

        % fid_w will be open in empcal.m
        fclose(fid_w);

        ZLreq=ZLreq1;  % print the first one in the intersection %


elseif (InstersectionExist ~= 0) & (Freq == Fmin)   % T1 and T1 identical, Zs=Zopt

    % open fid_w for append
    fid_w = fopen('out_int.txt','at+'); % write output to a file %

        % Zi value %
        fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real
and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
        fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22
Value: real and imaginer part = %f
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

    % ZLreq value %
    ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
    ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

        % ZLreq value %
        fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
        fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

    % ZSreq value %
    ZSreq=Zopt;

        % ZSreq value %
```

```
        fprintf('ZSreq Value: real and imaginer part      =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
        fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

        % fid_w will be open in empcal.m
        fclose(fid_w);

        ZLreq=ZLreq1;  % print the first one in the intersection %

    elseif (InstersectionExist ~= 0) & (Vireq == 1)   % T1 and T1 identical, Zs=conj(Zi)

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

        % Zi value %
        fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
        fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

      % ZLreq value %
      ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
      ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

        % ZLreq value %
        fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
        fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

      % ZSreq value %
      ZSreq1=conj(ZireqT11);
      ZSreq2=conj(ZireqT12);

        % ZSreq value %
        fprintf('ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
        fprintf(fid_w,'ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
        fprintf('ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);
        fprintf(fid_w,'ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);

        % fid_w will be open in empcal.m
        fclose(fid_w);

        ZLreq=ZLreq1;  % print the first one in the intersection %
        ZSreq=ZSreq1;  % print the first one in the intersection %

      else

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

      fprintf('No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');
      fprintf(fid_w,'No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');
```

```
        % fid_w will be open in empcal.m
        fclose(fid_w);

      ZSreq= NaN+i*NaN;
      ZLreq= NaN+i*NaN;

    end;

%       fprintf(fid_w_g,'%f      %f      %f      %f      %f      %f
        %f\n',f,GTMAX,GTMIN,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50); %
        fprintf(fid_w_g,'%f      %f      %f      %f      %f      %f      %f      %f      %f
        %f      %f
        %f\n',Vireq,FdB,f,GTMAX,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50,GTDBreq,real(ZLreq)
*50,imag(ZLreq)*50,real(ZSreq)*50,imag(ZSreq)*50);

    % open fid_w for append
    fid_w = fopen('out_int.txt','at+'); % write output to a file %

  elseif abs(ZCGmax-Zct2) < (rs+rt2)

        fprintf('abs(ZCGmax-Zct2) < (rs+rt2): Case d) T2 cuts CSSC\n');
        fprintf(fid_w,'abs(ZCGmax-Zct2) < (rs+rt2): Case d) T2 cuts CSSC\n');

    % Case d) T2 cuts CSSC %

    % GTMAX value; Case d) T2 cuts CSSC %
    GTMC = 2*C*Q/(abs(Z12)^2);

    Zm=abs(ZCSC)^2-rs^2;

    % Zi in intersection of CSSC-T2 points %
    A=(RCS-Rct2)/(Xct2+XCS);
    B=(abs(Zopt)^2-Zm)/(2*(Xct2+XCS));
    K=(Rct2-A*B+A*Xct2)/(1+A^2);
    D=(abs(Zopt)^2+B^2-2*B*Xct2)/(1+A^2);
    Ri_1=K+sqrt(K^2-D);
    Ri_2=K-sqrt(K^2-D);

        fprintf('Ri_1,Ri_2: %f %f\n',Ri_1,Ri_2);

    Xi_1=A*Ri_1+B;
        Xi_2=A*Ri_2+B;

    % Load and Source impedance calculation; Case d) T2 cuts CSSC %
        % Zi values %
    Zi_1=Ri_1+i*Xi_1;
    Zi_2=Ri_2+i*Xi_2;

        Zi=Zi_1;

        fprintf('Zi_1 Value: real and imaginer part =  %f %f\n',real(Zi_1)*50,imag(Zi_1)*50);
        fprintf(fid_w,'Zi_1 Value: real and imaginer part =  %f %f\n',real(Zi_1)*50,imag(Zi_1)*50);
```

```
        fprintf('Zi_2 Value: real and imaginer part =      %f %f\n',real(Zi_2)*50,imag(Zi_2)*50);
        fprintf(fid_w,'Zi_2 Value: real and imaginer part =  %f %f\n',real(Zi_2)*50,imag(Zi_2)*50);

        % ZL values %
        ZL_1=((Z12*Z21)/(Z11-Zi_1)-Z22);
        ZL_2=((Z12*Z21)/(Z11-Zi_2)-Z22);

        ZL=ZL_2;  % take just one %

        fprintf('ZL_1 Value: real and imaginer part =  %f %f\n',real(ZL_1)*50,imag(ZL_1)*50);
        fprintf(fid_w,'ZL_1 Value: real and imaginer part =  %f %f\n',real(ZL_1)*50,imag(ZL_1)*50);
        fprintf('ZL_2 Value: real and imaginer part =  %f %f\n',real(ZL_2)*50,imag(ZL_2)*50);
        fprintf(fid_w,'ZL_2 Value: real and imaginer part =  %f %f\n',real(ZL_2)*50,imag(ZL_2)*50);

        % ZS values %
        Rcv=(1+Rmod^2)*real(Zi_1)/C;
        Xcv=-imag(Zi_1);
        Zcv=Rcv+i*Xcv;
        rv=2*Rmod*real(Zi_1)/C;
    if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
      ZS_1=Zopt;
    else
          ZS_1=(((rn/(rn+rv))*Zcv)+((rv/(rv+rn))*Zcn));
        end

        Rcv=(1+Rmod^2)*real(Zi_2)/C;
        Xcv=-imag(Zi_2);
        Zcv=Rcv+i*Xcv;
        rv=2*Rmod*real(Zi_2)/C;
    if (Vireq == 1) & (Freq == Fmin)   % VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn,Rmod=0
Zi=conj(Zopt), Zcv=conj(Zi)
      ZS_2=Zopt;
    else
          ZS_2=(((rn/(rn+rv))*Zcv)+((rv/(rv+rn))*Zcn));
        end

    ZS=ZS_2;  % take just one %

        fprintf('ZS_1 Value: real and imaginer part =  %f %f\n',real(ZS_1)*50,imag(ZS_1)*50);
        fprintf(fid_w,'ZS_1 Value: real and imaginer part =  %f %f\n',real(ZS_1)*50,imag(ZS_1)*50);
        fprintf('ZS_2 Value: real and imaginer part =  %f %f\n',real(ZS_2)*50,imag(ZS_2)*50);
        fprintf(fid_w,'ZS_2 Value: real and imaginer part =  %f %f\n',real(ZS_2)*50,imag(ZS_2)*50);

    %  MSG value; Case d) T2 cuts CSSC (Rmod=0--> C=1) %
        MSGndB=2*(abs(Z21))/(abs(Z12))*ita;
        MSG=10*log10(MSGndB);  % MSG in dB %

        fprintf('MSG Value (dB) = %f\n',MSG);
        %fprintf(fid_w,'MSG Value (dB) = %f\n',MSG);

    %  GTMAX value; Case d) T2 cuts CSSC %
    GTMAX=10*log10(GTMC);
```

```
GTMIN=0; % in dB %

%     if GTMAX > MSG     % MSG is maximum stable gain %
%       GTMAX=MSG;
%     end;

fprintf('GTMAX Value (dB) = %f\n',GTMAX);
    fprintf(fid_w,'GTMAX Value (dB) = %f\n',GTMAX);
    fprintf('GTMIN Value (dB) = %f\n',GTMIN);
    fprintf(fid_w,'GTMIN Value (dB) = %f\n',GTMIN);

% Define values to plot f-GTmax %
x_f(counts)=f;
y_gtmax(counts)=GTMAX;
y_gtreq(counts)=GTDBreq;
z_gtmin(counts)=GTMIN;

    % fid_w will be open in empcal.m
    fclose(fid_w);

    % Caluclate empedances on T1 and T2 intersection points and other points in Region-3
    if (InstersectionExist ~= 0) & (Vireq ~= 1) & (Freq ~= Fmin)   % go to claculation in Region-3


[ZLreq,ZSreq]=empcal(ZireqT11,ZireqT12,ZireqT21,ZireqT22,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn
,RCGreq,XCGreq,rgreq,Rct2,Xct2,rt2,Rct1,Xct1,rt1,Ri1,Xi1,Ri2,Xi2,step_for_sample_on_circle,switch_for_plo
t)

elseif (InstersectionExist ~= 0) & (Vireq == 1) & (Freq == Fmin)  % T1,T2,Vi.F  are a point

  % open fid_w for append
  fid_w = fopen('out_int.txt','at+'); % write output to a file %

    % Zi value %
        fprintf('ZireqT11=ZireqT12=ZireqT21=ZireqT22=  Value: real and imaginer part =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);
       fprintf(fid_w,'ZireqT11=ZireqT12=ZireqT21=ZireqT22=  Value: real and imaginer part =
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50);

  % ZLreq value %
  ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);

    % ZLreq value %
    fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);

  % ZSreq value %
  ZSreq=Zopt;

    % ZSreq value %
    fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
    fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
```

```
        % fid_w will be open in empcal.m
        fclose(fid_w);

        ZLreq=ZLreq1;  % print the first one in the intersection %


    elseif (InstersectionExist ~= 0) & (Freq == Fmin)   % T1 and T1 identical, Zs=Zopt

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

          % Zi value %
          fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real
and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
          fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22
Value:  real  and  imaginer  part  =  %f
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

      % ZLreq value %
      ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
      ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

          % ZLreq value %
          fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
          fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
          fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
          fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

      % ZSreq value %
      ZSreq=Zopt;

          % ZSreq value %
          fprintf('ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);
          fprintf(fid_w,'ZSreq Value: real and imaginer part =  %f %f\n',real(ZSreq)*50,imag(ZSreq)*50);

          % fid_w will be open in empcal.m
          fclose(fid_w);

          ZLreq=ZLreq1;  % print the first one in the intersection %

    elseif (InstersectionExist ~= 0) & (Vireq == 1)   % T1 and T1 identical, Zs=conj(Zi)

      % open fid_w for append
      fid_w = fopen('out_int.txt','at+'); % write output to a file %

          % Zi value %
          fprintf('ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22 Value: real
and imaginer part = %f %f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);
          fprintf(fid_w,'ZireqT11=ZireqT21  Value: real and imaginer part = %f %f, ZireqT12=ZireqT22
Value:  real  and  imaginer  part  =  %f
%f\n',real(ZireqT11)*50,imag(ZireqT11)*50,real(ZireqT12)*50,imag(ZireqT12)*50);

      % ZLreq value %
```

```
ZLreq1=((Z12*Z21)/(Z11-ZireqT11)-Z22);
ZLreq2=((Z12*Z21)/(Z11-ZireqT12)-Z22);

    % ZLreq value %
    fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf(fid_w,'ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
    fprintf(fid_w,'ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

% ZSreq value %
ZSreq1=conj(ZireqT11);
ZSreq2=conj(ZireqT12);

    % ZSreq value %
    fprintf('ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
    fprintf(fid_w,'ZSreq1 Value: real and imaginer part =  %f %f\n',real(ZSreq1)*50,imag(ZSreq1)*50);
    fprintf('ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);
    fprintf(fid_w,'ZSreq2 Value: real and imaginer part =  %f %f\n',real(ZSreq2)*50,imag(ZSreq2)*50);

    % fid_w will be open in empcal.m
    fclose(fid_w);

    ZLreq=ZLreq1;  % print the first one in the intersection %
    ZSreq=ZSreq1;  % print the first one in the intersection %

  else

% open fid_w for append
fid_w = fopen('out_int.txt','at+'); % write output to a file %

fprintf('No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');
fprintf(fid_w,'No solution in Region-3: (InstersectionExist = 0) | (Vireq = 1) | (Freq = Fmin)\n');

    % fid_w will be open in empcal.m
    fclose(fid_w);

ZSreq= NaN+i*NaN;
ZLreq= NaN+i*NaN;

end;

%       fprintf(fid_w_g,'%f      %f      %f      %f      %f      %f
        %f\n',f,GTMAX,GTMIN,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50); %
        fprintf(fid_w_g,'%f      %f      %f      %f      %f      %f      %f      %f      %f      %f
        %f      %f
        %f\n',Vireq,FdB,f,GTMAX,real(ZL)*50,imag(ZL)*50,real(ZS)*50,imag(ZS)*50,GTDBreq,real(ZLreq)
*50,imag(ZLreq)*50,real(ZSreq)*50,imag(ZSreq)*50);

% open fid_w for append
fid_w = fopen('out_int.txt','at+'); % write output to a file %

else
```

```
    fprintf('Not defined\n');
    fprintf(fid_w,'Not defined\n');


  end; % Conserning Solution %

  if ZCGTM == 0

    % Drawing Circles %  % Ri41 --> GTmin, so not drawn %

    if switch_for_plot == 1

      figure,plot(RCGTreq,XCGTreq,'k-',Ri1,Xi1,'m-',Ri2,Xi2,'m-.',Ri5,Xi5,'g-',Ri3,Xi3,'c-
',Ri42,Xi42,'r:',Ri43,Xi43,'r:',Ri44,Xi44,'r:',Ri45,Xi45,'r:')
            axis([-1 1 -1 1])
            title('Zi-Plane')
            xlabel('real{Zi}'),ylabel('imag{Zi}'),grid
            gtext('GTreq'),gtext('T1'),gtext('T2'),gtext('ISC=GTmin'),gtext('CSSC=GTmax'),gtext('Gain
Circles'),gtext('Conditional Stability')
    end

  else  % to draw GTmax for tangential case b) %

    % Drawing Circles %  % Ri41 --> GTmin, so not drawn %

    if switch_for_plot == 1

      figure,plot(RCGTreq,XCGTreq,'k-',Ri1,Xi1,'m-',Ri2,Xi2,'m-.',RCGTM,XCGTM,'b-',Ri5,Xi5,'g-
',Ri3,Xi3,'c-',Ri42,Xi42,'r:',Ri43,Xi43,'r:',Ri44,Xi44,'r:',Ri45,Xi45,'r:')
            axis([-1 1 -1 1])
            title('Zi-Plane')
            xlabel('real{Zi}'),ylabel('imag{Zi}'),grid
            gtext('GTreq'),gtext('T1'),gtext('T2'),gtext('GTmax'),gtext('ISC=GTmin'),gtext('CSSC'),gtext('Gain
Circles'),gtext('Conditional Stability')
    end

  end;


 else  % Stability: else %

  fprintf('Not Conditional or unconditional stable\n');
  fprintf(fid_w,'Not Conditional or unconditional stable\n');
  fprintf('R11,R22,ita = %f,%f,%f \n ',R11,R22,ita);
  fprintf(fid_w,'R11,R22,ita = %f,%f,%f\n',R11,R22,ita);

 end;  % Stability: end %

end;  % loop under main loop %

%  Plots f-GTmax %

% fprintf('x_f= %f, y_gtmax= %f, y_gtreq=%f\n ',x_f,y_gtmax,y_gtreq);
```

```
if switch_for_plot == 1

  figure,plot(x_f,y_gtmax,'g-',x_f,y_gtreq,'c-')
      %axis([0 25 0 18])%
      title('Frequency-GTmax')
      xlabel('f(Hz)'),ylabel('GTmax(dB)'),grid
      gtext('GTmax'),gtext('GTreq');

  figure,plot(x_f,z_gtmin,'g-')
      %axis([0 25 0 18])%
      title('Frequency-GTmin')
      xlabel('f(Hz)'),ylabel('GTmin(dB)'),grid

end

end;  % main loop %

%  Close files %
fclose(fid_r_s);
fclose(fid_r_n);
fclose(fid_r_p);
fclose(fid_w);
fclose(fid_w_g);


% End of drawing Circles */


function
[ZLreq,ZSreq]=empcal(ZireqT11,ZireqT12,ZireqT21,ZireqT22,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn
,RCGreq,XCGreq,rgreq,Rct2,Xct2,rt2,Rct1,Xct1,rt1,Ri1,Xi1,Ri2,Xi2,step_for_sample_on_circle,switch_for_plo
t)
%/*************************************************************/
%/*                                                           */
%/*  File:          empcal.m                                  */
%/*                                                           */
%/*  Compiler:    Matlab v5.0                                 */
%/*                                                           */
%/*  Description:Caluclate empedances on T1 and T2            */
%/*            intersection points and other points           */
%/*            in Region-3                                    */
%/*                                                           */
%/*  Created:      Cemal Tepe 1999  Ph.D. Thesis              */
%/*                                                           */
%/*  Version:      1.0                                        */
%/*                                                           */
%/*  Notes:                                                   */
%/*  Zireq1 --> ZLreq1,Zs11,Zs12,ZsViF11,ZsViF12              */
%/*  Zireq2 --> ZLreq2,Zs21,Zs22,ZsViF21,ZsViF22              */
%/*                                                           */
%/*  ZLreq and ZSreq is one of the selection in calculated    */
%/*       empedances                                          */
```

```
%/*                                                                    */
%/*********************************************************************/
%
%
%

fid_w_int = fopen('out_int.txt','at+'); % append intersections to a file %
fid_w_oth = fopen('out_oth.txt','at+'); % append other points to a file %

% Caluclate empedances on T1 and T2 intersection points and other points in Region-3

if ~isnan(ZireqT11) & ~isnan(ZireqT12) &  ~isnan(ZireqT21) & ~isnan(ZireqT22)

    % T1 Intersection
        Zireq1=ZireqT11;
        Zireq2=ZireqT12;

        fprintf('T1 Intersection:\n');
        fprintf(fid_w_int,'T1 Intersection:\n');

        fprintf('Zireq1 Value: real and imaginer part =  %f %f\n',real(Zireq1)*50,imag(Zireq1)*50);
        fprintf(fid_w_int,'Zireq1 Value: real and imaginer part =  %f %f\n',real(Zireq1)*50,imag(Zireq1)*50);
        fprintf('Zireq2 Value: real and imaginer part =  %f %f\n',real(Zireq2)*50,imag(Zireq2)*50);
        fprintf(fid_w_int,'Zireq2 Value: real and imaginer part =  %f %f\n',real(Zireq2)*50,imag(Zireq2)*50);


[ZLreq1,ZLreq2,Zs11,Zs12,ZsViF11,ZsViF12,Zs21,Zs22,ZsViF21,ZsViF22,Rcv1,Xcv1,rv1,Rcv2,Xcv2,rv2]=e
mpcali(Zireq1,Zireq2,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn,switch_for_plot)

        % ZLreq value %
        fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf(fid_w_int,'ZLreq1 Value: real and imaginer part =  %f
%f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
        fprintf(fid_w_int,'ZLreq2 Value: real and imaginer part =  %f
%f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

        % ZSreq value by formulas
        fprintf('ZSreq11 Value: real and imaginer part =  %f %f\n',real(Zs11)*50,imag(Zs11)*50);
        fprintf(fid_w_int,'ZSreq11 Value: real and imaginer part =  %f %f\n',real(Zs11)*50,imag(Zs11)*50);
        fprintf('ZSreq12 Value: real and imaginer part =  %f %f\n',real(Zs12)*50,imag(Zs12)*50);
        fprintf(fid_w_int,'ZSreq12 Value: real and imaginer part =  %f %f\n',real(Zs12)*50,imag(Zs12)*50);

        fprintf('ZSreq21 Value: real and imaginer part =  %f %f\n',real(Zs21)*50,imag(Zs21)*50);
        fprintf(fid_w_int,'ZSreq21 Value: real and imaginer part =  %f %f\n',real(Zs21)*50,imag(Zs21)*50);
        fprintf('ZSreq22 Value: real and imaginer part =  %f %f\n',real(Zs22)*50,imag(Zs22)*50);
        fprintf(fid_w_int,'ZSreq22 Value: real and imaginer part =  %f %f\n',real(Zs22)*50,imag(Zs22)*50);

%       % ZSreq value by circle Intersection of Vi-F
%       if ~isnan(ZsViF11) & ~isnan(ZsViF12) & ~isnan(ZsViF21) & ~isnan(ZsViF22)
%
%                   fprintf('Vi-F:  Rcv1=  %f,  Xcv1=  %f,  rv1=  %f,  Rcn=  %f,  Xcn=  %f,  rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
```

```
%                   fprintf(fid_w_int,'Vi-F: Rcv1=    %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%             fprintf('1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf(fid_w_int,'1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%             fprintf('2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%           fprintf(fid_w_int,'2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                   fprintf(fid_w_int,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%             fprintf('1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf(fid_w_int,'1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%             fprintf('2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%           fprintf(fid_w_int,'2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%
%       elseif ~isnan(ZsViF11) & ~isnan(ZsViF12)
%
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                   fprintf(fid_w_int,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%             fprintf('1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf(fid_w_int,'1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%             fprintf('2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%           fprintf(fid_w_int,'2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%
%       elseif ~isnan(ZsViF21) & ~isnan(ZsViF22)
%
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                   fprintf(fid_w_int,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%             fprintf('1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf(fid_w_int,'1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%             fprintf('2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%           fprintf(fid_w_int,'2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%
%       else
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
```

```
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                   fprintf(fid_w_int,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                     fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                   fprintf(fid_w_int,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                 fprintf('No intersection on Vi-F; ZSreq11,ZSreq12,ZSreq21,ZSreq22= NaN\n');
%          fprintf(fid_w_int,'No intersection on Vi-F; ZSreq11,ZSreq12,ZSreq21,ZSreq22= NaN\n');
%     end


        % T2 Intersection
        Zireq1=ZireqT21;
        Zireq2=ZireqT22;

        fprintf('T2 Intersection:\n');
        fprintf(fid_w_int,'T2 Intersection:\n');

        fprintf('Zireq1 Value: real and imaginer part =  %f %f\n',real(Zireq1)*50,imag(Zireq1)*50);
        fprintf(fid_w_int,'Zireq1 Value: real and imaginer part =  %f %f\n',real(Zireq1)*50,imag(Zireq1)*50);
        fprintf('Zireq2 Value: real and imaginer part =  %f %f\n',real(Zireq2)*50,imag(Zireq2)*50);
        fprintf(fid_w_int,'Zireq2 Value: real and imaginer part =  %f %f\n',real(Zireq2)*50,imag(Zireq2)*50);

        [ZLreq1,ZLreq2,Zs11,Zs12,ZsViF11,ZsViF12,Zs21,Zs22,ZsViF21,ZsViF22,Rcv1,Xcv1,rv1,Rcv2,Xcv
2,rv2]=empcali(Zireq1,Zireq2,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn,switch_for_plot)

        % ZLreq value %
        fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf(fid_w_int,'ZLreq1 Value: real and imaginer part =  %f
%f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
        fprintf(fid_w_int,'ZLreq2 Value: real and imaginer part =  %f
%f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

        % ZSreq value by formulas
        fprintf('ZSreq11 Value: real and imaginer part =  %f %f\n',real(Zs11)*50,imag(Zs11)*50);
        fprintf(fid_w_int,'ZSreq11 Value: real and imaginer part =  %f %f\n',real(Zs11)*50,imag(Zs11)*50);
        fprintf('ZSreq12 Value: real and imaginer part =  %f %f\n',real(Zs12)*50,imag(Zs12)*50);
        fprintf(fid_w_int,'ZSreq12 Value: real and imaginer part =  %f %f\n',real(Zs12)*50,imag(Zs12)*50);

        fprintf('ZSreq21 Value: real and imaginer part =  %f %f\n',real(Zs21)*50,imag(Zs21)*50);
        fprintf(fid_w_int,'ZSreq21 Value: real and imaginer part =  %f %f\n',real(Zs21)*50,imag(Zs21)*50);
        fprintf('ZSreq22 Value: real and imaginer part =  %f %f\n',real(Zs22)*50,imag(Zs22)*50);
        fprintf(fid_w_int,'ZSreq22 Value: real and imaginer part =  %f %f\n',real(Zs22)*50,imag(Zs22)*50);


%       % ZSreq value by circle Intersection of Vi-F
%       if ~isnan(ZsViF11) & ~isnan(ZsViF12) & ~isnan(ZsViF21) & ~isnan(ZsViF22)
%
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                   fprintf(fid_w_int,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
```

```
%           fprintf('1. Intersection point on Vi-F;  ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf(fid_w_int,'1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%           fprintf(fid_w_int,'2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                  fprintf(fid_w_int,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf(fid_w_int,'1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%           fprintf(fid_w_int,'2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%
%       elseif ~isnan(ZsViF11) & ~isnan(ZsViF12)
%
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                  fprintf(fid_w_int,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf(fid_w_int,'1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%           fprintf(fid_w_int,'2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%
%       elseif ~isnan(ZsViF21) & ~isnan(ZsViF22)
%
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                  fprintf(fid_w_int,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf(fid_w_int,'1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%          fprintf(fid_w_int,'2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%
%       else
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                  fprintf(fid_w_int,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
```

```
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                       fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                     fprintf(fid_w_int,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                   fprintf('No intersection on Vi-F; ZSreq11,ZSreq12,ZSreq21,ZSreq22= NaN\n');
%          fprintf(fid_w_int,'No intersection on Vi-F; ZSreq11,ZSreq12,ZSreq21,ZSreq22= NaN\n');
%     end

        ZLreq=ZLreq1;  % print the first one in the intersection %
        ZSreq=Zs11; % print the first one in the intersection %

        % Calculation of many points of Zi in Region-3

        % GTreq cuts T1 and T2; take T1 as starting point and decrease test circle (rt1_step)  untill it cuts T2 or
it becomes tangent to GTreq
        rt1_step=rt1;

        fprintf('Calculation of many other points of Zi in Region-3; GTreq cuts T1 and T2\n');
        fprintf(fid_w_oth,'Calculation of many other points of Zi in Region-3; GTreq cuts T1 and T2\n');


        loop_count=0;
        while 1 == 1    % loop infinity

          rt1_step=rt1_step-step_for_sample_on_circle;

          [RintR,XintR]=circlint(RCGreq,XCGreq,rgreq,Rct1,Xct1,rt1_step);
          ZireqR1=RintR(1)+i*XintR(1);
          ZireqR2=RintR(2)+i*XintR(2);

          [RintTest,XintTest]=circlint(Rct2,Xct2,rt2,Rct1,Xct1,rt1_step);
          ZireqTest1=RintTest(1)+i*XintTest(1);
          ZireqTest2=RintTest(2)+i*XintTest(2);

            if (ZireqR1 == ZireqR2) | (isnan(ZireqR1) & isnan(ZireqR2)) | (~isnan(ZireqTest1) &
~isnan(ZireqTest2)) break; end   % tangent case or no intersection

          loop_count=loop_count+2;

           fprintf('Zireq%d Value: real and imaginer part =    %f %f\n',loop_count-1,real(ZireqR1)*50,
imag(ZireqR1)*50);
            fprintf(fid_w_oth,'Zireq%d Value: real and imaginer part =    %f %f\n',loop_count-
1,real(ZireqR1)*50, imag(ZireqR1)*50);
           fprintf('Zireq%d Value: real and imaginer part =    %f %f\n',loop_count,real(ZireqR2)*50,
imag(ZireqR2)*50);
          fprintf(fid_w_oth,'Zireq%d Value: real and imaginer part =   %f %f\n',loop_count,real(ZireqR2)*50,
imag(ZireqR2)*50);


          % Test Circle %
          %ZCTest=Rct1+i*Xct1
          %ZCZTest= ZCTest+rt1_step*exp(i*teta);
          %RCZTest=real(ZCZTest);
```

```
%XCZTest=imag(ZCZTest);

if switch_for_plot == 1
  %figure,plot(RCGTreq,XCGTreq,'k-',Ri1,Xi1,'m-',Ri2,Xi2,'m-.',RCZTest,XCZTest,'b-')
%axis([-1 1 -3 2])
%title('Zi-Plane (test)')
%xlabel('real{Zi}'),ylabel('imag{Zi}'),grid
%gtext('GTreq'),gtext('T1'),gtext('T2'), gtext('Test')
end
% call empcali
Zireq1=ZireqR1;
Zireq2=ZireqR2;


[ZLreq1,ZLreq2,Zs11,Zs12,ZsViF11,ZsViF12,Zs21,Zs22,ZsViF21,ZsViF22,Rcv1,Xcv1,rv1,Rcv2,Xcv2,rv2]=e
mpcali(Zireq1,Zireq2,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn,switch_for_plot)

% ZLreq value %
fprintf(' ZLreq1 Value: real and imaginer part = %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
    fprintf(fid_w_oth,' ZLreq1 Value: real and imaginer part = %f
%f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
fprintf(' ZLreq2 Value: real and imaginer part = %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
    fprintf(fid_w_oth,' ZLreq2 Value: real and imaginer part = %f
%f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

% ZSreq value by formulas
fprintf(' ZSreq11 Value: real and imaginer part = %f %f\n',real(Zs11)*50,imag(Zs11)*50);
    fprintf(fid_w_oth,' ZSreq11 Value: real and imaginer part = %f
%f\n',real(Zs11)*50,imag(Zs11)*50);
fprintf(' ZSreq12 Value: real and imaginer part = %f %f\n',real(Zs12)*50,imag(Zs12)*50);
    fprintf(fid_w_oth,' ZSreq12 Value: real and imaginer part = %f
%f\n',real(Zs12)*50,imag(Zs12)*50);

fprintf(' ZSreq21 Value: real and imaginer part = %f %f\n',real(Zs21)*50,imag(Zs21)*50);
    fprintf(fid_w_oth,' ZSreq21 Value: real and imaginer part = %f
%f\n',real(Zs21)*50,imag(Zs21)*50);
fprintf(' ZSreq22 Value: real and imaginer part = %f %f\n',real(Zs22)*50,imag(Zs22)*50);
    fprintf(fid_w_oth,' ZSreq22 Value: real and imaginer part = %f
%f\n',real(Zs22)*50,imag(Zs22)*50);

%       % ZSreq value by circle Intersection of Vi-F
%       if ~isnan(ZsViF11) & ~isnan(ZsViF12) & ~isnan(ZsViF21) & ~isnan(ZsViF22)
%
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                 fprintf(fid_w_oth,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%         fprintf(fid_w_oth,'1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%         fprintf(fid_w_oth,'2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
```

```
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                   fprintf(fid_w_oth,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf(fid_w_oth,'1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%           fprintf(fid_w_oth,'2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%
%       elseif ~isnan(ZsViF11) & ~isnan(ZsViF12)
%
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                   fprintf(fid_w_oth,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf(fid_w_oth,'1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%           fprintf(fid_w_oth,'2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%
%       elseif ~isnan(ZsViF21) & ~isnan(ZsViF22)
%
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                   fprintf(fid_w_oth,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf(fid_w_oth,'1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%           fprintf(fid_w_oth,'2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%
%       else
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                   fprintf(fid_w_oth,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                   fprintf(fid_w_oth,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                fprintf('No intersection on Vi-F; ZSreq11,ZSreq12,ZSreq21,ZSreq22= NaN\n');
%           fprintf(fid_w_oth,'No intersection on Vi-F; ZSreq11,ZSreq12,ZSreq21,ZSreq22= NaN\n');
```

```
%       end


    end


elseif ~isnan(ZireqT11) & ~isnan(ZireqT12)

    % T1 Intersection
        Zireq1=ZireqT11;
        Zireq2=ZireqT12;

        fprintf('T1 Intersection:\n');
        fprintf(fid_w_int,'T1 Intersection:\n');

        fprintf('Zireq1 Value: real and imaginer part =  %f %f\n',real(Zireq1)*50,imag(Zireq1)*50);
        fprintf(fid_w_int,'Zireq1 Value: real and imaginer part =  %f %f\n',real(Zireq1)*50,imag(Zireq1)*50);
        fprintf('Zireq2 Value: real and imaginer part =  %f %f\n',real(Zireq2)*50,imag(Zireq2)*50);
        fprintf(fid_w_int,'Zireq2 Value: real and imaginer part =  %f %f\n',real(Zireq2)*50,imag(Zireq2)*50);

        [ZLreq1,ZLreq2,Zs11,Zs12,ZsViF11,ZsViF12,Zs21,Zs22,ZsViF21,ZsViF22,Rcv1,Xcv1,rv1,Rcv2,Xcv
2,rv2]=empcali(Zireq1,Zireq2,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn,switch_for_plot)

        % ZLreq value %
        fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf(fid_w_int,'ZLreq1 Value: real and imaginer part = %f
%f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
        fprintf(fid_w_int,'ZLreq2 Value: real and imaginer part = %f
%f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

        % ZSreq value by formulas
        fprintf('ZSreq11 Value: real and imaginer part =  %f %f\n',real(Zs11)*50,imag(Zs11)*50);
        fprintf(fid_w_int,'ZSreq11 Value: real and imaginer part =  %f %f\n',real(Zs11)*50,imag(Zs11)*50);
        fprintf('ZSreq12 Value: real and imaginer part =  %f %f\n',real(Zs12)*50,imag(Zs12)*50);
        fprintf(fid_w_int,'ZSreq12 Value: real and imaginer part =  %f %f\n',real(Zs12)*50,imag(Zs12)*50);

        fprintf('ZSreq21 Value: real and imaginer part =  %f %f\n',real(Zs21)*50,imag(Zs21)*50);
        fprintf(fid_w_int,'ZSreq21 Value: real and imaginer part =  %f %f\n',real(Zs21)*50,imag(Zs21)*50);
        fprintf('ZSreq22 Value: real and imaginer part =  %f %f\n',real(Zs22)*50,imag(Zs22)*50);
        fprintf(fid_w_int,'ZSreq22 Value: real and imaginer part =  %f %f\n',real(Zs22)*50,imag(Zs22)*50);

%       % ZSreq value by circle Intersection of Vi-F
%       if ~isnan(ZsViF11) & ~isnan(ZsViF12) & ~isnan(ZsViF21) & ~isnan(ZsViF22)
%
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                  fprintf(fid_w_int,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%          fprintf(fid_w_int,'1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
```

```
%           fprintf('2. Intersection point on Vi-F;  ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%           fprintf(fid_w_int,'2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                   fprintf(fid_w_int,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf(fid_w_int,'1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%           fprintf(fid_w_int,'2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%
%       elseif ~isnan(ZsViF11) & ~isnan(ZsViF12)
%
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                   fprintf(fid_w_int,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf(fid_w_int,'1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%           fprintf(fid_w_int,'2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%
%       elseif ~isnan(ZsViF21) & ~isnan(ZsViF22)
%
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                   fprintf(fid_w_int,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf(fid_w_int,'1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%           fprintf(fid_w_int,'2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%
%       else
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                   fprintf(fid_w_int,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                   fprintf(fid_w_int,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
```

```
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                   fprintf('No intersection on Vi-F; ZSreq11,ZSreq12,ZSreq21,ZSreq22= NaN\n');
%          fprintf(fid_w_int,'No intersection on Vi-F; ZSreq11,ZSreq12,ZSreq21,ZSreq22= NaN\n');
%      end

        ZLreq=ZLreq1;  % print the first one in the intersection %
        ZSreq=Zs11; % print the first one in the intersection %


        % Calculation of many points of Zi in Region-3

        % GTreq cuts T1; take T1 as starting point and decrease test circle (rt1_step)  untill it does not cut or it
becomes tangent to GTreq
        rt1_step=rt1;

        fprintf('Calculation of many other points of Zi in Region-3; GTreq cuts only T1\n');
        fprintf(fid_w_oth,'Calculation of many other points of Zi in Region-3; GTreq cuts only T1\n');


        loop_count=0;
        while 1 == 1   % loop infinity

          rt1_step=rt1_step-step_for_sample_on_circle;

          [RintR,XintR]=circlint(RCGreq,XCGreq,rgreq,Rct1,Xct1,rt1_step);
          ZireqR1=RintR(1)+i*XintR(1);
          ZireqR2=RintR(2)+i*XintR(2);

           if (ZireqR1 == ZireqR2) | (isnan(ZireqR1) & isnan(ZireqR2)) break; end   % tangent case or no
intersection

          loop_count=loop_count+2;

            fprintf('Zireq%d Value: real and imaginer part =    %f %f\n',loop_count-1,real(ZireqR1)*50,
imag(ZireqR1)*50);
              fprintf(fid_w_oth,'Zireq%d Value: real and imaginer part =    %f %f\n',loop_count-
1,real(ZireqR1)*50, imag(ZireqR1)*50);
            fprintf('Zireq%d Value: real and imaginer part =    %f %f\n',loop_count,real(ZireqR2)*50,
imag(ZireqR2)*50);
          fprintf(fid_w_oth,'Zireq%d Value: real and imaginer part =   %f %f\n',loop_count,real(ZireqR2)*50,
imag(ZireqR2)*50);


          % Test Circle %
          %ZCTest=Rct1+i*Xct1
          %ZCZTest= ZCTest+rt1_step*exp(i*teta);
          %RCZTest=real(ZCZTest);
       %XCZTest=imag(ZCZTest);
          if switch_for_plot == 1
            %figure,plot(RCGTreq,XCGTreq,'k-',Ri1,Xi1,'m-',Ri2,Xi2,'m-.',RCZTest,XCZTest,'b-')
          %axis([-1 1 -3 2])
          %title('Zi-Plane (test)')
          %xlabel('real{Zi}'),ylabel('imag{Zi}'),grid
```

```
        %gtext('GTreq'),gtext('T1'),gtext('T2'),          gtext('Test')
    end

        % call empcali
        Zireq1=ZireqR1;
        Zireq2=ZireqR2;


[ZLreq1,ZLreq2,Zs11,Zs12,ZsViF11,ZsViF12,Zs21,Zs22,ZsViF21,ZsViF22,Rcv1,Xcv1,rv1,Rcv2,Xcv2,rv2]=e
mpcali(Zireq1,Zireq2,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn,switch_for_plot)

        % ZLreq value %
        fprintf(' ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
            fprintf(fid_w_oth,'     ZLreq1   Value:   real   and   imaginer   part   =        %f
%f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf(' ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
            fprintf(fid_w_oth,'     ZLreq2   Value:   real   and   imaginer   part   =        %f
%f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

        % ZSreq value by formulas
        fprintf(' ZSreq11 Value: real and imaginer part =  %f %f\n',real(Zs11)*50,imag(Zs11)*50);
            fprintf(fid_w_oth,'     ZSreq11   Value:   real   and   imaginer   part   =        %f
%f\n',real(Zs11)*50,imag(Zs11)*50);
        fprintf(' ZSreq12 Value: real and imaginer part =  %f %f\n',real(Zs12)*50,imag(Zs12)*50);
            fprintf(fid_w_oth,'     ZSreq12   Value:   real   and   imaginer   part   =        %f
%f\n',real(Zs12)*50,imag(Zs12)*50);

        fprintf(' ZSreq21 Value: real and imaginer part =  %f %f\n',real(Zs21)*50,imag(Zs21)*50);
            fprintf(fid_w_oth,'     ZSreq21   Value:   real   and   imaginer   part   =        %f
%f\n',real(Zs21)*50,imag(Zs21)*50);
        fprintf(' ZSreq22 Value: real and imaginer part =  %f %f\n',real(Zs22)*50,imag(Zs22)*50);
            fprintf(fid_w_oth,'     ZSreq22   Value:   real   and   imaginer   part   =        %f
%f\n',real(Zs22)*50,imag(Zs22)*50);

%       % ZSreq value by circle Intersection of Vi-F
%       if ~isnan(ZsViF11) & ~isnan(ZsViF12) & ~isnan(ZsViF21) & ~isnan(ZsViF22)
%
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%               fprintf(fid_w_oth,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%         fprintf(fid_w_oth,'1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%         fprintf(fid_w_oth,'2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%               fprintf(fid_w_oth,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
```

```
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf(fid_w_oth,'1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%           fprintf(fid_w_oth,'2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%
%       elseif ~isnan(ZsViF11) & ~isnan(ZsViF12)
%
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%               fprintf(fid_w_oth,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf(fid_w_oth,'1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%           fprintf(fid_w_oth,'2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%
%       elseif ~isnan(ZsViF21) & ~isnan(ZsViF22)
%
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%               fprintf(fid_w_oth,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf(fid_w_oth,'1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%           fprintf(fid_w_oth,'2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%
%       else
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%               fprintf(fid_w_oth,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%               fprintf(fid_w_oth,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%               fprintf('No intersection on Vi-F; ZSreq11,ZSreq12,ZSreq21,ZSreq22= NaN\n');
%       fprintf(fid_w_oth,'No intersection on Vi-F; ZSreq11,ZSreq12,ZSreq21,ZSreq22= NaN\n');
%       end

    end

elseif ~isnan(ZireqT21) & ~isnan(ZireqT22)
```

```
% T2 Intersection
Zireq1=ZireqT21;
Zireq2=ZireqT22;

fprintf('T2 Intersection:\n');
fprintf(fid_w_int,'T2 Intersection:\n');

fprintf('Zireq1 Value: real and imaginer part =  %f %f\n',real(Zireq1)*50,imag(Zireq1)*50);
fprintf(fid_w_int,'Zireq1 Value: real and imaginer part =  %f %f\n',real(Zireq1)*50,imag(Zireq1)*50);
fprintf('Zireq2 Value: real and imaginer part =  %f %f\n',real(Zireq2)*50,imag(Zireq2)*50);
fprintf(fid_w_int,'Zireq2 Value: real and imaginer part =  %f %f\n',real(Zireq2)*50,imag(Zireq2)*50);

[ZLreq1,ZLreq2,Zs11,Zs12,ZsViF11,ZsViF12,Zs21,Zs22,ZsViF21,ZsViF22,Rcv1,Xcv1,rv1,Rcv2,Xcv
2,rv2]=empcali(Zireq1,Zireq2,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn,switch_for_plot)

% ZLreq value %
fprintf('ZLreq1 Value: real and imaginer part =  %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
fprintf(fid_w_int,'ZLreq1 Value: real and imaginer part = %f
%f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
fprintf('ZLreq2 Value: real and imaginer part =  %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
fprintf(fid_w_int,'ZLreq2 Value: real and imaginer part = %f
%f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

% ZSreq value by formulas
fprintf('ZSreq11 Value: real and imaginer part =  %f %f\n',real(Zs11)*50,imag(Zs11)*50);
fprintf(fid_w_int,'ZSreq11 Value: real and imaginer part =  %f %f\n',real(Zs11)*50,imag(Zs11)*50);
fprintf('ZSreq12 Value: real and imaginer part =  %f %f\n',real(Zs12)*50,imag(Zs12)*50);
fprintf(fid_w_int,'ZSreq12 Value: real and imaginer part =  %f %f\n',real(Zs12)*50,imag(Zs12)*50);

fprintf('ZSreq21 Value: real and imaginer part =  %f %f\n',real(Zs21)*50,imag(Zs21)*50);
fprintf(fid_w_int,'ZSreq21 Value: real and imaginer part =  %f %f\n',real(Zs21)*50,imag(Zs21)*50);
fprintf('ZSreq22 Value: real and imaginer part =  %f %f\n',real(Zs22)*50,imag(Zs22)*50);
fprintf(fid_w_int,'ZSreq22 Value: real and imaginer part =  %f %f\n',real(Zs22)*50,imag(Zs22)*50);

%       % ZSreq value by circle Intersection of Vi-F
%       if ~isnan(ZsViF11) & ~isnan(ZsViF12) & ~isnan(ZsViF21) & ~isnan(ZsViF22)
%
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                 fprintf(fid_w_int,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%         fprintf(fid_w_int,'1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%         fprintf(fid_w_int,'2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                 fprintf(fid_w_int,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
```

```
%           fprintf('1. Intersection point on Vi-F;  ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf(fid_w_int,'1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%           fprintf(fid_w_int,'2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%
%       elseif ~isnan(ZsViF11) & ~isnan(ZsViF12)
%
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                  fprintf(fid_w_int,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf(fid_w_int,'1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%           fprintf(fid_w_int,'2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%
%       elseif ~isnan(ZsViF21) & ~isnan(ZsViF22)
%
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                  fprintf(fid_w_int,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf(fid_w_int,'1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%           fprintf(fid_w_int,'2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%
%       else
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                  fprintf(fid_w_int,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                  fprintf(fid_w_int,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                fprintf('No intersection on Vi-F; ZSreq11,ZSreq12,ZSreq21,ZSreq22= NaN\n');
%         fprintf(fid_w_int,'No intersection on Vi-F; ZSreq11,ZSreq12,ZSreq21,ZSreq22= NaN\n');
%     end


        ZLreq=ZLreq1; % print the first one in the intersection %
        ZSreq=Zs11; % print the first one in the intersection %
```

```
% Calculation of many points of Zi in Region-3

% GTreq cuts T2; take T2 as starting point and increase test circle (rt2_step)  untill it does not cut or it becomes tangent to GTreq
rt2_step=rt2;

fprintf('Calculation of many other points of Zi in Region-3; GTreq cuts only T2\n');
fprintf(fid_w_oth,'Calculation of many other points of Zi in Region-3; GTreq cuts only T2\n');


loop_count=0;
while 1 == 1    % loop infinity

  rt2_step=rt2_step+step_for_sample_on_circle;

  [RintR,XintR]=circlint(RCGreq,XCGreq,rgreq,Rct2,Xct2,rt2_step);
  ZireqR1=RintR(1)+i*XintR(1);
  ZireqR2=RintR(2)+i*XintR(2);

   if (ZireqR1 == ZireqR2) | (isnan(ZireqR1) & isnan(ZireqR2)) break; end   % tangent case or no intersection

  loop_count=loop_count+2;

    fprintf('Zireq%d Value: real and imaginer part =    %f %f\n',loop_count-1,real(ZireqR1)*50, imag(ZireqR1)*50);
     fprintf(fid_w_oth,'Zireq%d Value: real and imaginer part =    %f %f\n',loop_count-1,real(ZireqR1)*50, imag(ZireqR1)*50);
    fprintf('Zireq%d Value: real and imaginer part =    %f %f\n',loop_count,real(ZireqR2)*50, imag(ZireqR2)*50);
   fprintf(fid_w_oth,'Zireq%d Value: real and imaginer part =  %f %f\n',loop_count,real(ZireqR2)*50, imag(ZireqR2)*50);


  % Test Circle %
  %ZCTest=Rct2+i*Xct2
  %ZCZTest= ZCTest+rt2_step*exp(i*teta);
  %RCZTest=real(ZCZTest);
%XCZTest=imag(ZCZTest);

if switch_for_plot == 1
   %figure,plot(RCGTreq,XCGTreq,'k-',Ri1,Xi1,'m-',Ri2,Xi2,'m-.',RCZTest,XCZTest,'b-')
  %axis([-1 1 -3 2])
  %title('Zi-Plane (test)')
  %xlabel('real{Zi}'),ylabel('imag{Zi}'),grid
  %gtext('GTreq'),gtext('T1'),gtext('T2'), gtext('Test')
end

  % call empcali
  Zireq1=ZireqR1;
  Zireq2=ZireqR2;
```

```
[ZLreq1,ZLreq2,Zs11,Zs12,ZsViF11,ZsViF12,Zs21,Zs22,ZsViF21,ZsViF22,Rcv1,Xcv1,rv1,Rcv2,Xcv2,rv2]=e
mpcali(Zireq1,Zireq2,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn,switch_for_plot)

        % ZLreq value %
        fprintf(' ZLreq1 Value: real and imaginer part = %f %f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
              fprintf(fid_w_oth,' ZLreq1 Value: real and imaginer part = %f
%f\n',real(ZLreq1)*50,imag(ZLreq1)*50);
        fprintf(' ZLreq2 Value: real and imaginer part = %f %f\n',real(ZLreq2)*50,imag(ZLreq2)*50);
              fprintf(fid_w_oth,' ZLreq2 Value: real and imaginer part = %f
%f\n',real(ZLreq2)*50,imag(ZLreq2)*50);

        % ZSreq value by formulas
        fprintf(' ZSreq11 Value: real and imaginer part = %f %f\n',real(Zs11)*50,imag(Zs11)*50);
              fprintf(fid_w_oth,' ZSreq11 Value: real and imaginer part = %f
%f\n',real(Zs11)*50,imag(Zs11)*50);
        fprintf(' ZSreq12 Value: real and imaginer part = %f %f\n',real(Zs12)*50,imag(Zs12)*50);
              fprintf(fid_w_oth,' ZSreq12 Value: real and imaginer part = %f
%f\n',real(Zs12)*50,imag(Zs12)*50);

        fprintf(' ZSreq21 Value: real and imaginer part = %f %f\n',real(Zs21)*50,imag(Zs21)*50);
              fprintf(fid_w_oth,' ZSreq21 Value: real and imaginer part = %f
%f\n',real(Zs21)*50,imag(Zs21)*50);
        fprintf(' ZSreq22 Value: real and imaginer part = %f %f\n',real(Zs22)*50,imag(Zs22)*50);
              fprintf(fid_w_oth,' ZSreq22 Value: real and imaginer part = %f
%f\n',real(Zs22)*50,imag(Zs22)*50);

%       % ZSreq value by circle Intersection of Vi-F
%       if ~isnan(ZsViF11) & ~isnan(ZsViF12) & ~isnan(ZsViF21) & ~isnan(ZsViF22)
%
%                 fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%               fprintf(fid_w_oth,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%         fprintf(fid_w_oth,'1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%         fprintf(fid_w_oth,'2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%
%                 fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%               fprintf(fid_w_oth,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%         fprintf(fid_w_oth,'1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%         fprintf(fid_w_oth,'2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
```

```
%
%       elseif ~isnan(ZsViF11) & ~isnan(ZsViF12)
%
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                 fprintf(fid_w_oth,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%         fprintf(fid_w_oth,'1. Intersection point on Vi-F; ZSreq11 Value: real and imaginer part= %f
%f\n',real(ZsViF11)*50,imag(ZsViF11)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%         fprintf(fid_w_oth,'2. Intersection point on Vi-F; ZSreq12 Value: real and imaginer part= %f
%f\n',real(ZsViF12)*50,imag(ZsViF12)*50);
%
%       elseif ~isnan(ZsViF21) & ~isnan(ZsViF22)
%
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                 fprintf(fid_w_oth,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%           fprintf('1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%         fprintf(fid_w_oth,'1. Intersection point on Vi-F; ZSreq21 Value: real and imaginer part= %f
%f\n',real(ZsViF21)*50,imag(ZsViF21)*50);
%           fprintf('2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%         fprintf(fid_w_oth,'2. Intersection point on Vi-F; ZSreq22 Value: real and imaginer part= %f
%f\n',real(ZsViF22)*50,imag(ZsViF22)*50);
%
%       else
%                   fprintf('Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                 fprintf(fid_w_oth,'Vi-F: Rcv1= %f, Xcv1= %f, rv1= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
%                   fprintf('Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%                 fprintf(fid_w_oth,'Vi-F: Rcv2= %f, Xcv2= %f, rv2= %f, Rcn= %f, Xcn= %f, rn=
%f\n',Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
%               fprintf('No intersection on Vi-F; ZSreq11,ZSreq12,ZSreq21,ZSreq22= NaN\n');
%       fprintf(fid_w_oth,'No intersection on Vi-F; ZSreq11,ZSreq12,ZSreq21,ZSreq22= NaN\n');
%     end


  end

else

      fprintf('GTreq does not cut T1 or T2\n');
      fprintf(fid_w_int,'GTreq does not cut T1 or T2\n');
      fprintf(fid_w_oth,'GTreq does not cut T1 or T2\n');
end
```

```
fclose(fid_w_int);
fclose(fid_w_oth);


function
[ZLreq1,ZLreq2,Zs11,Zs12,ZsViF11,ZsViF12,Zs21,Zs22,ZsViF21,ZsViF22,Rcv1,Xcv1,rv1,Rcv2,Xcv2,rv2]=e
mpcali(Zireq1,Zireq2,Z11,Z12,Z21,Z22,Rmod,C,Zopt,teta,Rcn,Xcn,rn,switch_for_plot)
%/*************************************************************/
%/*                                                           */
%/*  File:          empcali.m                                 */
%/*                                                           */
%/*  Compiler:    Matlab v5.0                                 */
%/*                                                           */
%/*  Description:Find the intersections of two circles        */
%/*            in cartesian space (GTreq,T1,T2)               */
%/*                                                           */
%/*  Created:      Cemal Tepe 1999  Ph.D. Thesis              */
%/*                                                           */
%/*  Version:      1.0                                        */
%/*                                                           */
%/*  Notes:                                                   */
%/*  Zireq1 --> ZLreq1,Zs11,Zs12,ZsViF11,ZsViF12              */
%/*  Zireq2 --> ZLreq2,Zs21,Zs22,ZsViF21,ZsViF22              */
%/*                                                           */
%/*************************************************************/
%
%

% ZLreq value %
ZLreq1=((Z12*Z21)/(Z11-Zireq1)-Z22);
ZLreq2=((Z12*Z21)/(Z11-Zireq2)-Z22);


% ZSreq value  caluclated by  formulas (Zireq1)
Rcv1=(1+Rmod^2)*real(Zireq1)/C;
Xcv1=-imag(Zireq1);
Zcv1=Rcv1+i*Xcv1;
rv1=2*Rmod*real(Zireq1)/C;

A1=(Rcn-Rcv1)/(Xcv1-Xcn);
B1=(abs(Zireq1)^2-abs(Zopt)^2)/(2*(Xcv1-Xcn));
K1=(Rcn+A1*Xcn-A1*B1)/(1+A1^2);
D1=(B1^2+abs(Zopt)^2-2*Xcn*B1)/(1+A1^2);

Rs11=K1+sqrt(K1^2-D1);
Xs11=A1*Rs11+B1;
Rs12=K1-sqrt(K1^2-D1);
Xs12=A1*Rs12+B1;

% VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn, Zcv=conj(Zi)
% Rmod=0 --> Vireq=0,  but rn=0 --> Freq=Fmin not always true
if (Rmod == 0) & (rn == 0)
  Zs11=Zopt;
```

```
  Zs12=Zopt;
else
  Zs11=Rs11+i*Xs11;
  Zs12=Rs12+i*Xs12;
end

% ZSreq value caluclated by circle Intersection of Vi-F (Zireq1)
[RintViF1,XintViF1]=circlint(Rcv1,Xcv1,rv1,Rcn,Xcn,rn);
ZsViF11=RintViF1(1)+i*XintViF1(1);
ZsViF12=RintViF1(2)+i*XintViF1(2);

% Define VSWR (Vireq) Circles (center and radius)
ZVi= Zcv1+rv1*exp(i*teta);
RVi=real(ZVi);
XVi=imag(ZVi);

% Define Noise (Freq) Circles (center and radius)
Zcn=Rcn+i*Xcn;
ZF=Zcn+rn*exp(i*teta);
RF=real(ZF);
XF=imag(ZF);

if switch_for_plot == 1
  %figure,plot(RVi,XVi,'b-',RF,XF,'c-')
    %axis([-0.5 0.5 -1 3])
    %title('Zs-Plane (Noise and input VSWR) (Zireq1)')
    %xlabel('real{Zs}'),ylabel('imag{Zs}'),grid
    %gtext('Vireq circle'),gtext('Freq circle');
end

% ZSreq value  caluclated by  formulas (Zireq2)
Rcv2=(1+Rmod^2)*real(Zireq2)/C;
Xcv2=-imag(Zireq2);
Zcv2=Rcv2+i*Xcv2;
rv2=2*Rmod*real(Zireq2)/C;

A2=(Rcn-Rcv2)/(Xcv2-Xcn);
B2=(abs(Zireq2)^2-abs(Zopt)^2)/(2*(Xcv2-Xcn));
K2=(Rcn+A2*Xcn-A2*B2)/(1+A2^2);
D2=(B2^2+abs(Zopt)^2-2*Xcn*B2)/(1+A2^2);

Rs21=K2+sqrt(K2^2-D2);
Xs21=A2*Rs21+B2;
Rs22=K2-sqrt(K2^2-D2);
Xs22=A2*Rs22+B2;

% VSWR and Noise match, rn=rv=rt1=tr2=0, Zs=Zopt=Zcn, Zcv=conj(Zi)
% Rmod=0 --> Vi=0,  but rn=0 --> Freq=Fmin not always true
if (Rmod == 0) & (rn == 0)
  Zs21=Zopt;
  Zs22=Zopt;
else
```

```
  Zs21=Rs21+i*Xs21;
  Zs22=Rs22+i*Xs22;
end

% ZSreq value caluclated by circle Intersection of Vi-F (Zireq2)
[RintViF2,XintViF2]=circlint(Rcv2,Xcv2,rv2,Rcn,Xcn,rn);
ZsViF21=RintViF2(1)+i*XintViF2(1);
ZsViF22=RintViF2(2)+i*XintViF2(2);

% Define VSWR (Vireq) Circles (center and radius)
ZVi= Zcv2+rv2*exp(i*teta);
RVi=real(ZVi);
XVi=imag(ZVi);

% Define Noise (Freq) Circles (center and radius)
Zcn=Rcn+i*Xcn;
ZF=Zcn+rn*exp(i*teta);
RF=real(ZF);
XF=imag(ZF);

if switch_for_plot == 1
  %figure,plot(RVi,XVi,'b-',RF,XF,'c-')
    %axis([-0.5 0.5 -1 3])
    %title('Zs-Plane (Noise and input VSWR) (Zireq2)')
    %xlabel('real{Zs}'),ylabel('imag{Zs}'),grid
    %gtext('Vireq circle'),gtext('Freq circle');
end


function [xout,yout]=circlint(x1,y1,r1,x2,y2,r2)
%/*************************************************************/
%/*                                                           */
%/*  File:          circlint.m                                */
%/*                                                           */
%/*  Compiler:    Matlab v5.0                                 */
%/*                                                           */
%/*  Description:Find the intersections of two circles in     */
%/*            cartesian space                                */
%/*                                                           */
%/*  Created:      Cemal Tepe 1999  Ph.D. Thesis              */
%/*                                                           */
%/*  Version:      1.0                                        */
%/*                                                           */
%/*  Notes:                                                   */
%/*                                                           */
%/*  [xout,yout] = CIRCCIRC(x1,y1,r1,x2,y2,r2) finds the      */
%/*  points of intersection (if any), given two circles,      */
%/*  each defined by center and radius in x-y coordiantes.    */
%/*  In general, two points are returned.  When the circles   */
%/*  do not intersect or are identical, NaNs are returned.    */
%/*  When the two circles are tangent, two identical          */
%/*  points are returned.  All inputs must be scalars.        */
```

```
%/*                                                                    */
%/*                                                                  */
%/*******************************************************************/
%
%
  %fprintf('x1 = %f\n',x1);
  %fprintf('y1 = %f\n',y1);
  %fprintf('r1 = %f\n',r1);
  %fprintf('x2 = %f\n',x2);
  %fprintf('y2 = %f\n',y2);
  %fprintf('r2 = %f\n',r2);
if nargin ~= 6;  error('Incorrect number of arguments');  end

%  Input consistency test
if ~isequal(size(x1),size(y1),size(r1),size(x2),size(y2),size(r2),[1 1])
            error('Inputs must be scalars')
%elseif ~isreal([x1,y1,r1,x2,y2,r2])
                    error('inputs must be real')
%elseif r1 <=0 | r2 <=0
                    error('radius must be positive')
end

% Cartesian separation of the two circle centers
r3=sqrt((x2-x1).^2+(y2-y1).^2);

indx1=find(r3>r1+r2);  % too far apart to intersect
indx2=find(r2>r3+r1);  % circle one completely inside circle two
indx3=find(r1>r3+r2);  % circle two completely inside circle one
indx4=find((r3<10*eps)&(abs(r1-r2)<10*eps)); % circles identical
indx=[indx1(:);indx2(:);indx3(:);indx4(:)];

anought=atan2((y2-y1),(x2-x1));

%Law of cosines

aone=acos(-((r2.^2-r1.^2-r3.^2)./(2*r1.*r3)));

alpha1=anought+aone;
alpha2=anought-aone;

xout=[x1 x1]+[r1 r1].*cos([alpha1 alpha2]);
yout=[y1 y1]+[r1 r1].*sin([alpha1 alpha2]);

% Replace complex results (no intersection or identical)
% with NaNs.

if ~isempty(indx)
   xout(indx,:) = NaN;    yout(indx,:) = NaN;
end
```

ÖZGEÇMİŞ

Adı/Soyadı	Cemal TEPE
Doğum Tarihi:	20.05.1970
Doğum Yeri:	Sivas
İlkokul:	1976-1981 Günebakan İlkokulu
Ortaokul:	1981-1984 Okmeydanı Ortaokulu
Lise:	1984-1987 Şişli Kaptanpaşa Lisesi (Ortalama: 8.5/10)
Lisans:	1987-1991 İstanbul Teknik Üniversitesi Elektrik-Elektronik Fakültesi Elektronik ve Haberleşme Mühendisliği Bölümü Bitirme Konusu: Bikuadratik Alt Devrelerin Bilgisayar Yardımıyla Parametre ve Duyarlık Analizi (Ortalama: 73.8/100)
Yüksek Lisans:	1991-1994 İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü Elektrik-Elektronik Mühendisliği Anabilim Dalı, Elektronik ve Haberleşme Programı Tez Konusu: Nümerik Analiz Yöntemleri ve Programlama (Ortalama: 88.5/100)
Doktora:	1994-2000 Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Elektronik ve Haberleşme Mühendisliği Anabilim Dalı, Haberleşme Programı Tez Konusu: Mikrodalga Transistorunun Yapay Sinir Ağı ile Performans Analizi ve Modellenmesi (Derslerin ortalaması: 95.5/100)
Çalıştığı Kurumlar:	1993-1994 EKA, Güç Kaynakları ve Motor Kontrol Mühendisi 1994-Devam ALCATEL TELETAŞ, AR-GE Yazılım Geliştirme Mühendisi ve Proje Lideri