

RF Toolbox

For Use with MATLAB®

- Computation
- Visualization
- Programming

How to Contact The MathWorks:



www.mathworks.com Web
comp.soft-sys.matlab Newsgroup



support@mathworks.com Technical support
suggest@mathworks.com Product enhancement suggestions
bugs@mathworks.com Bug reports
doc@mathworks.com Documentation error reports
service@mathworks.com Order status, license renewals, passcodes
info@mathworks.com Sales, pricing, and general information



508-647-7000 Phone



508-647-7001 Fax



The MathWorks, Inc. Mail
3 Apple Hill Drive
Natick, MA 01760-2098

For contact information about worldwide offices, see the MathWorks Web site.

RF Toolbox User's Guide

© COPYRIGHT 2004 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

MATLAB, Simulink, Stateflow, Handle Graphics, and Real-Time Workshop are registered trademarks, and TargetBox is a trademark of The MathWorks, Inc.

Smith is a registered trademark of Analog Instruments Company, New Providence, NJ.

Touchstone is a U.S. registered trademark of Agilent Technologies, Inc.

Other product or brand names are trademarks or registered trademarks of their respective holders.

Printing History: June 2004 Online only New for Version 1.0 (Release 14)
August 2004 Online only Revised for Version 1.0.1 (Release 14+)

Introduction

1

What Is the RF Toolbox?	1-2
Work Directly with Network Parameter Data	1-2
Model RF Networks	1-2
Analyze Circuits Interactively	1-3
Other Features	1-3
 Help for Objects	 1-6

Working with Objects

2

Overview	2-2
 Creating an Object	 2-3
Constructing a New Object	2-3
Copying an Existing Object	2-5
 Properties and Property Values	 2-6
Setting Property Values	2-6
Retrieving Property Values	2-8
 Functions Acting on Objects	 2-10
 Examples	 2-12
RF Circuit Objects	2-12
RF Data Objects	2-19

3

Overview	3-2
Comments	3-3
Data Sections	3-4
S, Y, or Z Network Parameters	3-4
Power Data	3-6
Noise Parameters	3-8

RF Tool: An RF Analysis GUI

4

Overview	4-2
Sessions	4-2
Available RF Circuits	4-2
Getting Help	4-3
Opening RF Tool	4-4
Creating Circuits	4-5
Adding an RF Component to a Session	4-5
Adding an RF Network to a Session	4-7
Populating an RF Network	4-8
Reordering Circuits Within a Network	4-10
Deleting Circuits	4-11
Renaming Circuits	4-12
Setting Component Parameters	4-13
Analyzing Circuits	4-14
Plotting Circuit Data	4-15
Selecting the Plot Type	4-15
Generating the Plots	4-16

Importing RF Objects	4-18
Importing from the Workspace	4-18
Importing From a File	4-19
Exporting RF Objects	4-21
Exporting to the Workspace	4-21
Exporting to a File	4-22
Working with Sessions	4-24
Saving a Session	4-24
Opening an Existing Session	4-25
Renaming a Session	4-25
Starting a New Session	4-26

Function Reference

5

Functions — Categorical List	5-2
Circuit Objects	5-3
Data Objects	5-4
Calculations	5-4
Data Visualization	5-4
Utility Functions	5-4
Data I/O	5-5
Network Parameter Conversion	5-5
Graphical User Interface	5-6
Functions — Alphabetical List	5-7

Index

Introduction

What Is the RF Toolbox? (p. 1-2)

Introduces the RF Toolbox and describes its capabilities.

Help for Objects (p. 1-6)

Tells you where to find help for creating and working with RF Toolbox objects.

What Is the RF Toolbox?

The RF Toolbox enables you to create and combine RF (Radio Frequency) circuits for simulation in the frequency domain with support for both nonlinear and noise data. You can read, write, analyze, combine, and visualize RF network parameters.

RF technology is used to design and test RF circuits for cable television, wireless LAN, and other wireless applications such as broadcasting, radar, satellite communications, microwave relay, and mobile telephony.

Work Directly with Network Parameter Data

You can work directly with your own network parameter data or with data from files. Functions enable you to:

- Read and write RF data in Touchstone® SnP, YnP, ZnP, and HnP formats, as well as the MathWorks AMP format.
- Convert among S, Y, Z, h, T, and ABCD network parameters
- Plot your data on X-Y plane and polar plane plots, as well as Smith® charts
- Calculate cascaded S-parameters and de-embed S-parameters from a cascaded network
- Calculate input and output reflection coefficients, and voltage standing-wave ratio (VSWR) at the reflection coefficient

Model RF Networks

You can also assemble RF networks from circuit objects that model:

- Passive networks and general circuit elements using Touchstone .snp, .ynp, .znp, and .hnp files.
- Amplifiers and mixers using data from Touchstone format .s2p, .y2p, .z2p, and .h2p files as well as the MathWorks format .amp files.
- Transmission lines based on their geometries.
- LC ladder filters based on their electrical interactions.

From these components and previously created network objects, you can create cascaded, hybrid, parallel, and series networks.

Functions associated with these objects enable you to:

- Analyze network parameters at specified frequencies
- Calculate needed parameters
- Plot network parameters in X-Y plane, polar plane, and Smith chart formats.
- Extract parameters from an object
- Perform a variety of utility functions such as copying an object and listing valid parameters for visualization.

You can move network data among Touchstone format files or MathWorks format .amp files, your workspace, and circuit or data objects – wherever you need it.

The RF Blockset, which accepts data generated by the RF Toolbox, provides time-domain simulation

Analyze Circuits Interactively

A graphical tool, RF Tool, enables you to design, analyze, and visualize RF components and networks interactively, then export the circuits to your workspace or to a file for use with RF Toolbox functions and other circuit objects.

Other Features

- “RF Circuits” on page 1-3
- “Data Visualization” on page 1-4
- “Data Format Support” on page 1-4
- “Required Products” on page 1-5
- “Demos” on page 1-5

RF Circuits

The RF Toolbox provides circuit objects that enable you to model:

- Passive networks
- Amplifiers and mixers

- Transmission lines: coaxial, coplanar waveguide, microstrip, parallel-plate, two-wire, and general transmission
- SeriesRLC and shuntRLC circuits
- LC ladder filters: LC bandpass pi, LC bandpass tee, LC bandstop pi, LC bandstop tee, LC highpass pi, LC highpass tee, LC lowpass pi, and LC lowpass tee
- Networks: cascade, hybrid, parallel, and series

You can also model general circuit elements from data files.

Data Visualization

The RF Toolbox enables you to plot the network parameters of the circuits you create.

You can generate an X-Y plane plot, polar plane plot, or Smith chart of one or more selected network parameters directly from your data. You can also generate these plots from circuit objects you create using the RF Toolbox. See the `rfckt` and `rfddata` reference pages for information.

Data Format Support

The RF Toolbox supports the Touchstone SnP, YnP, ZnP, and HnP data file formats. It also introduces the MathWorks AMP format for amplifier data.

For information about the Touchstone file formats, see http://www.eda.org/pub/ibis/connector/touchstone_spec11.pdf.

For information about the AMP file format see Chapter 3, “AMP File Format.”

RF Analysis GUI

RF Tool is an RF analysis GUI that provides a visual interface for creating and analyzing RF (radio frequency) components and networks. You can create RF circuits quickly with the GUI. You can also import and export circuits from the MATLAB® workspace and RF data files.

RF Tool also provides the ability to set circuit parameters, analyze circuits, view their resulting S-parameter data, and visualize the data using X-Y plane plots, polar plane plots, and Smith charts.

Required Products

The RF Toolbox requires MATLAB. It provides simulation in the frequency domain. The RF Blockset, which can accept data generated by the RF Toolbox, provides time-domain simulation.

Demos

Demos of the RF Toolbox capabilities are available on the Demos tab of the MATLAB Help browser. These demos show examples of

- RF data objects
- RF circuit objects
- De-embedding S-parameters
- Placing circles on a Smith chart
- Designing impedance matching networks

Help for Objects

Follow these instructions to get specific information about RF Toolbox objects and their methods. Note that methods are treated and referred to as functions in the rest of this user's guide. For help in using objects, see Chapter 2, "Working with Objects."

Lists of Objects and Methods

To get a list of available circuit objects and methods, type `help rfckt` or `doc rfckt` at the command line.

Similarly for data objects, type `help rfdata` or `doc rfdata`.

Method Descriptions

To get detailed descriptions of the methods for circuit (`rfckt`) and data (`rfdata`) objects:

- At the command line, type `doc methodname`. For example, type `doc analyze`. If more than one product has a method or function by that name, MATLAB returns a list from which you can choose.
- At the command line, type `help rfckt.objecttype.methodname` for circuit objects, or `help rfdata.data.methodname` for data objects. For example, type `help rfckt.amplifier.analyze`.

Object and Property Descriptions

To get detailed information about a specific object and its properties:

- At the command line, type `doc rfckt.objecttype` or `doc rfdata.data`. For example, type `doc rfckt.amplifier`, `doc rfckt.lcbandpasspi`, or `doc rfdata.data`.
- At the command line, type `help rfckt.objecttype` or `help rfdata.data`. For example, type `help rfckt.amplifier`, or `help rfdata.data`.

Working with Objects

Overview (p. 2-2)	Introduces the types of objects used by the RF Toolbox.
Creating an Object (p. 2-3)	Describes two ways you can create objects.
Properties and Property Values (p. 2-6)	Tells you how to set and retrieve an object's property values.
Functions Acting on Objects (p. 2-10)	Introduces the RF Toolbox functions you can use to act on objects.
Examples (p. 2-12)	Shows you how to perform some basic operations on circuit and data objects.

Overview

The RF Toolbox uses circuit objects to create

- Circuit components such as amplifiers, transmission lines, and ladder filters
- RLC network components
- Networks of RF components. Networks can be cascaded, parallel, series, or hybrid. They can include both circuit and network components.

The RF Toolbox also uses data objects, created from files or derived from circuit objects, to hold analyzed data for existing components or parameter data for general components.

This chapter explains concepts you need to know to work with these objects.

Creating an Object

You can create a new object by doing one of the following

- “Constructing a New Object” on page 2-3
- “Copying an Existing Object” on page 2-5

Constructing a New Object

Use the `rfckt` function to construct a new circuit object such as an amplifier or transmission line. Objects can be amplifiers, mixers, transmission lines, ladder filters, or networks.

Each type of object has a name. For example, an amplifier is an `rfckt.amplifier` object. A cascaded network is an `rfckt.cascade` object. The following table lists the types of objects you can create.

Type of Object	Description
<code>rfckt.amplifier</code>	Amplifier, described by a data file
<code>rfckt.cascade</code>	Cascaded network,
<code>rfckt.coaxial</code>	Coaxial transmission line
<code>rfckt.cpw</code>	Coplanar waveguide transmission line
<code>rfckt.datafile</code>	General circuit, described by a data file
<code>rfckt.hybrid</code>	Hybrid connected network
<code>rfckt.lcbandpasspi</code>	LC bandpass pi network
<code>rfckt.lcbandpasstee</code>	LC bandpass tee network
<code>rfckt.lcbandstoppi</code>	LC bandstop pi network
<code>rfckt.lcbandstoptee</code>	LC bandstop tee network
<code>rfckt.lchighpasspi</code>	LC highpass pi network
<code>rfckt.lchighpasstee</code>	LC highpass tee network

Type of Object	Description
<code>rfckt.lclowpasspi</code>	LC lowpass pi network
<code>rfckt.lclowpasstee</code>	LC lowpass tee network
<code>rfckt.microstrip</code>	Microstrip transmission line
<code>rfckt.mixer</code>	Mixer, described by a data file
<code>rfckt.parallel</code>	Parallel connected network
<code>rfckt.parallelplate</code>	Parallel-plate transmission line
<code>rfckt.series</code>	Series connected network
<code>rfckt.seriesrlc</code>	Series RLC network
<code>rfckt.shuntrlc</code>	Shunt RLC network
<code>rfckt.twowire</code>	Two-wire transmission line
<code>rfckt.txline</code>	General transmission line

Every type of object has predefined fields called properties. The properties define the characteristics of a particular object. You can specify object property values by either:

- Specifying the property values when you create the object
- Creating an object with default property values, and changing some or all of the property values later

Example. This example creates a microstrip transmission line object with default properties. The output `h` is the handle of the newly created transmission line object.

```
h = rfckt.microstrip
```

The RF Toolbox lists the properties of the transmission line you created along with the associated default property values.

```
h =  
    Name: 'Microstrip Transmission Line'  
    nPort: 2
```



```
RFdata: []
Z0: []
PV: []
Loss: []
LineLength: 0.0100
StubMode: 'None'
Termination: 'None'
Width: 6.0000e-004
Height: 6.3500e-004
Thickness: 5.0000e-006
EpsilonR: 9.8000
SigmaCond: Inf
LossTangent: 0
```

The `rfckt.microstrip` reference page describes these properties in detail.

For examples of setting object properties, see “Properties and Property Values” on page 2-6.

Copying an Existing Object

If you already have an object with all or most property values set the way you want them, you can create a new one with the same property values by copying the first object. For example,

```
h2 = copy(h);
```

creates a new object which has the same property values as the microstrip transmission line object with handle `h`. You can later change specific property values for this copy.

Note The syntax `h2 = h` copies only the object handle and does not create a new object.

Properties and Property Values

All circuit (`rfckt`) and data (`rfdata`) objects have properties associated with them. The properties define the characteristics of a particular object.

Each property associated with an object is assigned a value. You can set the values of many properties or you can accept the default values. Some properties have read-only values.

To learn about properties that are specific to a specific type of circuit or data object, see the reference page for that type of object. For example, the `rfckt.amplifier` reference page describes the properties of amplifier objects.

Note The `rfckt` and `rfdata` reference pages list the available types of circuit and data objects and provide links to their reference pages.

- “Setting Property Values” on page 2-6
- “Retrieving Property Values” on page 2-8

Setting Property Values

You can set circuit and data object property values when you construct the object or at a later time using the `set` command.

- “Setting Property Values at Construction” on page 2-6
- “Setting Property Values for an Existing Object” on page 2-7

Setting Property Values at Construction

To set a property directly when you construct an object, include a property/value pair in the argument list of the object construction command. A property/value pair consists of:

- A string for the property name you want to set followed by a comma
- The associated property value.

Include as many property names in the argument list as there are properties you want to set directly. Any property values you do not set, retain their

default values. The circuit and data object reference pages list the valid values as well as the default value for each property.

This example creates a coaxial transmission line circuit object. Note that RF Toolbox lists the available properties and their values.

```
h = rfckt.coaxial('LineLength',0.05)

h =
    Name: 'Coaxial Transmission Line'
    nPort: 2
    RFdata: []
    Z0: []
    PV: []
    Loss: []
    LineLength: 0.0500
    StubMode: 'None'
    Termination: 'None'
    OuterRadius: 1.0000e-003
    InnerRadius: 5.0000e-005
    MuR: 1
    EpsilonR: 1
    SigmaCond: Inf
    SigmaDiel: 0
```

Setting Property Values for an Existing Object

Once you construct an object, you can modify its property values using the `set` command. You can use the `set` command to both:

- Set specific property values
- Display a listing of all property values you can set

For example, this code creates a copy of the coaxial transmission line from the previous example then changes it to be a series stub with open termination.

```
h2 = copy(h);
set(h2,'StubMode','series','Termination','open')
```

Note When you set any object property values, the strings for property names and their values are case-insensitive. In addition, you only need to type the shortest uniquely identifying string for the property name. You could have written the previous function call as

```
set(h2, 'st', 'series', 't', 'open')
```

To display a list of all properties you can set for a specific object, use the `set` command without specifying any property/value pairs. This example lists the properties you can set for the coaxial transmission line `h2`.

```
set(h2)

ans =
    LineLength: {}
    StubMode: {}
    Termination: {}
    OuterRadius: {}
    InnerRadius: {}
    MuR: {}
    EpsilonR: {}
    SigmaCond: {}
    SigmaDiel: {}
```

Retrieving Property Values

For an existing object, you can retrieve its property values using the `get` command. You can use the `get` command to both

- Retrieve specific property values for an object
- Display a list of properties associated with an object and their current values

For example, this code retrieves the value of the inner radius and outer radius for the coaxial transmission line in the previous example.

```
ir = get(h2, 'InnerRadius')
or = get(h2, 'OuterRadius')
```

```
ir =  
    5.0000e-005
```

```
or =  
    1.0000e-003
```

To display a list of properties associated with a specific object as well as their current values, use the `get` command without specifying a property name.

```
get(h2)  
    Name: 'Coaxial Transmission Line'  
    nPort: 2  
    RFdata: []  
    Z0: []  
    PV: []  
    Loss: []  
    LineLength: 0.0500  
    StubMode: 'series'  
    Termination: 'open'  
    OuterRadius: 1.0000e-003  
    InnerRadius: 5.0000e-005  
    MuR: 1  
    EpsilonR: 1  
    SigmaCond: Inf  
    SigmaDiel: 0
```

Note that this list includes read-only properties that do not appear when you type `set(h2)`. For a coaxial transmission line object, the read-only properties are `Name`, `nPort`, `RFdata`, `Z0`, `PV`, and `Loss`. The `Name` and `nPort` properties are fixed by the RF Toolbox. The remaining read-only property values are calculated and set by the toolbox when you analyze the component at specified frequencies.

Functions Acting on Objects

The RF Toolbox provides a variety of functions that act on circuit (rfckt) and data (rfdata) objects. The following table lists these functions and tells you the types of objects on which each can act. These functions are also referred to as methods.

“Examples” on page 2-12 illustrates the use of these functions.

Function	Types of Objects	Description
analyze	rfckt, rfdata	Calculate network parameters and noise figure for a circuit or data object at specified frequencies.
calculate	rfckt, rfdata	Calculate specified network parameters for a circuit or data object.
copy	rfckt, rfdata	Copy a circuit or data object.
extract	rfdata	Extract the specified network parameters from a data object and returns the result in a matrix.
getdata	rfckt	Get data object containing analyzed data.
listformat	rfckt, rfdata	List valid formats for a specified network parameter for a specified circuit or data object.
listparam	rfckt, rfdata	List valid network parameters for a specified circuit or data object.
plot	rfckt, rfdata	Plot network parameters from a circuit or data object on an X-Y plane.
polar	rfckt, rfdata	Plot network parameters from a circuit or data object on polar coordinates.
read	rfdata	Read RF network parameters from a file to a new or existing data object.

Function	Types of Objects	Description
smith	rfckt, rfdata	Plot network parameters from a circuit or data object on a Smith chart.
write	rfdata	Write RF data from a data object to a file.

Examples

These examples show you how to perform some basic operations with objects.

- “RF Circuit Objects” on page 2-12
- “RF Data Objects” on page 2-19

RF Circuit Objects

This example first creates three circuit (`rfckt`) objects: two transmission lines and an amplifier. It visualizes the amplifier data using RF Toolbox functions and shows you how to retrieve frequency data that was read from a file into the amplifier `rfckt` object. The example then analyzes the amplifier over a different frequency range and visualizes the results.

The example continues by cascading the three components to get a cascaded `rfckt` object. Finally, the example analyzes the cascaded network and visualizes its S-parameters over the original frequency range.

1 Create three `rfckt` objects and view their properties. Create a default transmission line, an amplifier described by the data in the data file `'default.amp'`, and a second transmission line. Use the `get` command to view the properties of the three `rfckt` objects.

Setting the interpolation method for the amplifier to `'cubic'` anticipates the interpolation that takes place later in this example when the amplifier is analyzed over a different frequency range.

```
% Create three circuit objects
FirstCkt = rfckt.txline;
SecondCkt = rfckt.amplifier('File','default.amp',...
    'IntpType','cubic');
ThirdCkt = rfckt.txline('LineLength',0.025,'PV',2.0e8);

% View their properties
PropertiesOfFirstCkt = get(FirstCkt)
PropertiesOfSecondCkt = get(SecondCkt)
PropertiesOfThirdCkt = get(ThirdCkt)
```


The toolbox displays the following output.

```
PropertiesOfFirstCkt =
    Name: 'Transmission Line'
    nPort: 2
    RFdata: []
    Z0: 50
    PV: 299792458
    Loss: 0
    LineLength: 0.0100
    StubMode: 'None'
    Termination: 'None'

PropertiesOfSecondCkt =
    Name: 'Amplifier'
    nPort: 2
    RFdata: [1x1 rfddata.data]
    File: 'default.amp'
    IntpType: 'cubic'
    OIP3: Inf
    NF: 0

PropertiesOfThirdCkt =
    Name: 'Transmission Line'
    nPort: 2
    RFdata: []
    Z0: 50
    PV: 200000000
    Loss: 0
    LineLength: 0.0250
    StubMode: 'None'
    Termination: 'None'
```

2 Change properties of rfckt objects. Use the set command to change the line length of the first transmission line (FirstCkt).

```
DefaultLength = get(FirstCkt,'LineLength')
set(FirstCkt,'LineLength',.001);
NewLength = get(FirstCkt,'LineLength')
```

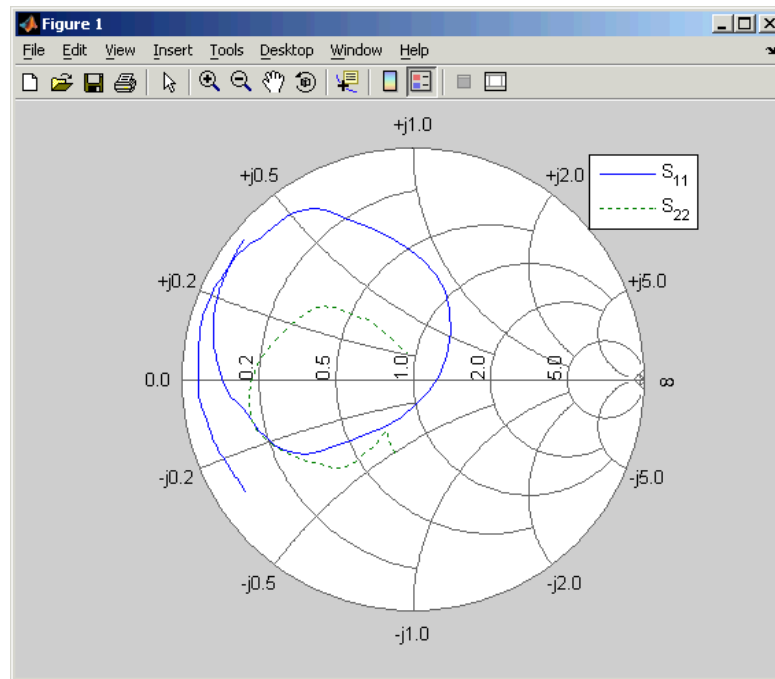
The toolbox displays the following output.

```
DefaultLength =  
    0.0100
```

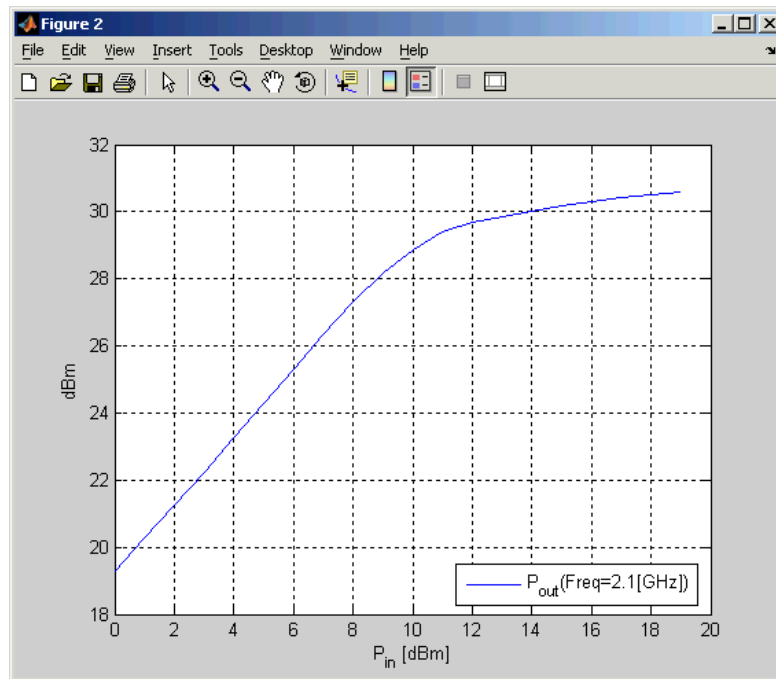
```
NewLength =  
    1.0000e-003
```

- 3 Plot the amplifier S11 and S22 parameters.** Use the smith function to plot the original S11 and S22 parameters of the amplifier (SecondCkt) on a ZY Smith chart. In step 1, the example read these parameters from the file default.amp. The amplifier has S-parameters over the frequency range 1GHz to 2.9GHz. It also has Pin-Pout data at the frequency 2.1GHz. Type `legend show` to display the legend.

```
lineseries1 = smith(SecondCkt,'S11','S22');  
set(lineseries1(1),'LineStyle','-','LineWidth',1);  
set(lineseries1(2),'LineStyle',':','LineWidth',1);  
legend show
```



- 4 Plot the amplifier Pout parameter.** Use the RF Toolbox plot function to plot Pin-Pout data, in dBm, of the amplifier (SecondCkt) on an X-Y plane figure
- ```
plot(SecondCkt, 'Pout', 'dBm');
legend('Location', 'SouthEast')
```



- 5 Get the frequency data that was obtained from the file default.amp.** When the RF Toolbox reads data from a file into an amplifier (rfckt) object, it also creates a data (rfdata) object and stores data from the file as properties of the data object. You can use the getdata function to retrieve the handle of this data object, then use the get command to retrieve property values. The following code gets the frequency values from the rfdata object associated with the SecondCkt amplifier object.

```
f = get(getdata(SecondCkt), 'Freq');
```

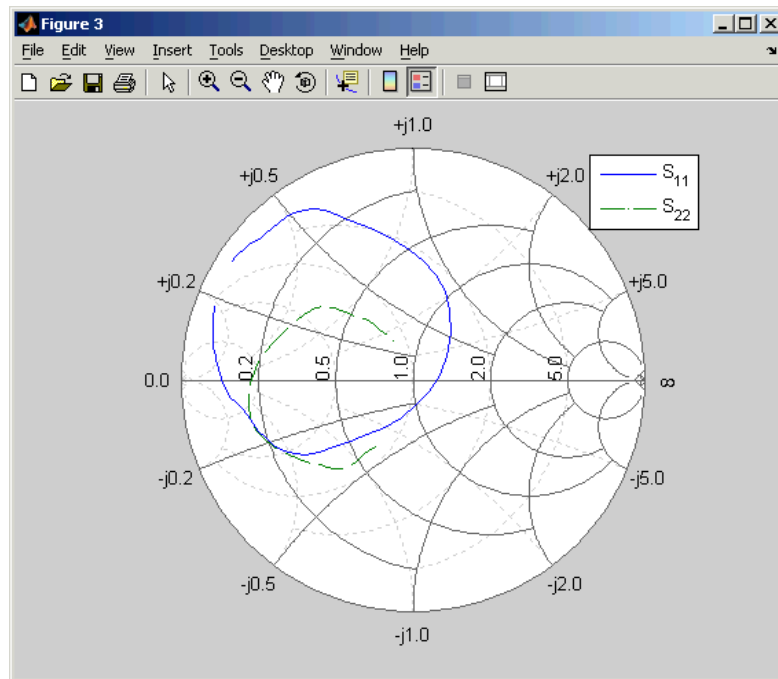
This example uses these frequencies to analyze the cascaded circuit in a later step.

- 6 Analyze the amplifier over a new frequency range and plot S11 and S22 again.** If you want to visualize the S-parameters of a circuit over a different frequency range, you need to first analyze the circuit over that frequency range.

```

analyze(SecondCkt,[1.85e9:1e7:2.55e9]);
figure
lineseries2 = smith(SecondCkt,'S11','S22','zy');
set(lineseries2(1),'LineStyle','-','LineWidth',1);
set(lineseries2(2),'LineStyle','-','LineWidth',1);
legend show

```



- 7 Create a cascaded circuit object and analyze it.** Cascade the three circuit objects to create a new cascaded circuit object, and then analyze it at the original amplifier frequencies

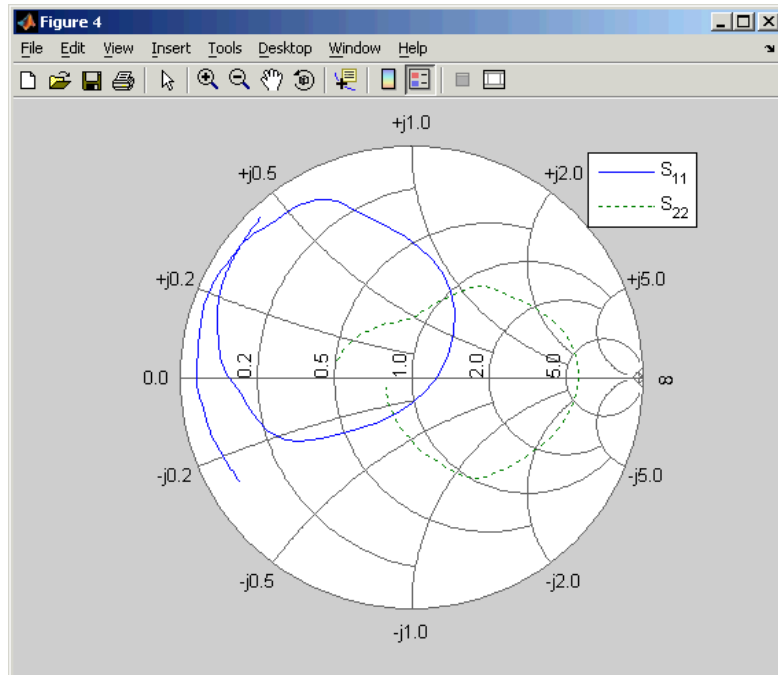
```

CascadedCkt = rfckt.cascade('Ckts',{FirstCkt,SecondCkt,...
 ThirdCkt});
analyze(CascadedCkt,f);

```

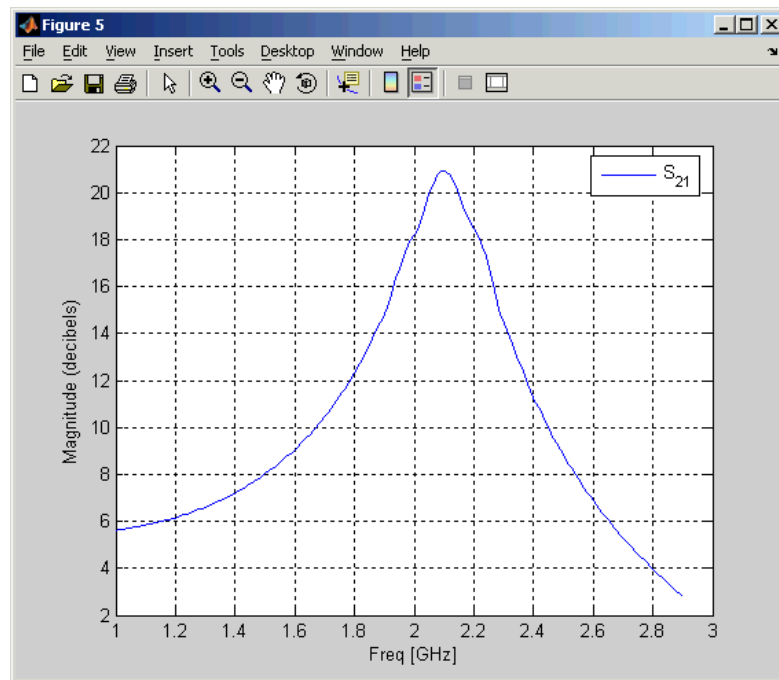
- 8 Plot the S11 and S22 parameters of the cascaded circuit.** Use the smith function to plot the S11 and S22 parameters of the cascaded circuit (CascadedCkt) on a Z Smith chart.

```
figure
lineseries3 = smith(CascadedCkt,'S11','S22','z');
set(lineseries3(1),'LineStyle','-','LineWidth',1);
set(lineseries3(2),'LineStyle',':','LineWidth',1);
legend show
```



- 9 Plot the S21 parameters of the cascaded circuit.** Use the RF Toolbox plot function to plot the S21 parameters of the cascaded circuit (CascadedCkt) on an X-Y plane.

```
figure
plot(CascadedCkt,'S21','dB');
legend show
```



## RF Data Objects

This example first creates an `rfdata.data` object by reading the S-parameters of a 2-port passive network stored in the Touchstone format data file `passive.s2p`, and visualizes the original S-parameters. Then, it calculates and visualizes the S-parameters over a new frequency range.

- 1 Read parameter data from a data file.** Use the `read` function to read the Touchstone data file `passive.s2p`. The `read` function creates an `rfdata.data` object and stores data from the file in the object's properties.

```
h = read(rfdata.data, 'passive.s2p');
```

- 2 View the properties of the data object.** Use the `get` command to view the object's properties.

```
PropertiesOfData = get(h)
```

The toolbox displays the following output.

```
PropertiesOfData =
 Name: 'rfdata.data object'
 Freq: [202x1 double]
 S_Parameters: [2x2x202 double]
 Z0: 50
 OIP3: Inf
 NF: 1
 IntpType: 'linear'
 ZS: 50
 ZL: 50
```

- 3 Change the properties of the data object.** Use the set command to change the Z1 (load impedance) property.

```
DefaultZ1 = get(h,'Z1')
set(h,'Z1',30+60i);
NewZ1 = get(h,'Z1')
```

The toolbox displays the following output.

```
DefaultZ1 =
 50

NewZ1 =
 30.0000 +60.0000i
```

- 4 List the network parameters that can be visualized.** Use the listparam function to display the network parameters of the rfdata.data object that you can visualize. You can plot these parameters using the smith, plot, and polar functions. For any parameter, the listformat function tells you what formats you can use for the plot.

This code first lists the parameters of the rfdata.data object h that you can visualize. It then lists the formats you can use to visualize the S11 parameters and GammaIn. The first format in each list is the default.

```
ParamsOfData = listparam(h)
FormatsOfS11 = listformat(h,'S11')
FormatsOfGammaIn = listformat(h,'GammaIn')
```



The toolbox displays the following output.

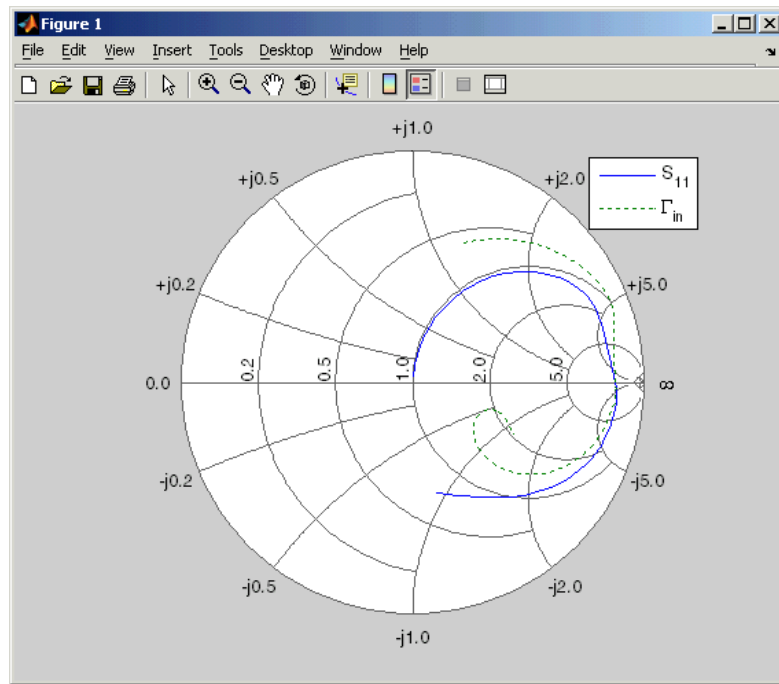
```
ParamsOfData =
 'S11'
 'S12'
 'S21'
 'S22'
 'GAMMAIn'
 'GAMMAOut'
 'VSWRIn'
 'VSWROut'
 'OIP3'
 'NF'

FormatsOfS11 =
 'dB'
 'Magnitude (decibels)'
 'Abs'
 'Mag'
 'Magnitude (linear)'
 'Angle'
 'Angle (degrees)'
 'Angle (radians)'
 'Real'
 'Imag'
 'Imaginary'

FormatsOfGammaIn =
 'dB'
 'Magnitude (decibels)'
 'Abs'
 'Mag'
 'Magnitude (linear)'
 'Angle'
 'Angle (degrees)'
 'Angle (radians)'
 'Real'
 'Imag'
 'Imaginary'
```

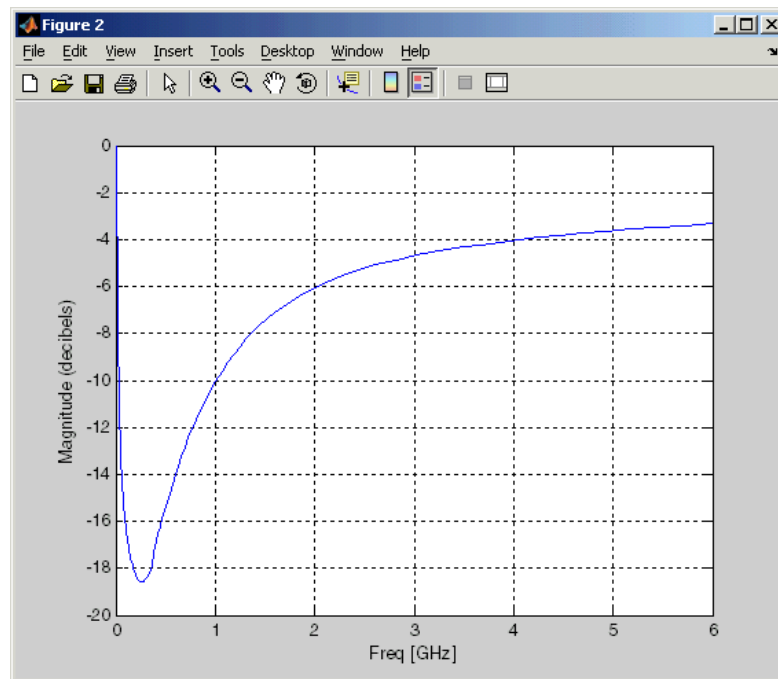
- 5 Plot the S11 parameters and the input reflection coefficient.** Use the smith function to plot the S11 parameters and the input reflection coefficient, GAMMAIn, on a Z Smith chart. Type legend show to display the legend.

```
lineseries1 = smith(h,'S11','GAMMAIn','z');
set(lineseries1(1),'LineStyle','-','LineWidth',1);
set(lineseries1(2),'LineStyle',':','LineWidth',1);
legend show
```



- 6 Plot the S21 parameters.** Use the plot function to plot the S21 parameters, in dB, on an X-Y plane.

```
figure
plot(h,'S21','db');
```

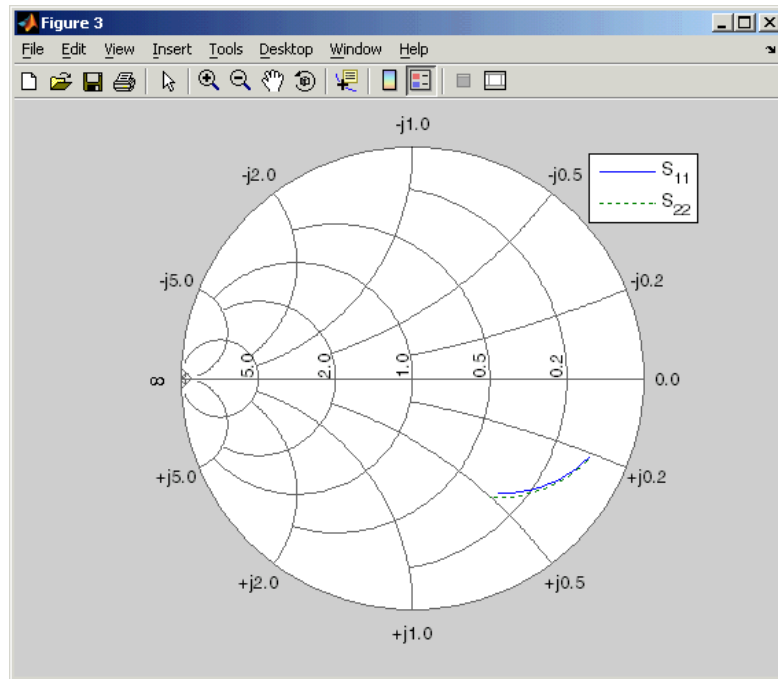


- 7 Analyze the data object over a new frequency range.** The original S-parameters are given over the frequency range 315KHz to 6GHz. If you want to see the S-parameters over a new frequency range, you need to analyze the data object first.

```
f = [1e9:1e8:3e9];
analyze(h,f);
```

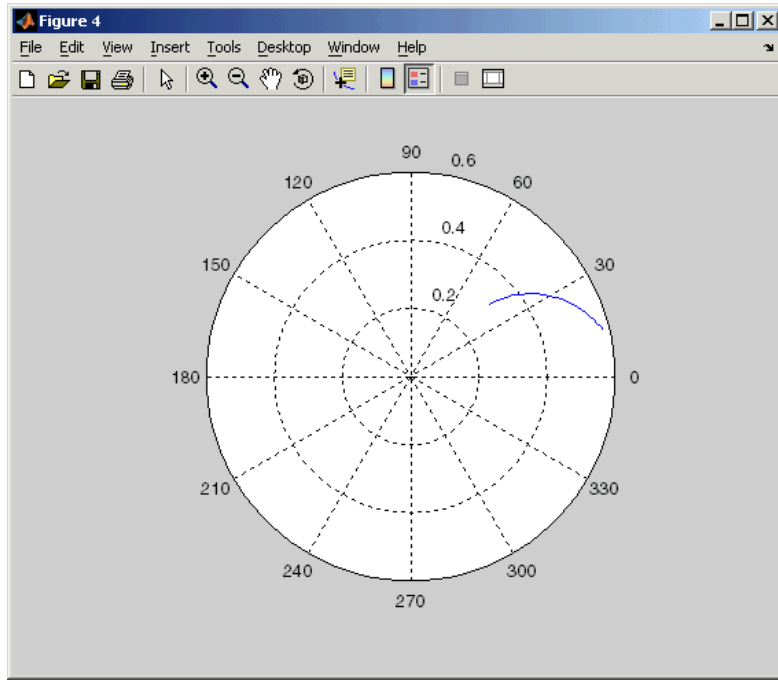
- 8 Plot the S11 and S22 parameters over the new frequency range.** Use the smith function to plot the S11 and S22 parameters on a Y Smith chart.

```
figure
lineseries2 = smith(h,'S11','S22','y');
set(lineseries2(1),'LineStyle','-','LineWidth',1);
set(lineseries2(2),'LineStyle',':','LineWidth',1);
legend show
```



- 9 Plot the  $S_{21}$  parameters over the new frequency range.** Use the `polar` function to plot the  $S_{21}$  parameters on a polar plane.

```
figure
polar(h, 's21');
```





# AMP File Format

---

AMP File Format includes these sections:

Overview (p. 3-2)

Introduces the AMP file format.

Comments (p. 3-3)

Defines the syntax for including comments in an AMP file.

Data Sections (p. 3-4)

Describes the formats for networks parameters, noise data, and power parameters.

### Overview

The AMP data file describes a single amplifier. Its format can contain three types of data. The following topics describe the sections of the file that contain this data.

- “S, Y, or Z Network Parameters” on page 3-4
- “Power Data” on page 3-6
- “Noise Parameters” on page 3-8

An AMP file must contain either power data and/or network parameter data. To accommodate analysis at more than one frequency, the file can contain more than one section of power data. Noise data is optional.

A sample AMP file, `default.amp`, ships with the RF Toolbox. It describes a nonlinear 2-port amplifier with noise. “RF Circuit Objects” on page 2-12 is an example that uses `default.amp`.



## Comments

An asterisk (\*) or an exclamation point (!) precedes a comment that appears on a separate line.

A semicolon (;) precedes a comment that appears following data on the same line.

## Data Sections

Each kind of data resides in its own section. Each section consists of a two-line header followed by lines of numeric data. Numeric values can be in any valid MATLAB format.

A new header indicates the end of the previous section. The data sections can appear in any order in the file.

In the following descriptions, brackets ([ ]) indicate optional data or characters. All values are case-insensitive.

- “S, Y, or Z Network Parameters” on page 3-4
- “Power Data” on page 3-6
- “Noise Parameters” on page 3-8

### S, Y, or Z Network Parameters

#### Header Line 1

The first line of the header has the format

```
Keyword [Parameter] [R[REF][=]value]
```

Keyword indicates the type of network parameter. It can be S[PARAMETERS], Y[PARAMETERS], or Z[PARAMETERS]. Parameter indicates the form of the data. It can be MA, DB, or RI. The default for S-parameters is MA. The default for Y- and Z-parameters is RI. R[REF][=]value is the reference impedance. The default reference impedance is 50 ohms.

The following table explains the meaning of the allowable Parameter values.

| Parameter | Description                                                                                 |
|-----------|---------------------------------------------------------------------------------------------|
| MA        | Data is given in (magnitude, angle) pairs with angle in degrees (default for S-parameters). |
| DB        | Data is given in (dB-magnitude, angle) pairs with angle in degrees.                         |
| RI        | Data is given in (real, imaginary) pairs (default for Y- and Z-parameters).                 |

This example of a first line indicates that the section contains S-parameter data given in (real, imaginary) pairs, and that the reference impedance is 50 ohms.

```
S RI R 50
```

## Header Line 2

The second line of the header has the format

```
Independent_variable Units
```

The data in a section is a function of the `Independent_variable`. Currently, for S-, Y-, and Z-parameters, the value of `Independent_variable` is always `FREQ`. `Units` indicates the default units of the frequency data. It can be GHz, MHz, or KHz. You must specify `Units`, but you can override this default on any given line of data.

This example of a second line indicates that the default units for frequency data is GHz.

```
FREQ GHZ
```

## Data

The data that follows the header typically consists of nine columns.

The first column contains the frequency points where network parameters were measured. If the frequency is given in units other than those you specified as the default, you must follow the value with the appropriate units; there should be no intervening spaces. For example,

```
FREQ GHZ
1000MHZ ...
2000MHZ ...
3 ...
4 ...
5 ...
```

Columns two through nine contain 2-port network parameters in the order N11, N21, N12, N22. Similar to the Touchstone format, each `Nnn` corresponds to two consecutive columns of data in the chosen form: MA, DB, or RI. The data can be in any valid MATLAB format.

This example is derived from the file `default.amp`. A comment line explains the column arrangement of the data.

```
S RI R 50
FREQ GHZ
* FREQ reS11 imS11 reS21 imS21 reS12 imS12 reS22 imS22
 1.00 -0.724725 -0.481324 -0.685727 1.782660 0.000000 0.000000 -0.074122 -0.321568
 1.01 -0.731774 -0.471453 -0.655990 1.798041 0.001399 0.000463 -0.076091 -0.319025
 1.02 -0.738760 -0.461585 -0.626185 1.813092 0.002733 0.000887 -0.077999 -0.316488
```

## Power Data

An AMP file describes power data as input power-dependent output power.

### Header Line 1

The first line of the header has the format

```
Keyword [Units]
```

For power data, `Keyword` must be `POUT`, indicating that this section contains power data. Because output power is complex, `Units` indicates the default units of the magnitude of the output power data. It can be `dBW`, `dBm`, `mW`, or `W`. The default is `W`. You can override this default on any given line of data.

The following table explains the meaning of the allowable `Units` values.

| Units | Description                          |
|-------|--------------------------------------|
| dBW   | Decibels referenced to one watt      |
| dBm   | Decibels referenced to one milliwatt |
| mW    | Milliwatts                           |
| W     | Watts                                |

This example of a first line indicates that the section contains output power data whose magnitude is assumed to be in decibels referenced to one milliwatt, unless other units are specified.

```
POUT dBm
```

## Header Line 2

The second line of the header has the format

```
Keyword [Units] FREQ[=]value
```

Keyword must be PIN. Units indicates the default units of input power data. It can be dBW, dBm, mW, or W. The default is W. You can override this default on any given line of data. FREQ[=]value is the frequency point at which the power is measured. The value must include the units as GHz, MHz, kHz, or Hz.

This example of a second line indicates that the section contains input power data that is assumed to be in decibels referenced to one milliwatt, unless other units are specified. It also indicates that the power data was measured at a frequency of 2.1E+009Hz.

```
PIN dBm FREQ=2.1E+009Hz
```

## Data

The data that follows the header typically consists of three columns.

The first column contains input power data in increasing order. The second column contains the corresponding output power magnitude. The third column contains the output phase shift in degrees. If all phases are zero, you can omit the third column.

If the power is given in units other than those you specified as the default, you must follow the value with the appropriate units; there should be no intervening spaces.

This example is derived from the file `default.amp`. A comment line explains the column arrangement of the data.

```
POUT dbm
PIN dBm FREQ = 2.10GHz
* Pin Pout Phase(degrees)
 0.0 19.28 0.0
 1.0 20.27 0.0
 2.0 21.26 0.0
```

## Noise Parameters

### Header Line 1

The first line of the header has the format

```
Keyword
```

Keyword must be NOI[SE].

### Header Line 2

The second line of the header has the format

```
Variable Units
```

Variable must be `FREQ`. Units indicates the default units of the frequency data. It can be `GHZ`, `MHZ`, or `KHZ`. You can override this default on any given line of data. This example of a second line indicates that frequency data is assumed to be in `GHZ`, unless other units are specified.

```
FREQ GHZ
```

### Data

The data that follows the header typically consists of five columns.

The first column contains the frequency points at which noise parameters were measured. If the frequency is given in units other than those you specified as the default, you must follow the value with the appropriate units; there should be no intervening spaces. For example,

```
NOI
FREQ GHZ
1000MHZ ...
2000MHZ ...
3 ...
4 ...
5 ...
```

Columns two through five contain, in order,

- Minimum noise figure in decibels
- Magnitude of the source reflection coefficient to realize minimum noise figure

- Phase in degrees of the source reflection coefficient
- Effective noise resistance normalized to the reference impedance of the network parameters

This example is derived from the file `default.amp`. A comment line explains the column arrangement of the data.

```
NOI RN
FREQ GHz
* Freq Fmin(dB) GammaOpt(MA:Mag) GammaOpt(MA:Ang) RN/Zo
 1.90 10.200000 1.234000 -78.400000 0.240000
 1.93 12.300000 1.235000 -68.600000 0.340000
 2.06 13.100000 1.254000 -56.700000 0.440000
 2.08 13.500000 1.534000 -52.800000 0.540000
 2.10 13.900000 1.263000 -44.400000 0.640000
```





# RF Tool: An RF Analysis GUI

---

|                                        |                                                                       |
|----------------------------------------|-----------------------------------------------------------------------|
| Overview (p. 4-2)                      | Introduction to RF Tool                                               |
| Opening RF Tool (p. 4-4)               | How to start RF Tool                                                  |
| Creating Circuits (p. 4-5)             | Creating RF circuit objects in RF Tool                                |
| Deleting Circuits (p. 4-11)            | Deleting RF circuit objects from an RF Tool session                   |
| Renaming Circuits (p. 4-12)            | Renaming components and networks in an RF Tool session                |
| Setting Component Parameters (p. 4-13) | Setting parameter values of RF component objects                      |
| Analyzing Circuits (p. 4-14)           | Setting parameters for circuit analysis and performing the analysis   |
| Plotting Circuit Data (p. 4-15)        | Plotting file data or the results of a circuit analysis               |
| Importing RF Objects (p. 4-18)         | Importing RF circuit objects from a file or from the MATLAB workspace |
| Exporting RF Objects (p. 4-21)         | Exporting RF circuit objects to a file or to the MATLAB workspace     |
| Working with Sessions (p. 4-24)        | Working with RF Tool sessions                                         |

## Overview

The RF Tool is a GUI that provides a visual interface for creating and analyzing RF (radio frequency) components and networks. You can use the RF Tool as a convenient alternative to command line RF circuit design and analysis functions that come with the RF Toolbox.

RF Tool provides the ability to set circuit parameters, analyze the circuits, and display their S-parameters in both tabular and graphical form using X-Y plots, polar plots, and Smith charts. You can also import and export circuit data from the MATLAB workspace and RF data files.

RF Tool is available on supported UNIX and Windows platforms.

## Sessions

The work you do with this tool is organized into sessions. Each session is a collection of independent RF circuits, which can be RF components or RF networks. You can save sessions and then load them for later use.

See “Working with Sessions” on page 4-24 for more information.

## Available RF Circuits

The following tables lists the RF components and networks that you can create using RF Tool. These are the RF components:

| <b>RF Component</b>                  | <b>Corresponding RF Toolbox Function</b> |
|--------------------------------------|------------------------------------------|
| Data File                            | rfckt.datafile                           |
| Coaxial Transmission Line            | rfckt.coaxial                            |
| Coplanar Waveguide Transmission Line | rfckt.cpw                                |
| Microstrip Transmission Line         | rfckt.microstrip                         |
| Parallel-Plate Transmission Line     | rfckt.parallelplate                      |
| Transmission Line                    | rfckt.txline                             |
| Two-Wire Transmission Line           | rfckt.twowire                            |

| <b>RF Component</b> | <b>Corresponding RF Toolbox Function</b> |
|---------------------|------------------------------------------|
| Series RLC          | <code>rfckt.seriesrlc</code>             |
| Shunt RLC           | <code>rfckt.shuntrlc</code>              |
| LC Bandpass Pi      | <code>rfckt.lcbandpasspi</code>          |
| LC Bandpass Tee     | <code>rfckt.lcbandpasstee</code>         |
| LC Bandstop Pi      | <code>rfckt.lcbandstoppi</code>          |
| LC Bandstop Tee     | <code>rfckt.lcbandstoptee</code>         |
| LC Highpass Pi      | <code>rfckt.lchighpasstee</code>         |
| LC Highpass Tee     | <code>rfckt.lchighpasstee</code>         |
| LC Lowpass Pi       | <code>rfckt.lclowpasspi</code>           |
| LC Lowpass Tee      | <code>rfckt.lclowpasstee</code>          |

The following table lists the available RF networks.

| <b>RF Network</b>          | <b>Corresponding RF Toolbox Function</b> |
|----------------------------|------------------------------------------|
| Cascaded Network           | <code>rfckt.cascade</code>               |
| Series Connected Network   | <code>rfckt.series</code>                |
| Parallel Connected Network | <code>rfckt.parallel</code>              |
| Hybrid Connected Network   | <code>rfckt.hybrid</code>                |

## Getting Help

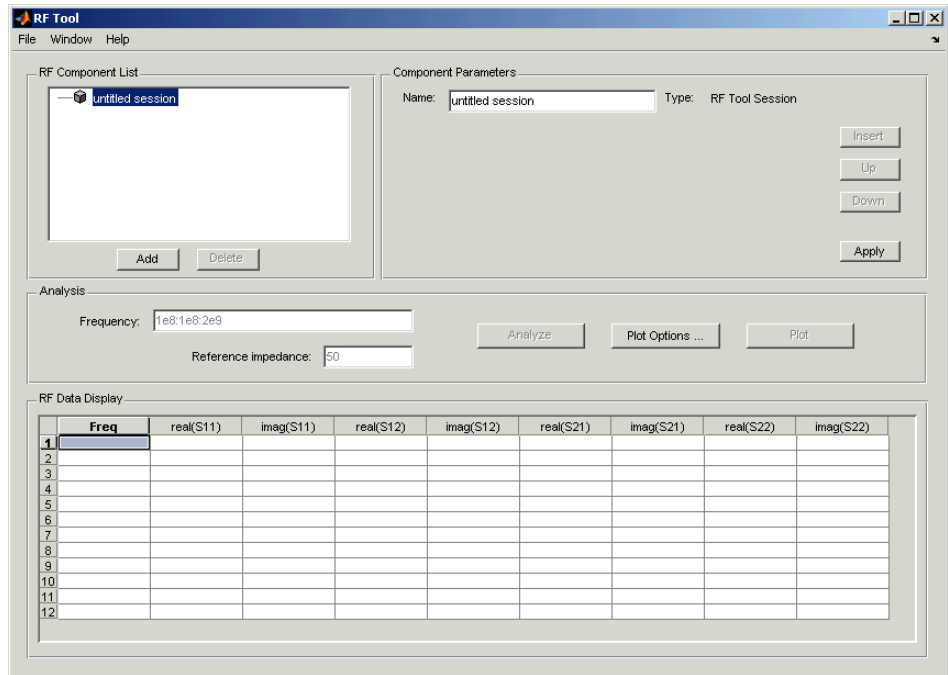
At any time, you can access the **Help** menu to see complete Help information on the RF Tool, RF Toolbox, and RF Demos.

## Opening RF Tool

To open RF Tool, type

```
rftool
```

The RF Tool opens with a new untitled session.



To give the session a name:

- 1 Type the desired name in the **Name** field of the **Component Parameters** panel.
- 2 Click **Apply**.

## Creating Circuits

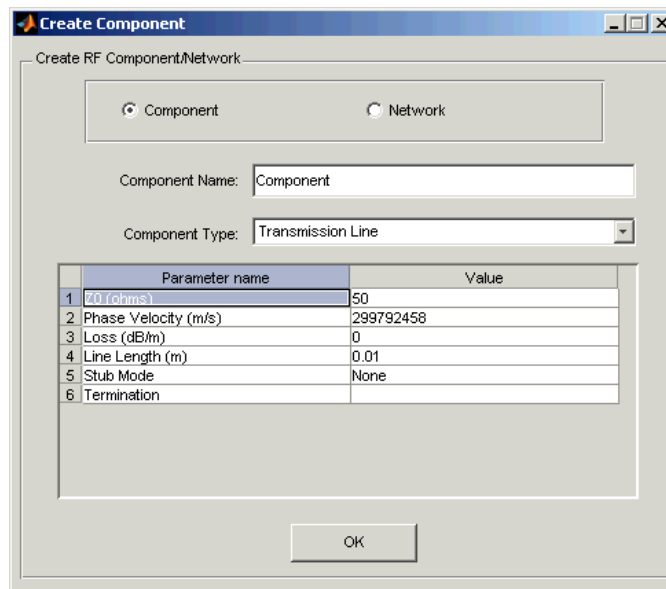
In the RF Tool, you can create circuits that include RF components and RF networks. Networks can contain both components and other networks. “Available RF Circuits” on page 4-2 lists the kinds of RF circuits you can create.

Topics in this section include:

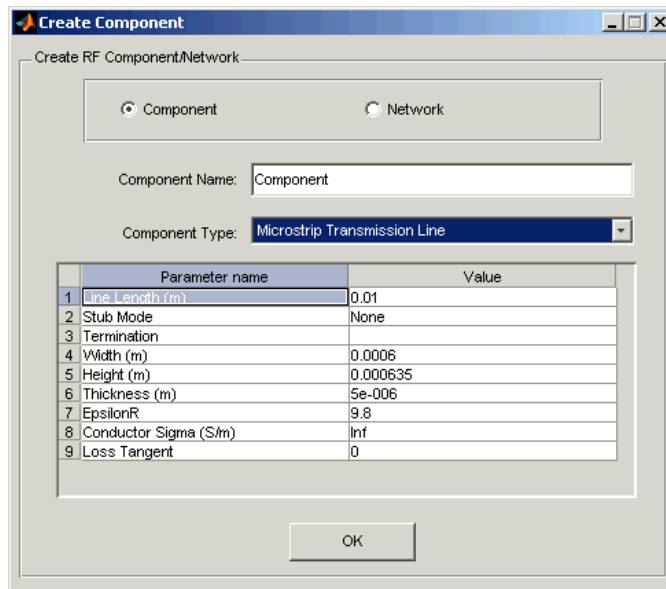
- “Adding an RF Component to a Session” on page 4-5
- “Adding an RF Network to a Session” on page 4-7
- “Populating an RF Network” on page 4-8
- “Reordering Circuits Within a Network” on page 4-10

## Adding an RF Component to a Session

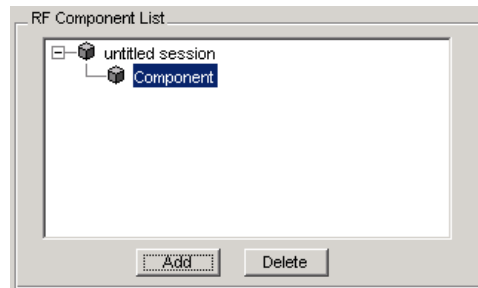
- 1 In the **RF Component List** of the RF Tool, click **Add** to open the **Create Component** dialog box.



- In the **Create Component** dialog box, click the **Component** radio button if it is not already selected.
- In the **Component Name** field, enter a name for the component.
- From the **Component Type** pull-down menu, select the type of RF component you want to create. The RF Tool displays a list of that component's parameters in the **Create Component** dialog box. See "Available RF Circuits" on page 4-2 for a list of the available components.



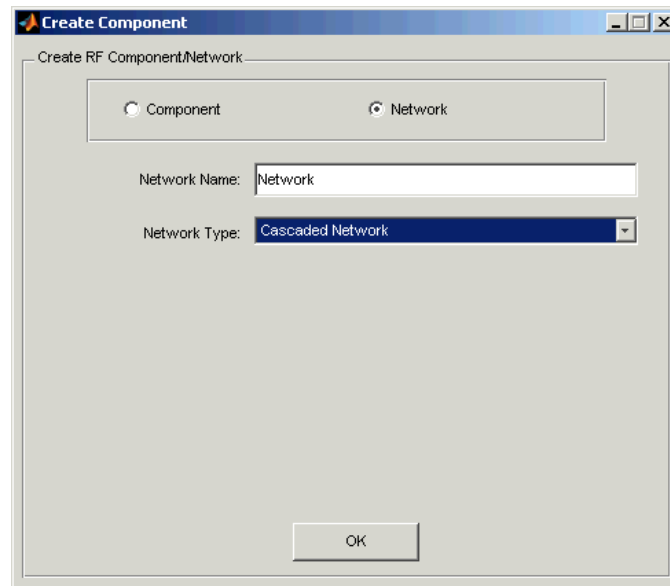
- Modify the parameters as necessary. See "Setting Component Parameters" on page 4-13 for more information.
- Click **OK**. The RF Tool adds the component to your session.



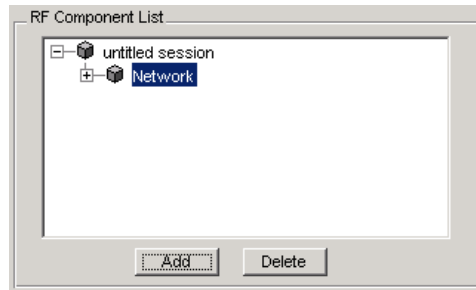
## Adding an RF Network to a Session

To create a network, first add the network to your session, then populate the network by adding components and networks to it.

You add an RF network to a session in much the same way you add a component. However, to create a network, you must select the **Network** radio button. See “Adding an RF Component to a Session” on page 4-5 for details.



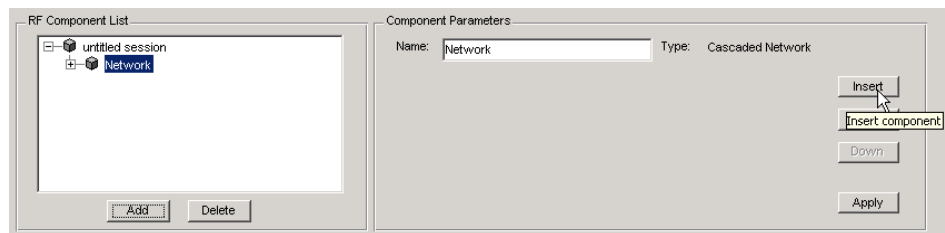
The RF Component List panel shows the new network.



## Populating an RF Network

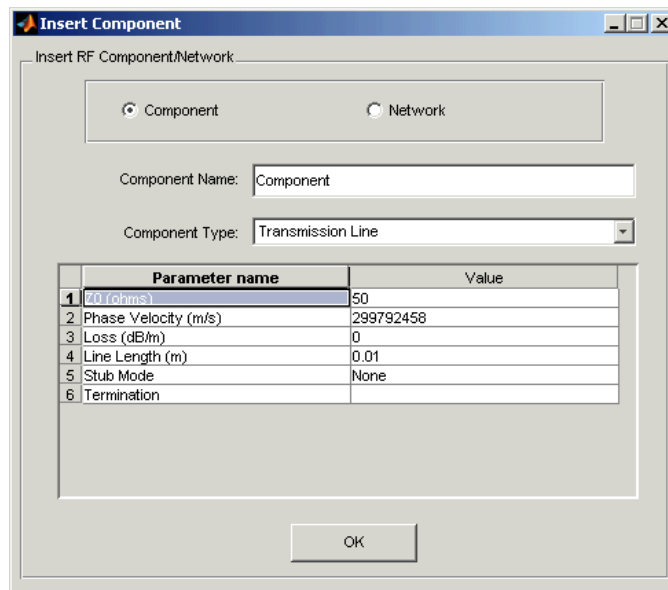
After you create a network, you must populate it with RF components and networks.

- 1 In the **RF Component List** panel of RF Tool, select the network component you want to modify, and then click **Insert** in the **Component Parameters** panel.



- 2 The **Insert Component** dialog box appears.

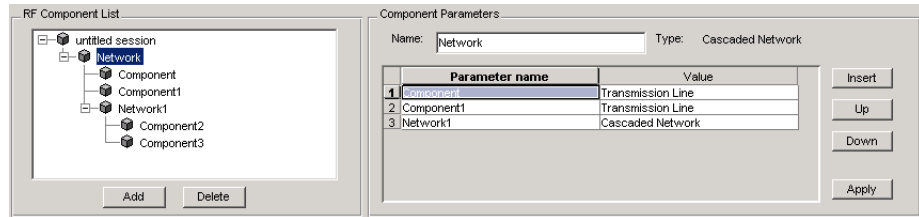




- 3** You insert a component or network in a network in much the same way you add one to a session.

In the **Insert Component** dialog box, start by selecting the **Component** or **Network** radio button as appropriate. Continue by giving the component or network a name, and selecting the appropriate type. If you are inserting a component, modify parameter values as necessary. See “Adding an RF Component to a Session” on page 4-5 or “Adding an RF Network to a Session” on page 4-7 for details.

As you insert components and networks into a network, they are reflected in the **RF Component List** and **Component Parameters** panels. This is an example of a cascaded network that contains two components and a network. The subnetwork, in turn, contains two components.

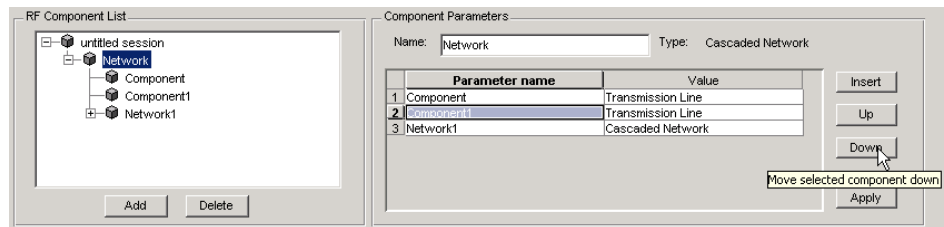


## Reordering Circuits Within a Network

To change the order of the components and networks within a network:

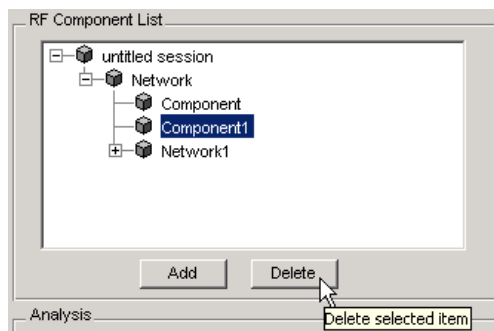
- 1 In the **RF Component List** panel, select the network whose circuits you want to reorder.
- 2 In the **Component Parameters** panel, select the circuit whose position you want to change.
- 3 Click **Up** or **Down** until the circuit is where you want it.

In the example below, clicking **Down** after selecting **Network** in the **RF Component List** panel and selecting **Component1** in the **Component Parameters** panel, would reverse the positions of **Component1** and **Network1**.



## Deleting Circuits

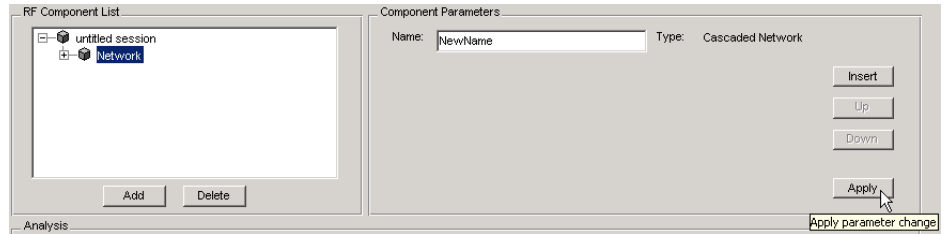
To delete a circuit from your session, select the circuit in the **RF Component List** panel and click **Delete**. If you select a network, the network and all its circuits are deleted.



## Renaming Circuits

To rename a component or a network:

- 1 Select the component or network in the **RF Component List** panel.
- 2 Type the new name in the **Name** field of the **Component Parameters** panel.
- 3 Click **Apply**.



To rename a session see “Renaming a Session” on page 4-25.

## Setting Component Parameters

You can change the values of component parameters. To modify these values:

- 1** Select the component in the **RF Component List** panel.
- 2** In the **Component Parameters** panel, select the value you want to change, and enter the new value.

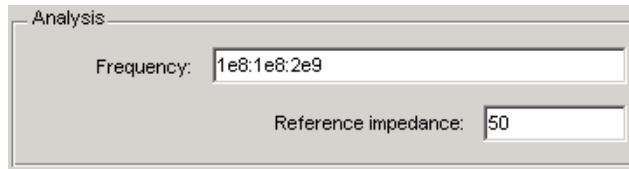
Valid values for component parameters are listed on the corresponding RF Toolbox reference page. Use the links in “Available RF Circuits” on page 4-2 to access these pages. All values are case-insensitive.

- 3** Click **Apply**.

## Analyzing Circuits

Once you have added your circuits, you can analyze them with the RF Tool:

- 1 Select the component or network you want to analyze in the **RF Component List** panel of the RF Tool.
- 2 In the **Analysis** panel, enter the analysis frequency range and step size in Hz in the **Frequency** field. Enter  $Z_0$  in the **Reference impedance** field. You can specify these values as MATLAB workspace variables or as valid MATLAB expressions.



- 3 Click **Analyze**. This populates the data display panel with the component's S-parameter data as a function of the specified frequencies.

The following figure shows the analysis data for a default Microstrip transmission line at the default frequencies and reference impedance shown in step 2.

|    | Freq     | real(S11) | imag(S11) | real(S12) | imag(S12) | real(S21) | imag(S21) | real(S22) | imag(S22) |
|----|----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| 1  | 1e+008   | +0.000    | +0.000    | +0.999    | -0.054    | +0.999    | -0.054    | +0.000    | +0.000    |
| 2  | 2e+008   | +0.000    | +0.000    | +0.994    | -0.107    | +0.994    | -0.107    | +0.000    | +0.000    |
| 3  | 3e+008   | +0.000    | +0.000    | +0.987    | -0.161    | +0.987    | -0.161    | +0.000    | +0.000    |
| 4  | 4e+008   | +0.000    | +0.001    | +0.977    | -0.213    | +0.977    | -0.213    | +0.000    | +0.001    |
| 5  | 5e+008   | +0.000    | +0.001    | +0.964    | -0.266    | +0.964    | -0.266    | +0.000    | +0.001    |
| 6  | 6e+008   | +0.000    | +0.001    | +0.948    | -0.317    | +0.948    | -0.317    | +0.000    | +0.001    |
| 7  | 7e+008   | +0.000    | +0.001    | +0.930    | -0.367    | +0.930    | -0.367    | +0.000    | +0.001    |
| 8  | 8e+008   | +0.001    | +0.001    | +0.909    | -0.417    | +0.909    | -0.417    | +0.001    | +0.001    |
| 9  | 9e+008   | +0.001    | +0.001    | +0.885    | -0.465    | +0.885    | -0.465    | +0.001    | +0.001    |
| 10 | 1e+009   | +0.001    | +0.002    | +0.859    | -0.512    | +0.859    | -0.512    | +0.001    | +0.002    |
| 11 | 1.1e+009 | +0.001    | +0.002    | +0.830    | -0.558    | +0.830    | -0.558    | +0.001    | +0.002    |
| 12 | 1.2e+009 | +0.001    | +0.002    | +0.799    | -0.602    | +0.799    | -0.602    | +0.001    | +0.002    |
| 13 | 1.3e+009 | +0.002    | +0.002    | +0.765    | -0.644    | +0.765    | -0.644    | +0.002    | +0.002    |

## Plotting Circuit Data

You can plot the analysis results or the data from a file on X-Y plane and polar plane plots, as well as Z, Y, ZY, and YZ Smith charts.

This topic uses a Data File component created from the `samplebjt1.s2p` data file. When you create a Data File component, RF Tool sets the **Frequency** field to the file frequencies and the **Reference impedance** field to the file value. It also populates the **RF Data Display** with data from the file

Analysis

Frequency: 009 5.179e+009 5.44e+009 5.713e+00

Reference impedance: 50

RF Data Display

|    | Freq       | real(S11) | imag(S11) | real(S12) | imag(S12) | real(S21) | imag(S21) | real(S22) | imag(S22) |
|----|------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| 1  | 3.151e+005 | +0.004    | +0.025    | +0.996    | -0.025    | +0.996    | -0.025    | +0.004    | +0.025    |
| 2  | 3.309e+005 | +0.004    | +0.026    | +0.996    | -0.026    | +0.996    | -0.027    | +0.004    | +0.026    |
| 3  | 3.475e+005 | +0.004    | +0.027    | +0.996    | -0.028    | +0.996    | -0.028    | +0.004    | +0.027    |
| 4  | 3.65e+005  | +0.004    | +0.029    | +0.996    | -0.029    | +0.996    | -0.029    | +0.004    | +0.029    |
| 5  | 3.833e+005 | +0.004    | +0.030    | +0.996    | -0.030    | +0.996    | -0.030    | +0.004    | +0.030    |
| 6  | 4.026e+005 | +0.004    | +0.032    | +0.996    | -0.031    | +0.995    | -0.032    | +0.004    | +0.032    |
| 7  | 4.228e+005 | +0.004    | +0.033    | +0.996    | -0.033    | +0.995    | -0.033    | +0.004    | +0.033    |
| 8  | 4.441e+005 | +0.005    | +0.035    | +0.995    | -0.035    | +0.995    | -0.035    | +0.005    | +0.035    |
| 9  | 4.664e+005 | +0.005    | +0.037    | +0.995    | -0.037    | +0.995    | -0.037    | +0.005    | +0.037    |
| 10 | 4.898e+005 | +0.005    | +0.039    | +0.995    | -0.039    | +0.995    | -0.038    | +0.005    | +0.039    |
| 11 | 5.144e+005 | +0.005    | +0.040    | +0.996    | -0.040    | +0.995    | -0.041    | +0.005    | +0.041    |
| 12 | 5.403e+005 | +0.005    | +0.042    | +0.995    | -0.042    | +0.994    | -0.043    | +0.005    | +0.043    |
| 13 | 5.674e+005 | +0.006    | +0.045    | +0.995    | -0.044    | +0.994    | -0.044    | +0.006    | +0.045    |

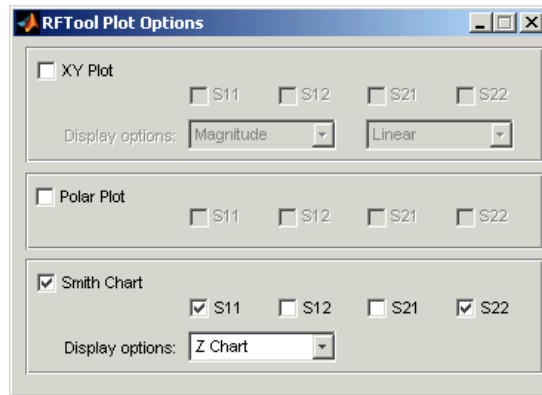
Topics in this section include:

- “Selecting the Plot Type” on page 4-15
- “Generating the Plots” on page 4-16

### Selecting the Plot Type

Click **Plot Options** in the **Analysis** panel to select the plots you want. In the RF Tool Plot Options dialog box, check one or more of the plot types. For each plot type, select the parameters and display options as appropriate.

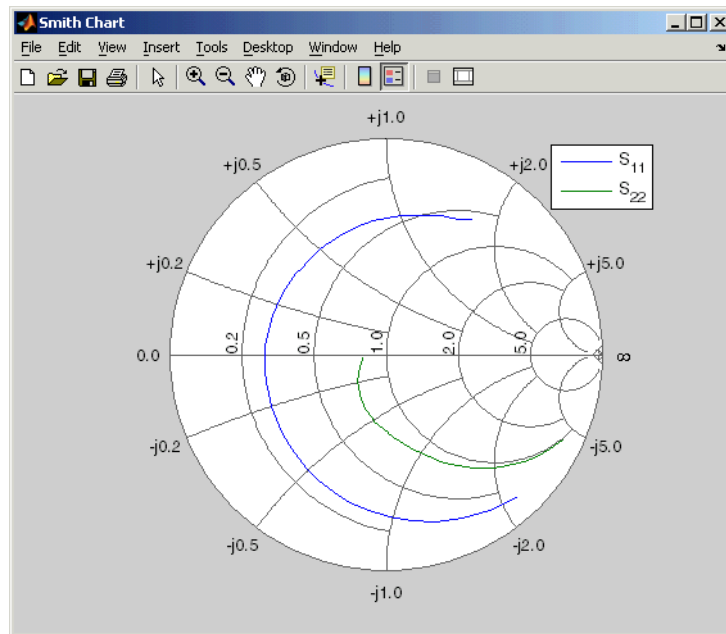
This example plots the `samplebjt1.s2p` data using the default plot options, shown below. These options generate a Z Smith chart of S11 and S22 data.



### Generating the Plots

To generate the selected plots, click **Plot**. RF Tool creates a separate plot for each type of plot you checked: X-Y plot, Polar plot, and Smith chart. Note that for all circuits except Data File, the **Plot** button becomes available only after you perform an analysis.





---

**Note** To display data tips for a plotted line, check **Data Cursor** on the **Tool** menu, and then click the data cursor on the plotted line to see the frequency and the plotted parameter at that point. See “Data Cursor — Displaying Data Values Interactively” in the MATLAB documentation for more information.

---

## Importing RF Objects

RF Tool enables you to import RF objects from your workspace and from files to the top level of your session. You may want to import complex component and network objects that you created in your workspace using RF Toolbox functions. You may also want to import components and networks you exported into your workspace from another RF Tool session.

Once you have imported an object, you can change its name and work with it as you would any other component or network.

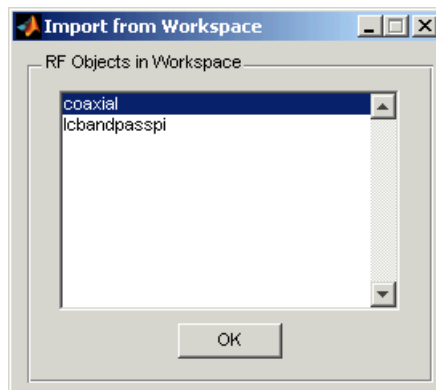
Topics in this section include:

- “Importing from the Workspace” on page 4-18
- “Importing From a File” on page 4-19

### Importing from the Workspace

To import RF circuit objects from the MATLAB workspace into the top level of your session:

- 1** Select **Import From Workspace** from the **File** menu. The **Import from Workspace** dialog box appears. It lists the handles of all RF circuit (rfckt) objects in the workspace.



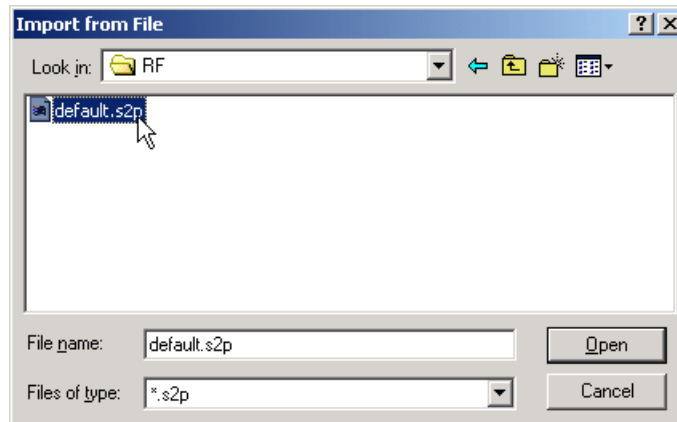
- 2** From the list of RF circuit objects, select the object you want to import, and click **OK**. The object is added to your session with the same name as the

object handle. If there is already a circuit by that name, RF Tool appends a numeral, starting with 1, to the new circuit name.

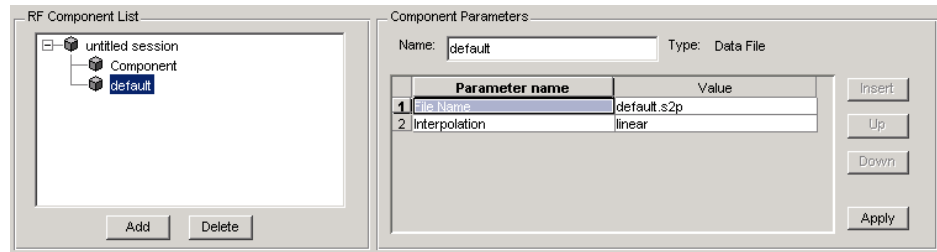
## Importing From a File

You can import RF components from S2P, Y2P, Z2P, and H2P files. To import a component from one of these files:

- 1 Select **Import From File** from the **File** menu. A file browser appears.
- 2 Select the file type you want to import.
- 3 From the list of files in the browser, select the file to import.



- 4 Click **Open**. RF Tool adds the object to your session as a component.



The name of the component is the filename without the extension. If there is already a component by that name, RF Tool appends a numeral, starting with 1, to the new component name. The full filename appears as the value of the component's File Name parameter. If the file is not on the MATLAB path, the value of the File Name parameter also contains the file path.

---

**Note** To import an RF component from an S2P, Y2P, Z2P, or H2P file into a network, insert it in the network as a Data File component. See "Populating an RF Network" on page 4-8 for details.

---

## Exporting RF Objects

RF Tool enables you to export RF components and networks that you have created and refined in RF Tool to your MATLAB workspace or to files. You may want to export circuits and then incorporate them into larger RF systems, or you may want to import them into another session.

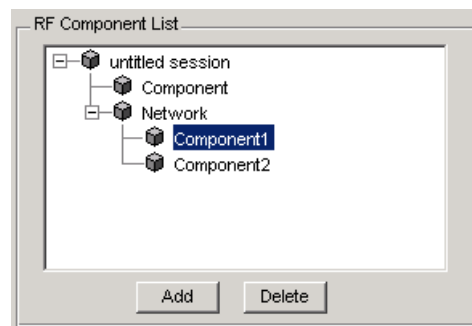
Topics in this section include:

- “Exporting to the Workspace” on page 4-21
- “Exporting to a File” on page 4-22

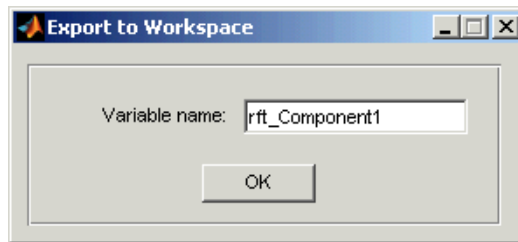
### Exporting to the Workspace

RF Tool enables you to export components and networks to the MATLAB workspace. Once in your workspace, you can use the resulting circuit (rftckt) object as you would any other RF circuit object.

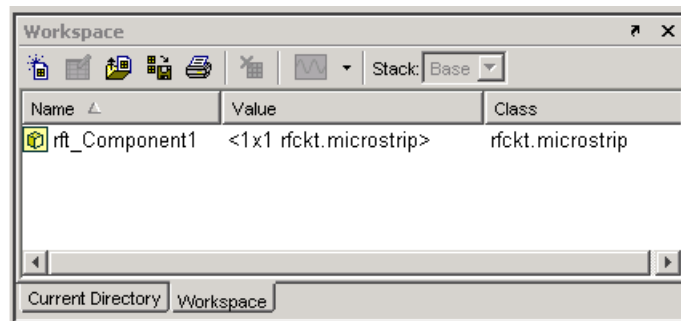
- 1 Select the component or network to export in the **RF Component List** panel of the RF Tool.



- 2 Select **Export to Workspace** from the **File** menu.
- 3 In the **Variable name** field, enter the name you want to give the exported object and click **OK**. The default name is the current name of the component or network prefaced with the string 'rft\_'.



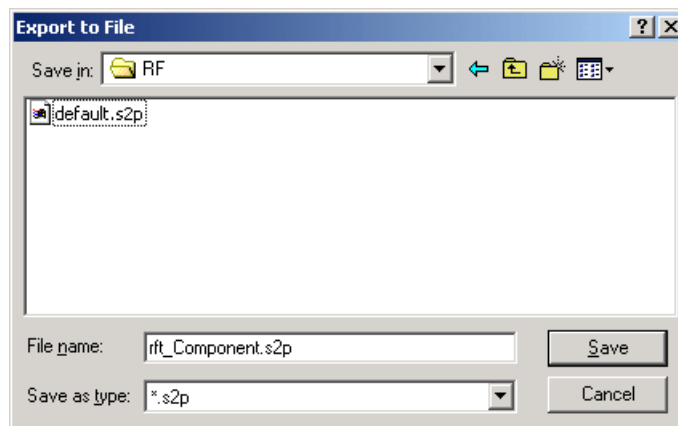
- 4 The component or network becomes accessible in the your workspace via its object handle `rft_Component1`.



### Exporting to a File

RF Tool enables you to export components and networks to files in S2P format. Note that you must have analyzed a component or network in RF Tool before you can export it to a file. See “Analyzing Circuits” on page 4-14 for more information.

- 1 Select **Export To File** from the **File** menu. A file browser appears.



- 2 Browse to the appropriate directory. Enter the name you want to give the file and click **Save**.

The default filename is the current name of the component or network prefaced with the string 'rft\_'. RF Tool also converts any characters that are not alphanumeric to underscores (\_).

## Working with Sessions

The work you do with the RF Tool is organized into sessions. Each session is a collection of independent RF circuits, which can be RF components or RF networks.

Topics in this section include:

- “Saving a Session” on page 4-24
- “Opening an Existing Session” on page 4-25
- “Renaming a Session” on page 4-25
- “Starting a New Session” on page 4-26

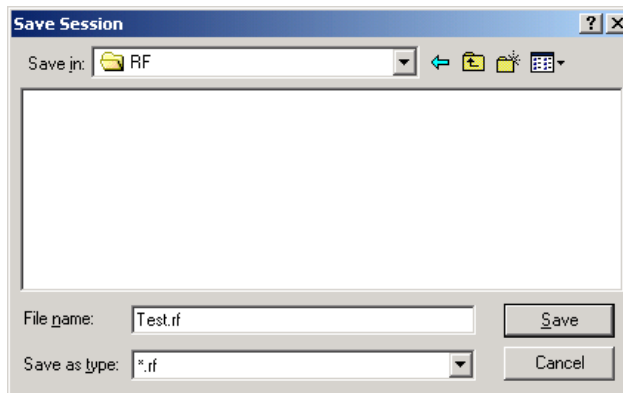
### Saving a Session

To save your session, select **Save Session** or **Save Session As** from the **File** menu. The first time you save a session a browser opens, prompting you for a file name.

---

**Note** The default file name is the session name with any characters that are not alphanumeric converted to underscores (\_). The name of the session itself is unchanged.

---



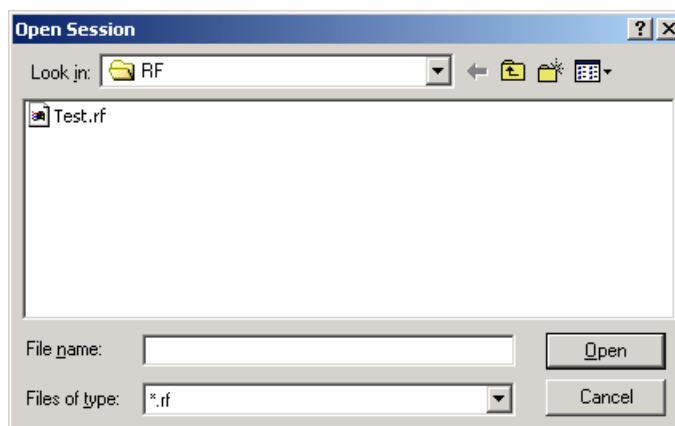


For example, to save your session as `Test.rf` in your current working directory, you would type `Test` in the **File name** field as shown above. RF Tool adds the `.rf` extension automatically to all RF Tool sessions you save.

If the name of your session is `gk's session`, the default file name is `gk_s_session.rf`.

## Opening an Existing Session

You can load an existing session into the RF Tool by selecting **Open Session** from the **File** menu. A browser enables you to select from your previously saved sessions.



Before opening the requested session, RFTool prompts you to save your current session.

## Renaming a Session

To rename a session:

- 1 Select the session in the **RF Component List** panel.
- 2 Type the desired name in the **Name** field of the **Component Parameters** panel.
- 3 Click **Apply**.

### **Starting a New Session**

To start a new session, select **New Session** from the **File** menu. A new session opens in the RF Tool. All its values are set to their defaults.

Before starting a new session, RFTool prompts you to save your current session.

# Function Reference

---

- Functions — Categorical List (p. 5-2) Lists the RF Toolbox functions and objects according to their purpose.
- Functions — Alphabetical List (p. 5-7) Lists the RF Toolbox functions and objects alphabetically.

## Functions – Categorical List

This section lists the RF Toolbox functions and objects according to their purpose.

- “Circuit Objects” on page 5-3
- “Data Objects” on page 5-4
- “Calculations” on page 5-4
- “Data Visualization” on page 5-4
- “Utility Functions” on page 5-4
- “Data I/O” on page 5-5
- “Network Parameter Conversion” on page 5-5
- “Graphical User Interface” on page 5-6

## Circuit Objects

|                                  |                                      |
|----------------------------------|--------------------------------------|
| <code>rfckt</code>               | RF circuit object.                   |
| <code>rfckt.amplifier</code>     | Amplifier, from a data file          |
| <code>rfckt.cascade</code>       | Cascaded network,                    |
| <code>rfckt.coaxial</code>       | Coaxial transmission line            |
| <code>rfckt.cpw</code>           | Coplanar waveguide transmission line |
| <code>rfckt.datafile</code>      | General circuit, from a data file    |
| <code>rfckt.hybrid</code>        | Hybrid connected network             |
| <code>rfckt.lcbandpasspi</code>  | LC bandpass pi network               |
| <code>rfckt.lcbandpasstee</code> | LC bandpass tee network              |
| <code>rfckt.lcbandstoppi</code>  | LC bandstop pi network               |
| <code>rfckt.lcbandstoptee</code> | LC bandstop tee network              |
| <code>rfckt.lchighpasspi</code>  | LC highpass pi network               |
| <code>rfckt.lchighpasstee</code> | LC highpass tee network              |
| <code>rfckt.lclowpasspi</code>   | LC lowpass pi network                |
| <code>rfckt.lclowpasstee</code>  | LC lowpass tee network               |
| <code>rfckt.microstrip</code>    | Microstrip transmission line         |
| <code>rfckt.mixer</code>         | Mixer, from a data file              |
| <code>rfckt.parallel</code>      | Parallel connected network           |
| <code>rfckt.parallelplate</code> | Parallel-plate transmission line     |
| <code>rfckt.series</code>        | Series connected network             |
| <code>rfckt.seriesrlc</code>     | Series RLC network                   |
| <code>rfckt.shuntrlc</code>      | Shunt RLC network                    |
| <code>rfckt.twowire</code>       | Two-wire transmission line           |
| <code>rfckt.txline</code>        | General transmission line            |

## Data Objects

|                          |                                 |
|--------------------------|---------------------------------|
| <code>rfdata</code>      | Data object.                    |
| <code>rfdata.data</code> | Network parameters data object. |

## Calculations

|                             |                                                                                                      |
|-----------------------------|------------------------------------------------------------------------------------------------------|
| <code>analyze</code>        | Calculate network parameters and noise figure for a circuit or data object at specified frequencies. |
| <code>calculate</code>      | Calculate specified network parameters for a circuit or data object.                                 |
| <code>cascadesparams</code> | Calculate cascaded S-parameters.                                                                     |
| <code>deembedsparams</code> | De-embed S-parameters from a cascaded circuit.                                                       |
| <code>gammain</code>        | Calculate GammaIn.                                                                                   |
| <code>gammaout</code>       | Calculate GammaOut.                                                                                  |
| <code>vswr</code>           | Calculate VSWR.                                                                                      |

## Data Visualization

|                         |                                                                             |
|-------------------------|-----------------------------------------------------------------------------|
| <code>plot</code>       | Plot network parameters from a circuit or data object on an X-Y plane.      |
| <code>polar</code>      | Plot network parameters from a circuit or data object on polar coordinates. |
| <code>smith</code>      | Plot network parameters from a circuit or data object on a Smith chart.     |
| <code>smithchart</code> | Plot a complex vector on a Smith chart.                                     |

## Utility Functions

|                      |                                                                                             |
|----------------------|---------------------------------------------------------------------------------------------|
| <code>copy</code>    | Copy a circuit or data object.                                                              |
| <code>extract</code> | Extract specified network parameters from a data object and returns the result in a matrix. |
| <code>getdata</code> | Get data object containing analyzed data.                                                   |

|                         |                                                                                              |
|-------------------------|----------------------------------------------------------------------------------------------|
| <code>listformat</code> | List valid formats for a specified network parameter for a specified circuit or data object. |
| <code>listparam</code>  | List valid network parameters for a specified circuit or data object.                        |

## Data I/O

|                    |                                                                          |
|--------------------|--------------------------------------------------------------------------|
| <code>read</code>  | Read RF network parameters from a file to a new or existing data object. |
| <code>write</code> | Write RF data from a data object to a file.                              |

## Network Parameter Conversion

|                     |                                                                |
|---------------------|----------------------------------------------------------------|
| <code>abcd2h</code> | Convert ABCD-parameters to H-parameters.                       |
| <code>abcd2s</code> | Convert ABCD-parameters to S-parameters.                       |
| <code>abcd2y</code> | Convert ABCD-parameters to Y-parameters.                       |
| <code>abcd2z</code> | Convert ABCD-parameters to Z-parameters.                       |
| <code>h2abcd</code> | Convert H-parameters to ABCD-parameters.                       |
| <code>h2s</code>    | Convert H-parameters to S-parameters.                          |
| <code>h2y</code>    | Convert H-parameters to Y-parameters.                          |
| <code>h2z</code>    | Convert H-parameters to Z-parameters.                          |
| <code>s2abcd</code> | Convert S-parameters to ABCD-parameters.                       |
| <code>s2h</code>    | Convert S-parameters to H-parameters.                          |
| <code>s2s</code>    | Convert S-parameters to S-parameters with different impedance. |
| <code>s2t</code>    | Convert S-parameters to T-parameters.                          |
| <code>s2y</code>    | Convert S-parameters to Y-parameters.                          |
| <code>s2z</code>    | Convert S-parameters to Z-parameters.                          |
| <code>t2s</code>    | Convert T-parameters to S-parameters.                          |
| <code>y2abcd</code> | Convert Y-parameters to ABCD-parameters.                       |

|        |                                          |
|--------|------------------------------------------|
| y2h    | Convert Y-parameters to H-parameters.    |
| y2s    | Convert Y-parameters to S-parameters.    |
| y2z    | Convert Y-parameters to Z-parameters.    |
| z2abcd | Convert Z-parameters to ABCD-parameters. |
| z2h    | Convert Z-parameters to H-parameters.    |
| z2s    | Convert Z-parameters to S-parameters.    |
| z2y    | Convert Z-parameters to Y-parameters.    |

### **Graphical User Interface**

|         |                                                                       |
|---------|-----------------------------------------------------------------------|
| rf tool | Visual interface for creating and analyzing RF circuits and networks. |
|---------|-----------------------------------------------------------------------|



## **Functions — Alphabetical List**

This section contains function reference pages listed alphabetically.

# abcd2h

---

**Purpose** Convert ABCD-parameters to h-parameters

**Syntax** `h_params = abcd2h(abcd_params)`

**Description** `h_params = abcd2h(abcd_params)` converts the ABCD-parameters `abcd_params` into the hybrid parameters `h_params`. The `abcd_params` input is a complex 2-by-2-by-`m` array, representing `m` two-port ABCD-parameters. `h_params` is a complex 2-by-2-by-`m` array, representing `m` two-port h-parameters.

**See Also** `abcd2s`, `abcd2y`, `abcd2z`, `h2abcd`, `s2h`, `y2h`, `z2h`

**Purpose** Convert ABCD-parameters to S-parameters

**Syntax** `s_params = abcd2h(abcd_params, z0)`

**Description** `s_params = abcd2h(abcd_params, z0)` converts the ABCD-parameters `abcd_params` into the scattering parameters `s_params`. The `abcd_params` input is a complex 2-by-2-by-`m` array, representing `m` two-port ABCD-parameters. `z0` is the reference impedance; its default is 50 ohms. `s_params` is a complex 2-by-2-by-`m` array, representing `m` two-port S-parameters.

**See Also** `abcd2h`, `abcd2y`, `abcd2z`, `s2abcd`, `s2h`, `y2h`, `z2h`

# abcd2y

---

**Purpose** Convert ABCD-parameters to Y-parameters

**Syntax** `y_params = abcd2y(abcd_params)`

**Description** `y_params = abcd2y(abcd_params)` converts the ABCD-parameters `abcd_params` into the admittance parameters `y_params`. The `abcd_params` input is a complex 2-by-2-by-*m* array, representing *m* two-port ABCD-parameters. `y_params` is a complex 2-by-2-by-*m* array, representing *m* two-port Y-parameters.

**See Also** `abcd2h`, `abcd2s`, `abcd2z`, `h2y`, `s2y`, `z2y`, `y2abcd`

**Purpose** Convert ABCD-parameters to Z-parameters

**Syntax** `z_params = abcd2z(abcd_params)`

**Description** `z_params = abcd2z(abcd_params)` converts the ABCD-parameters `abcd_params` into the impedance parameters `z_params`. The `abcd_params` input is a complex 2-by-2-by-`m` array, representing `m` two-port ABCD-parameters. `z_params` is a complex 2-by-2-by-`m` array, representing `m` two-port Z-parameters.

**See Also** `abcd2h`, `abcd2s`, `abcd2y`, `h2y`, `s2z`, `y2z`, `z2abcd`

# analyze

---

**Purpose** Calculate the network parameters and noise figure for a circuit or data object

**Syntax** `analyze(h, freq)`

**Description** `analyze(h, freq)` calculates the circuit network parameters and noise figure for the specified frequencies. `h` is the handle of the circuit or data object to be analyzed. `freq` is a vector of frequencies, specified in Hz, at which the circuit is analyzed.

---

**Note** The actual analysis varies for different objects. See the individual `rfckt` and `rfdata` reference pages for specific information.

---

## Circuit Objects

If `h` is a circuit (`rfckt`) object,

- `freq` must be nonnegative for `rfckt.amplifier`, `rfckt.datafile`, and `rfckt.mixer` objects. For all other circuit objects it must be strictly positive.
- `analyze` stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

---

**Note** See the `rfckt` reference page for a list of functions that act on `rfckt` objects.

---

## Data Objects

If `h` is a data object (`rfdata`),

- `freq` must be nonnegative
- If the parameter data originally came from a file, `analyze` performs its analysis using the original data, then stores the results in `h`. Otherwise, `analyze` uses the current S-parameters in `h` and overwrites them with the results.

---

**Note** See the `rfd` reference page for a list of functions that act on `rfd` objects.

---

## See Also

`getdata`, `rfckt`, `rfd`

# calculate

---

**Purpose** Calculate specified network parameters for a circuit or data object

**Syntax** `[data,params] = calculate(h,'parameter1',..., 'parameterN',  
'format')`

**Description** `[data,params] = calculate(h,'parameter1',..., 'parameterN',  
'format')` calculates the specified network parameters for the object `h` and returns them in the `n`-element cell array `data`. The input `h` is the handle of a circuit or data object. `parameter1`, ..., `parameterN` are the network parameters to be calculated. `format` is the format of the output data. Specify `format` as `'none'` to return the network parameters unchanged.

`params` is an `n`-element cell array containing the names, as strings, of the parameters in `data`.

---

**Note** Before calling `calculate`, you must use the `analyze` function to perform a frequency domain analysis for the circuit or data object.

---

For example, `[data,params] = calculate(h,'S11','S22','dB')` returns the `S11` and `S22` parameters in decibel format for the circuit object `h`.

Use the `listparam` and `listformat` functions to get lists of valid network parameters for a circuit or data object and the valid formats for a particular parameter.

## Examples

Analyze a general transmission line, `trl`, with the default characteristic impedance of 50 ohms, phase velocity of 299792458 meters per second, and line length of 0.01 meters for frequencies of 1.0 GHz to 3.0 GHz. Then calculate `S11` and `S22` parameters in decibels.

```
trl = rfckt.txline;
f = [1e9:1.0e7:3e9];
analyze(trl,f);
[data,params] = calculate(trl,'S11','S22','dB')

data =
 [300x1 double] [300x1 double]
```



```
params =
 'S_{11}' 'S_{22}'
```

The first few elements of `data{1}` look like

```
ans =

-313.0712
-312.5446
-312.8039
-312.8039
-312.8039
-312.8039
-312.2928
-312.8039
...
```

**See Also**

`analyze`, `rfckt`, `rfckt.txline`, `rfdata`

# cascadesparams

---

**Purpose** Calculate the cascaded S-parameters

**Syntax** `s_params = cascadesparams(s1_params, s2_params, ..., sn_params)`

**Description** `s_params = cascadesparams(s1_params, s2_params, ..., sn_params)` calculates the scattering parameters, `s_params`, of the cascaded network.

Each of the input networks must be a two-port network described by a 2-by-2-by-*m* array of its S-parameters. All networks must have the same reference impedance.

`s_params` is a 2-by-2-by-*m* array containing the S-parameters of the resulting cascaded network.

**See Also** `t2s`, `s2t`, `deembedsparams`

**Purpose** Copy a circuit or data object

**Syntax** `h2 = copy(h)`

**Description** `h2 = copy(h)` returns a copy of the circuit or data object `h`.

---

**Note** The syntax `h2 = h` copies only the object handle and does not create a new object.

---

**See Also** `rfckt`, `rfdata`

# deembedsparams

---

**Purpose** De-embed S-parameters from a cascaded network

**Syntax** `s2_params = deembedsparams(s_params, s1_params, s3_params)`

**Description** `s2_params = deembedsparams(s_params, s1_params, s3_params)` derives the `s2_params` from the cascaded S-parameters `s_params`, by removing the effects of `s1_params`, and `s3_params`.

Each of the input networks must be a two-port network described by a 2-by-2-by- $m$  array of S-parameters. All networks must have the same reference impedance. `s_params` must contain the S-parameters of the cascaded network of `s1_params`, `s2_params`, and `s3_params`.

`s2_params` is a 2-by-2-by- $m$  array. It contains the de-embedded S-parameters.

**See Also** `t2s`, `s2t`, `cascadesparams`

**Purpose** Extract network parameters from a data object

**Syntax** `outmatrix = extract(h,outtype)`

**Description** `outmatrix = extract(h,outtype)` extracts the network parameters of type `outtype` from `rfdata.data` object `h` and returns them in `outmatrix`.

`outtype` can be one of these case-insensitive strings 'ABCD\_parameters', 'S\_parameters', 'Y\_parameters', 'Z\_parameters', 'H\_parameters', 'T\_parameters'.

---

**Note** Before calling `extract`, you must use the `analyze` function to perform a frequency domain analysis for the data object.

---

**See Also** `analyze`, `rfckt`, `rfdata`

# gammain

---

**Purpose** Calculates the input reflection coefficient of a two port network

**Syntax** `result = gammain(s_params,z0,z1)`

**Description** `result = gammain(s_params,z0,z1)` calculates the input reflection coefficient of a two port network as

$$\Gamma_{In} = S_{11} + \frac{(S_{12} * S_{21}) * \Gamma_L}{1 - S_{22} * \Gamma_L}$$

where

$$\Gamma_L = \frac{Z_l - Z_0}{Z_l + Z_0}$$

`s_params` is a complex 2-by-2-by-`m` array, representing `m` two-port S-parameters. `z0` is the reference impedance  $Z_0$ ; its default is 50 ohms. `z1` is the load impedance  $Z_l$ ; its default is also 50 ohms. `result` is an `m`-element complex vector.

**See Also** `gammaout`

**Purpose** Calculates the output reflection coefficient of a two port network

**Syntax** `result = gammaout(s_params, z0, zs)`

**Description** `result = gammaout(s_params, z0, zs)` calculates the output reflection coefficient of a two port network as

$$\text{GammaOut} = S_{22} + \frac{(S_{12} * S_{21}) * \text{GammaS}}{1 - S_{11} * \text{GammaS}}$$

where

$$\text{GammaS} = \frac{zs - z0}{zs + z0}$$

`s_params` is a complex 2-by-2-by-`m` array, representing `m` two-port S-parameters. `z0` is the reference impedance; its default is 50 ohms. `zs` is the source impedance; its default is also 50 ohms. `result` is an `m`-element complex vector.

**See Also** `gammain`

# getdata

---

**Purpose** Get data object containing analyzed data

**Syntax** `hd = getdata(h)`

**Description** `hd = getdata(h)` returns a handle `hd` to the `rfdata.data` object containing the analysis data, if any, for circuit (`rfckt`) object `h`. If the circuit object `h` has not been analyzed, i.e., there is no analysis data, `getdata` displays an error message.

---

**Note** For all objects except those of type `rfckt.amplifier`, `rfckt.datafile`, and `rfckt.mixer`, before calling `getdata`, you must use the `analyze` function to perform a frequency domain analysis for the circuit (`rfckt`) object.

When you create an object of type `rfckt.amplifier`, `rfckt.datafile`, or `rfckt.mixer`, by reading data from a file, the RF Toolbox automatically creates an `rfdata.data` object and stores data from the file as properties of the data object. You can use the `getdata` function, without first calling `analyze`, to retrieve the handle of this data object.

---

**See Also** `rfckt`, `rfdata`



**Purpose** Convert h-parameters to ABCD-parameters

**Syntax** `abcd_params = h2abcd(h_params)`

**Description** `abcd_params = h2abcd(h_params)` converts the hybrid parameters `h_params` into the ABCD-parameters `abcd_params`. The `h_params` input is a complex 2-by-2-by-`m` array, representing `m` two-port h-parameters. `abcd_params` is a complex 2-by-2-by-`m` array, representing `m` two-port ABCD-parameters.

**See Also** `abcd2h`, `h2s`, `h2y`, `h2z`, `s2abcd`, `y2abcd`, `z2abcd`

# h2s

---

**Purpose** Convert h-parameters to S-parameters

**Syntax** `s_params = h2s(h_params, z0)`

**Description** `s_params = h2s(h_params, z0)` converts the hybrid parameters `h_params` into the scattering parameters `abcd_params`. The `h_params` input is a complex 2-by-2-by-`m` array, representing `m` two-port h-parameters. `z0` is the reference impedance; its default is 50 ohms. `s_params` is a complex 2-by-2-by-`m` array, representing `m` two-port S-parameters.

**See Also** `abcd2s`, `h2abcd`, `h2y`, `h2z`, `s2h`, `y2s`, `z2s`

**Purpose** Convert h-parameters to Y-parameters

**Syntax** `y_params = h2y(h_params)`

**Description** `y_params = h2y(h_params)` converts the hybrid parameters `h_params` into the admittance parameters `y_params`. The `h_params` input is a complex 2-by-2-by-`m` array, representing `m` two-port h-parameters. `y_params` is a complex 2-by-2-by-`m` array, representing `m` two-port Y-parameters.

**See Also** `abcd2z`, `h2abcd`, `h2s`, `h2y`, `s2z`, `y2z`, `z2h`

# h2z

---

**Purpose** Convert h-parameters to  $Z$ -parameters

**Syntax** `z_params = h2z(h_params)`

**Description** `z_params = h2z(h_params)` converts the hybrid parameters `h_params` into the impedance parameters `z_params`. The `h_params` input is a complex 2-by-2-by- $m$  array, representing  $m$  two-port h-parameters. `z_params` is a complex 2-by-2-by- $m$  array, representing  $m$  two-port  $Z$ -parameters.

**See Also** `abcd2z`, `h2abcd`, `h2s`, `h2y`, `s2z`, `y2z`, `z2h`

**Purpose** List valid formats for a network parameter

**Syntax** `list = listformat(h,'parameter')`

**Description** `list = listformat(h,'parameter')` lists the allowable formats for the specified network parameter. The first listed format is the default format for the specified parameter.

In these lists, 'Abs' and 'Mag' are the same as 'Magnitude (linear)', and 'Angle' is the same as 'Angle (degrees)'.

Use the `listparam` function to get the valid network parameters of a circuit or data object.

---

**Note** Before calling `listformat`, you must use the `analyze` function to perform a frequency domain analysis for the circuit or data object.

---

## Examples

```
trl = rfckt.txline;
f = [1e9:1.0e7:3e9];
analyze(trl,f);
list = listformat(trl,'S11')
```

```
list =
 'dB'
 'Magnitude (decibels)'
 'Abs'
 'Mag'
 'Magnitude (linear)'
 'Angle'
 'Angle (degrees)'
 'Angle (radians)'
 'Real'
 'Imag'
 'Imaginary'
```

**See Also** `listparam`, `rfckt`, `rfdata`

# listparam

---

**Purpose** List valid network parameters for a circuit or data object

**Syntax** `list = listparam(h)`

**Description** `list = listparam(h)` lists the valid network parameters for the specified circuit object `h`.

---

**Note** Before calling `listparam`, you must use the `analyze` function to perform a frequency domain analysis for the circuit object.

---

**Examples**

```
trl = rfckt.txline;
f = [1e9:1.0e7:3e9];
analyze(trl,f);
list = listparam(trl)
```

```
list =
 'S11'
 'S12'
 'S21'
 'S22'
 'GAMMAIn'
 'GAMMAOut'
 'VSWRIn'
 'VSWROut'
 'OIP3'
 'NF'
```

**See Also** `listformat`, `rfckt`, `rfdata`

---

|                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Purpose</b>     | Plot network parameters from a circuit or data object on an X-Y plane                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>Syntax</b>      | <code>lineseries = plot(h,parameter1,...,parametern,format)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Description</b> | <p><code>lineseries = plot(h,parameter1,...,parametern,format)</code> plots the network parameters <code>parameter1,...,parametern</code> from the object <code>h</code> on an X-Y plane. <code>h</code> is the handle of a circuit (<code>rfckt</code>) or data (<code>rfddata</code>) object. <code>format</code> is the format of the data to be plotted, e.g. 'Magnitude (decibels)'.<br/><code>plot</code> returns a column vector of handles to <code>lineseries</code> objects, one handle per line. This is the same as the output returned by the MATLAB <code>plot</code> function.</p> <p>Use the Property Editor (<code>propertyeditor</code>) or the MATLAB <code>set</code> function to change the properties. The reference pages for MATLAB functions such as <code>figure</code>, <code>axes</code>, and <code>text</code> list available properties and provide links to more complete descriptions.</p> <p>Type <code>listparam(h)</code> to get a list of valid network parameters for a circuit or data object <code>h</code>. Type <code>listformat(h,parameter)</code> to see the legitimate formats for a specified parameter.</p> <hr/> <p><b>Note</b> For all circuit and data objects except those that contain data from a data file, you must use the <code>analyze</code> function to perform a frequency domain analysis before calling <code>plot</code>.</p> <hr/> <p>for all circuits except Data File, the <b>Plot</b> button becomes available only after you perform an analysis.</p> <hr/> <p><b>Note</b> Use the MATLAB <code>plot</code> function to plot network parameters that are not part of a circuit (<code>rfckt</code>) or data (<code>rfddata</code>) object, but are specified as vector data.</p> <hr/> |
| <b>See Also</b>    | <code>listformat</code> , <code>listparam</code> , <code>plot</code> , <code>polar</code> , <code>smith</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

# polar

---

**Purpose** Plot network parameters from a circuit object or data object on polar coordinates

**Syntax** `lineseries = polar(h,parameter1,...,parametern,format)`

**Description** `lineseries = polar(h,parameter1,...,parametern,format)` plots the network parameters `parameter1,...,parametern` from the object `h` on polar coordinates. `h` is the handle of a circuit (`rfckt`) or data (`rfddata`) object. `format` is the format of the data to be plotted, e.g. 'Magnitude (decibels)'.  
  
`polar` returns a column vector of handles to `lineseries` objects, one handle per line. This is the same as the output returned by the MATLAB `polar` function.

Use the Property Editor (`propertyeditor`) or the MATLAB `set` function to change the properties. The reference pages for MATLAB functions such as `figure`, `axes`, and `text` list available properties and provide links to more complete descriptions.

Type `listparam(h)` to get a list of valid network parameters for a circuit or data object `h`. Type `listformat(h,parameter)` to see the legitimate formats for a specified parameter.

---

**Note** For all circuit and data objects except those that contain data from a data file, you must use the `analyze` function to perform a frequency domain analysis before calling `polar`.

---

---

**Note** Use the MATLAB `polar` function to plot network parameters that are not part of a circuit (`rfckt`) or data (`rfddata`) object, but are specified as vector data.

---

**See Also** `listformat`, `listparam`, `plot`, `polar`, `smith`



---

|                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Purpose</b>     | Read RF network parameters from a file                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Syntax</b>      | <pre>data = read(h) data = read(h, filename) data = read(rfdata.data, filename)</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| <b>Description</b> | <p><code>h = read(h)</code> prompts you to select a <code>.snp</code>, <code>.ynp</code>, <code>.znp</code>, <code>.hnp</code>, or <code>.amp</code> file, where <code>n</code> is the number of ports. <code>read</code> then updates the <code>rfdata.data</code> object <code>h</code> with data from the file you select. See Chapter 3, “AMP File Format,” for information about the <code>.amp</code> format.</p> <p><code>h = read(h, filename)</code> updates the <code>rfdata.data</code> object <code>h</code> with data from the specified file. <code>filename</code> is a string, representing the filename of a <code>.snp</code>, <code>.ynp</code>, <code>.znp</code>, <code>.hnp</code>, or <code>.amp</code> file. The filename must include the file extension.</p> <p><code>data = read(rfdata.data, filename)</code> creates an <code>rfdata.data</code> object <code>data</code>, reads the network parameters from the specified file, and stores them in <code>data</code>.</p> <hr/> <p><b>Note</b> See the <code>rfdata</code> reference page for a list of functions that act on <code>rfdata.data</code> objects.</p> <hr/> |
| <b>References</b>  | [1] EIA/IBIS Open Forum, “Touchstone File Format Specification,” Rev. 1.1, 2002 ( <a href="http://www.eda.org/pub/ibis/connector/touchstone_spec11.pdf">http://www.eda.org/pub/ibis/connector/touchstone_spec11.pdf</a> ).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>See Also</b>    | <code>rfckt</code> , <code>rfdata</code> , <code>write</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

**Purpose** Construct an RF circuit object

**Syntax** `h = rfckt.component('Property1',value1,...)`

**Description** `h = rfckt.component('Property1',value1,...)` returns a circuit object, *h*, of type *component*. See the individual rfckt component reference pages for information about a specific circuit object and its properties. See Chapter 2, “Working with Objects,” for additional information.

**Objects** The component for an rfckt object specifies the type of RF circuit object. The following table lists the available RF circuit objects.

| <b>rfckt.component</b>           | <b>Description</b>                   |
|----------------------------------|--------------------------------------|
| <code>rfckt.amplifier</code>     | Amplifier described by a data file   |
| <code>rfckt.cascade</code>       | Cascaded network                     |
| <code>rfckt.coaxial</code>       | Coaxial transmission line            |
| <code>rfckt.cpw</code>           | Coplanar waveguide transmission line |
| <code>rfckt.datafile</code>      | Circuit described by a data file     |
| <code>rfckt.hybrid</code>        | Hybrid connected network             |
| <code>rfckt.lcbandpasspi</code>  | LC bandpass pi network               |
| <code>rfckt.lcbandpasstee</code> | LC bandpass tee network              |
| <code>rfckt.lcbandstoppi</code>  | LC bandstop pi network               |
| <code>rfckt.lcbandstoptee</code> | LC bandstop tee network              |
| <code>rfckt.lchighpasspi</code>  | LC highpass pi network               |
| <code>rfckt.lchighpasstee</code> | LC highpass tee network              |
| <code>rfckt.lclowpasspi</code>   | LC lowpass pi network                |
| <code>rfckt.lclowpasstee</code>  | LC lowpass tee network               |
| <code>rfckt.microstrip</code>    | Microstrip transmission line         |

| <b>rfckt.component</b> | <b>Description</b>               |
|------------------------|----------------------------------|
| rfckt.mixer            | Mixer described by a data file   |
| rfckt.parallel         | Parallel connected network       |
| rfckt.parallelplate    | Parallel-plate transmission line |
| rfckt.series           | Series connected network         |
| rfckt.seriesrlc        | Series RLC network               |
| rfckt.shuntrlc         | Shunt RLC network                |
| rfckt.twowire          | Two-wire transmission line       |
| rfckt.txline           | General transmission line        |

## Functions

The following functions act on rfckt objects.

| <b>Function</b> | <b>Purpose</b>                                                             |
|-----------------|----------------------------------------------------------------------------|
| analyze         | Analyze a circuit or data object in the frequency domain.                  |
| calculate       | Calculate specified network parameters for a circuit or data object        |
| copy            | Copy a circuit or data object                                              |
| getdata         | Get data object containing analyzed data                                   |
| listformat      | List valid formats for a network parameter                                 |
| listparam       | List valid network parameter for a circuit or data object                  |
| plot            | Plot network parameters from a circuit or data object on an X-Y plane      |
| polar           | Plot network parameters from a circuit or data object on polar coordinates |
| smith           | Plot network parameters from a circuit or data object on a Smith chart     |

## Properties

Properties vary for each type of component. See the individual component reference pages for information about properties.

### Viewing Object Properties

You can use `get` to view an `rfckt` object's properties. To see a specific property of an object `h`, use

```
get(h, 'PropertyName')
```

To see all properties for an object `h`, use

```
get(h)
```

### Changing Object Properties

To see the properties of an object `h` whose values you can change use

```
set(h)
```

To change specific properties of object `h`, use

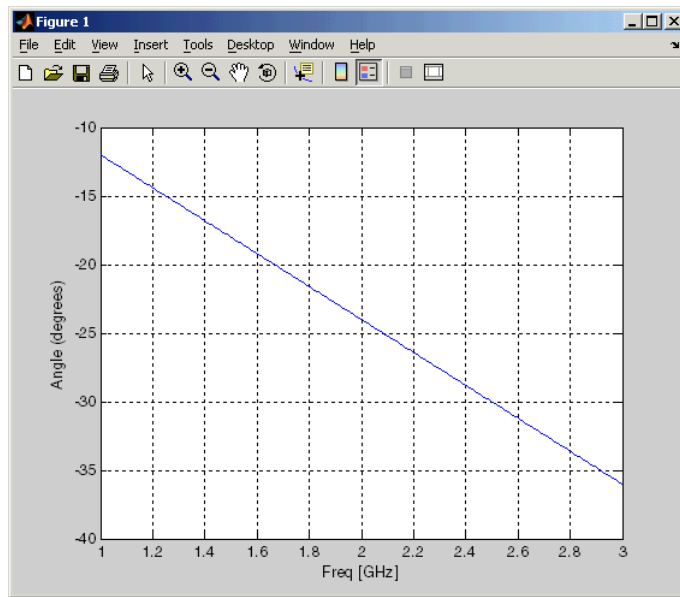
```
set(h, 'PropertyName1', value1, 'PropertyName2', value2, ...)
```

Note that you must use single quotation marks around the property name.

## Examples

Construct a general transmission line, `tr1`, with the default characteristic impedance of 50 ohms, phase velocity of 299792458 meters per second, and line length of 0.01 meters. Then perform frequency domain analysis from 1.0 GHz to 3.0 GHz. Plot the resulting S21 network parameters, using the 'angle' format, on the X-Y plane.

```
tr1 = rfckt.txline;
f = [1e9:1.0e7:3e9]; % Simulation frequencies
analyze(tr1,f); % Do frequency domain analysis
figure
plot(tr1,'s21','angle'); % Plot magnitude of S21 in XY plane
```



You can also use other RF Toolbox functions such as `polar` and `smith` to visualize results.

### See Also

`rfdata`

`analyze`, `calculate`, `copy`, `getdata`, `listformat`, `listparam`, `plot`, `polar`, `rfdata`, `smith`

# rfckt.amplifier

---

**Purpose** Construct an amplifier object from a data file

**Syntax** `h = rfckt.amplifier('Property1',value1,'Property2',value2,...)`  
`h = rfckt.amplifier`

**Description** `h = rfckt.amplifier('Property1',value1,'Property2',value2,...)` returns an `rfckt.amplifier` circuit object, `h`, based on the specified properties. Use the 'File' property to specify a source `.amp`, `.s2p`, `.y2p`, or `.z2p` file that describes a nonlinear amplifier. Properties you do not specify retain their default values. See Chapter 3, “AMP File Format,” for information about the `.amp` format.

`h = rfckt.amplifier` returns an amplifier circuit object whose properties all have their default values.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `rfckt.amplifier` circuit object, use the `analyze` function to calculate the network parameters, output power intercept point, and noise figure at specified frequencies. For `rfckt.amplifier` objects, `freq` must be nonnegative.

```
analyze(h, freq)
```

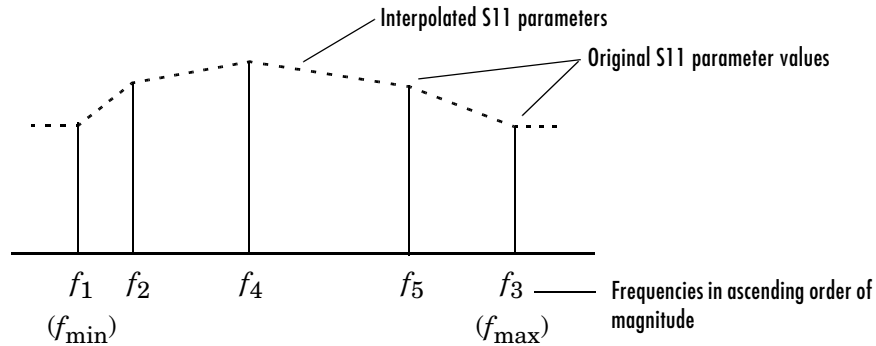
The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

## Network Parameters

If the file you specify with the 'File' property contains network Y- or Z-parameters, `analyze` first converts these parameters, as they exist in the `rfckt.amplifier` object, to S-parameters. Using the interpolation method you specify with the 'IntpType' property, `analyze` interpolates the S-parameters to determine the S-parameters at the specified frequencies.

Specifically, analyze orders the S-parameters according to the ascending order of their frequencies,  $f_n$ . It then interpolates the S-parameters, using the MATLAB interp1 function. For example, the curve in the following diagram illustrates the result of interpolating the S11 parameters at five different frequencies.



You can specify the interpolation method as cubic, linear (default), or spline. For more information, see “One-Dimensional Interpolation” and the interp1 reference page in the MATLAB documentation.

As shown in the diagram above, analyze uses the parameter values at  $f_{\min}$ , the minimum input frequency, for all simulation frequencies smaller than  $f_{\min}$ . It uses the parameters values at  $f_{\max}$ , the maximum input frequency, for all simulation frequencies greater than  $f_{\max}$ . In both cases, the results may not be accurate.

### OIP3

If your file contains power data, the analyze function uses it to calculate OIP3. If your file contains no power data, set the value of the 'OIP3' property to an appropriate value.

### Noise Figure

If active spot noise data exists in the file you specify with the 'File' property, the analyze function uses the data to calculate the noise figure. It first interpolates the noise data for the simulation frequencies, using the interpolation method specified by the 'IntpType' property. It then calculates the noise figure using the resulting values.

# rfckt.amplifier

If the file you specify contains no noise data, set the value of the 'NF' property to an appropriate value.

## Properties

This table lists properties useful to `rfckt.amplifier` objects along with units, valid values, and property descriptions.

| Property | Description                                                                                                                                                              | Units, Values                                       |
|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------|
| File     | .amp, .s2p, .y2p, or .z2p file describing a nonlinear amplifier                                                                                                          | Default is 'default.amp'.                           |
| IntpType | Interpolation method                                                                                                                                                     | 'linear' (default), 'spline', or 'cubic'            |
| Name     | Object name (read only)                                                                                                                                                  | String. 'Amplifier'                                 |
| NF       | Noise figure. The amount of noise relative to a noise temperature of 290 degrees Kelvin. 0 dB indicates a noiseless system                                               | Decibels. Default is 0.                             |
| nPort    | Number of ports (read only)                                                                                                                                              | Integer. The value is always 2.                     |
| OIP3     | Third order output intercept point of the amplifier                                                                                                                      | dBm. Default is Inf.                                |
| RFdata   | Handle of an <code>rfdata.data</code> object that describes a nonlinear amplifier. The <code>analyze</code> function creates this object and sets the value of 'RFdata'. | Handle. Default is [1x1 <code>rfdata.data</code> ]. |

## References

[1] EIA/IBIS Open Forum, "Touchstone File Format Specification," Rev. 1.1, 2002 ([http://www.eda.org/pub/ibis/connector/touchstone\\_spec11.pdf](http://www.eda.org/pub/ibis/connector/touchstone_spec11.pdf)).

## See Also

`analyze`, `rfckt`, `rfckt.cascade`, `rfckt.datafile`, `rfckt.hybrid`, `rfckt.mixer`, `rfckt.parallel`, `rfckt.series`, `rfckt.seriesrlc`, `rfckt.shuntrlc`



---

|                    |                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Purpose</b>     | Construct a cascaded network object                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>Syntax</b>      | <pre>h = rfckt.cascade('Property1',value1,'Property2',value2,...) h = rfckt.cascade</pre>                                                                                                                                                                                                                                                                                                                                                   |
| <b>Description</b> | <p><code>h = rfckt.cascade('Property1',value1,'Property2',value2,...)</code> returns a cascaded network object, <code>h</code>, based on the specified properties. Use the 'Ckts' property to specify the <code>rfckt</code> objects to be cascaded. Properties you do not specify retain their default values.</p> <p><code>h = rfckt.cascade</code> returns a cascaded network object whose properties all have their default values.</p> |

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the cascade network object, use the `analyze` function calculate the network parameters and noise figure at specified frequencies. For `rfckt.cascade` objects, `freq` must be strictly positive.

```
analyze(h,freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

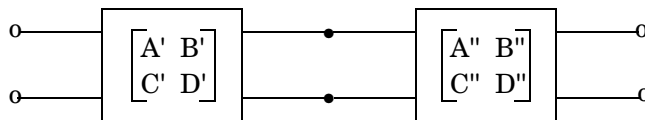
```
hd = getdata(h);
```

### Network Parameters

The `analyze` function first calculates the ABCD parameters of the cascaded network. It starts by converting each component network's parameters to an ABCD parameters matrix. The figure shows a cascaded network consisting of two 2-port networks, each represented by its ABCD matrix.

The `analyze` function then calculates the ABCD parameter matrix for the cascaded network by calculating the product of the ABCD matrices of the individual networks.

The figure shows a cascaded network consisting of two 2-port networks, each represented by its ABCD-parameters.



The following equation illustrates calculations of the ABCD-parameters for two 2-port networks.

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} A' & B' \\ C' & D' \end{bmatrix} \begin{bmatrix} A'' & B'' \\ C'' & D'' \end{bmatrix}$$

Finally, analyze converts the ABCD parameters of the cascaded network to S-parameters at the frequencies specified in the analyze input argument freq.

### OIP3

The analyze function calculates the output power at the third-order intercept point (OIP3) for an N-element cascade using the following equation

$$OIP_3 = \frac{1}{\frac{1}{OIP_{3,N}} + \frac{1}{(G_N \cdot OIP_{3,N-1})} + \dots + \frac{1}{(G_N \cdot G_{N-1} \cdot \dots \cdot G_2 \cdot OIP_{3,1})}}$$

where  $G_n$  is the gain of the  $n$ th element of the cascade and  $OIP_{3,n}$  is the OIP3 of the  $n$ th element.

### Noise Figure

The analyze function calculates the noise figure for an N-element cascade using the following equation

$$NF = NF_1 + \frac{NF_2 - 1}{G_1} + \frac{NF_3 - 1}{G_1 \cdot G_2} + \dots + \frac{NF_N - 1}{G_1 \cdot G_2 \cdot \dots \cdot G_{N-1}}$$

where  $G_n$  is the gain of the  $n$ th element of the cascade and  $NF_n$  is the noise figure of the  $n$ th element.

**Properties**

This table lists properties useful to `rfckt.cascade` objects along with units, valid values, and property descriptions.

| Property | Description                                                                                                                                              | Units, Values                                         |
|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| Ckts     | Cell array containing all circuit objects in the network, in order from source to load. All circuits must be 2-port                                      | Handles to <code>rfckt</code> objects. Default is {}. |
| Name     | Object name (read only)                                                                                                                                  | String. 'Cascaded Network'                            |
| nPort    | Number of ports (read only)                                                                                                                              | Integer. The value is always 2.                       |
| RFdata   | <code>rfdata.data</code> object that describes a cascaded network. The <code>analyze</code> function creates this object and sets the value of 'RFdata'. | Handle. Default is [].                                |

**References**

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

**See Also**

`analyze`, `rfckt`, `rfckt.amplifier`, `rfckt.datafile`, `rfckt.hybrid`, `rfckt.parallel`, `rfckt.series`, `rfckt.seriesrlc`, `rfckt.shuntrlc`

# rfckt.coaxial

---

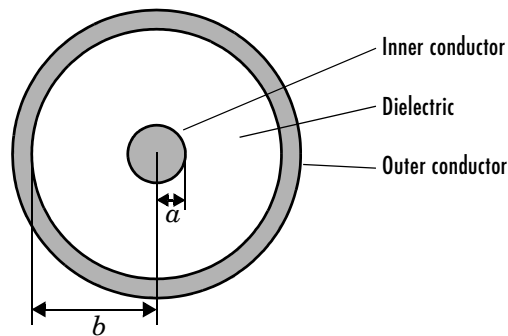
**Purpose** Construct a coaxial transmission line object

**Syntax** `h = rfckt.coaxial('Property1',value1,'Property2',value2,...)`  
`h = rfckt.coaxial`

**Description** `h = rfckt.coaxial('Property1',value1,'Property2',value2,...)` returns a coaxial transmission line object, `h`, with the specified properties. Properties you do not specify retain their default values.

`h = rfckt.coaxial` returns a coaxial transmission line object whose properties all have their default values.

A coaxial transmission line is shown here in cross-section. Its physical characteristics include the radius of the inner conductor of the coaxial transmission line  $a$ , and the radius of the outer conductor  $b$ .



---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the coaxial circuit object, use the `analyze` function to calculate the network parameters and noise figure at specified frequencies. For `rfckt.coaxial` objects, `freq` must be strictly positive.

`analyze(h, freq)`

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

### Network Parameters

A coaxial transmission line object enables you to model the transmission line as a stub or as a stubless line.

**Stubless Transmission Line.** If you model the transmission line as a stubless line, the `analyze` function calculates the S-parameters for the specified frequencies, based on the physical length of the transmission line,  $D$ , and the complex propagation constant,  $k$ .

$$S_{11} = 0$$

$$S_{12} = e^{-kD}$$

$$S_{21} = e^{-kD}$$

$$S_{22} = 0$$

$k$  is a vector whose elements correspond to the elements of the input vector `freq`.  $k$  can be expressed in terms of the resistance ( $R$ ), inductance ( $L$ ), conductance ( $G$ ), and capacitance ( $C$ ) per unit length (meters) as

$$k = k_r + jk_i = \sqrt{(R + j2\pi fL)(G + j2\pi fC)}$$

where  $f$  is the frequency range specified in the `analyze` input argument `freq`, and

$$R = \frac{1}{2\pi\sigma_{\text{cond}}\delta} \left( \frac{1}{a} + \frac{1}{b} \right)$$

$$L = \frac{\mu}{2\pi} \ln(b/a)$$

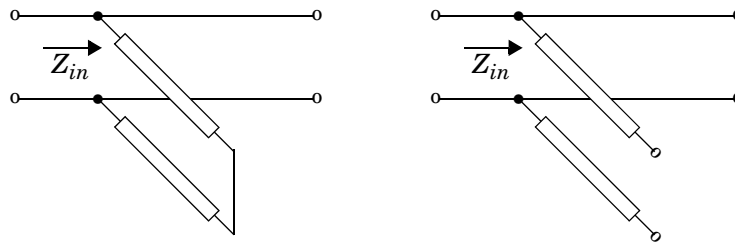
$$G = \frac{2\pi\sigma_{\text{diel}}}{\ln(b/a)}$$

$$C = \frac{2\pi\epsilon}{\ln(b/a)}$$

In these equations,  $\sigma_{\text{cond}}$  is the conductivity in the conductor and  $\sigma_{\text{diel}}$  is the conductivity in the dielectric.  $\mu$  is the relative permeability of the dielectric,  $\epsilon$  is its permittivity as derived from the EpsilonR property, and skin depth  $\delta$  is calculated as  $1/\sqrt{\pi f \mu \sigma_{\text{cond}}}$ .

**Shunt and Series Stubs.** If you model the transmission line as a shunt or series stub, the analyze function first calculates the ABCD-parameters at the specified frequencies. It then uses the abcd2s function to convert the ABCD-parameters to S-parameters.

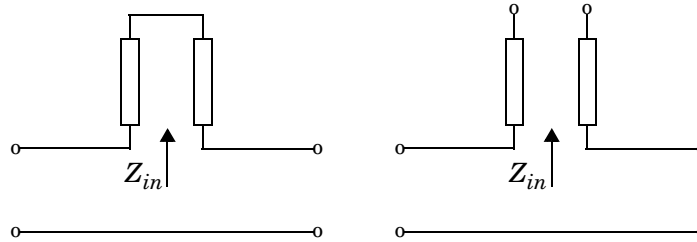
When you set the StubMode property to 'Shunt', the 2-port network consists of a stub transmission line that you can terminate with either a short circuit or an open circuit as shown here.



$Z_{in}$  is the input impedance of the shunt circuit. The ABCD-parameters for the shunt stub are calculated as

$$\begin{aligned} A &= 1 \\ B &= 0 \\ C &= 1/Z_{in} \\ D &= 1 \end{aligned}$$

When you set the StubMode property to 'Series', the 2-port network consists of a series transmission line that you can terminate with either a short circuit or an open circuit as shown here.



$Z_{in}$  is the input impedance of the series circuit. The ABCD-parameters for the series stub are calculated as

$$\begin{aligned}
 A &= 1 \\
 B &= Z_{in} \\
 C &= 0 \\
 D &= 1
 \end{aligned}$$

## Properties

This table lists properties useful to `rfckt.coaxial` objects along with units, valid values, and property descriptions.

| Property     | Description                                                                                                                                     | Units, Values                       |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------|
| EpsilonR     | Relative permittivity of the dielectric expressed as the ratio of the permittivity of the dielectric to permittivity in free space $\epsilon_0$ | Default is 2.3.                     |
| Inner Radius | Radius of the inner conductor                                                                                                                   | Meters. Default is 7.25e-4.         |
| LineLength   | Physical length of the transmission line                                                                                                        | Meters. Default is 0.01.            |
| Loss         | Reduction in strength of the signal as it travels over the transmission line. Read-only; set by the <code>analyze</code> function.              | Decibels per meter. Default is [ ]. |

| Property     | Description                                                                                                                                     | Units, Values                                  |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------|
| MuR          | Relative permeability of the dielectric expressed as the ratio of the permeability of the dielectric to permeability in free space $\mu_0$      | Default is 1.                                  |
| Name         | Object name (read only)                                                                                                                         | String. 'Coaxial Transmission Line'            |
| nPort        | Number of ports (read only)                                                                                                                     | Integer. The value is always 2.                |
| Outer Radius | Radius of the outer conductor                                                                                                                   | Meters. Default is 0.0026.                     |
| PV           | Phase velocity. Propagation velocity of a uniform plane wave on the transmission line. Read-only; set by the analyze function.                  | Meters per second. Default is [ ].             |
| RFdata       | rfdata.data object describing the specified coaxial transmission line. The analyze function creates this object and sets the value of 'RFdata'. | Handle. Default is [ ].                        |
| SigmaCond    | Conductivity in the conductor                                                                                                                   | Siemens per meter (S/m). Default is Inf.       |
| SigmaDie1    | Conductivity in the dielectric                                                                                                                  | Siemens per meter (S/m). Default is 0.         |
| StubMode     | Type of stub.                                                                                                                                   | String. 'None' (default), 'Series', or 'Shunt' |



| Property    | Description                                                             | Units, Values                                                                                 |
|-------------|-------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------|
| Termination | Stub termination for stub models<br>Shunt and Series                    | String.<br>'None' (default),<br>'Open', or 'Short'.<br>Use 'None' when<br>StubMode is 'None'. |
| Z0          | Characteristic impedance.<br>Read-only; set by the analyze<br>function. | Ohms. Default is [ ].                                                                         |

## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

## See Also

analyze, rfckt, rfckt.cpw, rfckt.microstrip, rfckt.parallelplate,  
rfckt.twowire, rfckt.txline

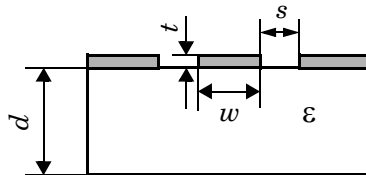
**Purpose** Construct a coplanar waveguide transmission line object

**Syntax**  
`h = rfckt.cpw('Property1',value1,'Property2',value2,...)`  
`h = rfckt.cpw`

**Description** `h = rfckt.cpw('Property1',value1,'Property2',value2,...)` returns a coplanar waveguide transmission line object, `h`, with the specified properties. Properties you do not specify retain their default values.

`h = rfckt.cpw` returns a coplanar waveguide transmission line object whose properties all have their default values.

A coplanar waveguide transmission line is shown here in cross-section. Its physical characteristics include the conductor width ( $w$ ), the conductor thickness ( $t$ ), the slot width ( $s$ ), the substrate height ( $d$ ), and the permittivity constant ( $\epsilon$ ).



---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `rfckt.cpw` circuit object, use the `analyze` function to calculate the network parameters and noise figure at specified frequencies. For `rfckt.cpw` objects, `freq` must be strictly positive.

```
analyze(h,freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

## Network Parameters

A coplanar waveguide transmission line object enables you to model the transmission line as a stub or as a stubless line.

**Stubless Transmission Line.** If you model the transmission line as a stubless line, the analyze function calculates the S-parameters for the specified frequencies, based on the physical length of the transmission line,  $D$ , and the complex propagation constant,  $k$ .

$$S_{11} = 0$$

$$S_{12} = e^{-kD}$$

$$S_{21} = e^{-kD}$$

$$S_{22} = 0$$

$k = \alpha_a + i\beta$ , where  $\alpha_a$  is the attenuation coefficient and  $\beta$  is the wave number. The attenuation coefficient  $\alpha_a$  is related to the loss,  $\alpha$ , by

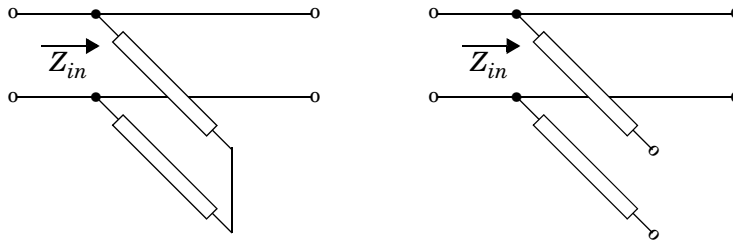
$$\alpha_a = -\ln 10^{-\frac{\alpha}{20}}$$

where  $\alpha$  is the reduction in signal strength, in dB, per unit length.  $\alpha$  combines both conductor loss and dielectric loss and is derived from the rfckt.cpw object properties.

The analyze function normalizes the S-parameters to 50 ohms. This is the default reference impedance of the rfdata.data object that the analyze function creates.

**Shunt and Series Stubs.** If you model the transmission line as a shunt or series stub, the analyze function first calculates the ABCD-parameters at the specified frequencies. It then uses the abcd2s function to convert the ABCD-parameters to S-parameters.

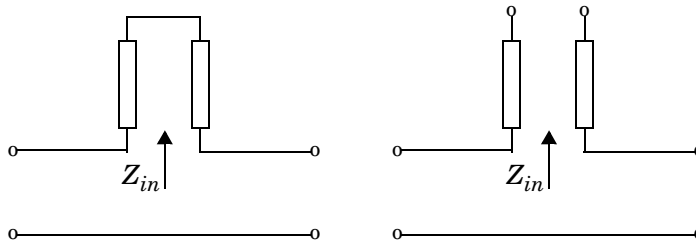
When you set the StubMode property to 'Shunt', the 2-port network consists of a stub transmission line that you can terminate with either a short circuit or an open circuit as shown here.



$Z_{in}$  is the input impedance of the shunt circuit. The ABCD-parameters for the shunt stub are calculated as

$$\begin{aligned} A &= 1 \\ B &= 0 \\ C &= 1/Z_{in} \\ D &= 1 \end{aligned}$$

When you set the StubMode property to 'Series', the 2-port network consists of a series transmission line that you can terminate with either a short circuit or an open circuit as shown here.



$Z_{in}$  is the input impedance of the series circuit. The ABCD-parameters for the series stub are calculated as

$$\begin{aligned} A &= 1 \\ B &= Z_{in} \\ C &= 0 \\ D &= 1 \end{aligned}$$

**Properties**

This table lists properties useful to `rfckt.cpw` objects along with units, valid values, and property descriptions.

| <b>Property</b> | <b>Description</b>                                                                                                                              | <b>Units, Values</b>                           |
|-----------------|-------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------|
| ConductorWidth  | Physical width of the conductor.                                                                                                                | Meters. Default is $0.6e-4$ .                  |
| EpsilonR        | Relative permittivity of the dielectric expressed as the ratio of the permittivity of the dielectric to permittivity in free space $\epsilon_0$ | Default is 9.8.                                |
| Height          | Thickness of the dielectric on which the conductor resides.                                                                                     | Meters. Default is $0.635e-4$ .                |
| LineLength      | Physical length of the transmission line                                                                                                        | Meters. Default is 0.01.                       |
| Loss            | Reduction in strength of the signal as it travels over the transmission line. Read-only; set by the analyze function.                           | Decibels per meter. Default is [ ].            |
| LossTangent     | Loss angle tangent of the dielectric                                                                                                            | Default is 0.                                  |
| Name            | Object name (read only)                                                                                                                         | String. 'Coplanar Waveguide Transmission Line' |
| nPort           | Number of ports (read only)                                                                                                                     | Integer. The value is always 2.                |
| PV              | Phase velocity. Propagation velocity of a uniform plane wave on the transmission line. Read-only; set by the analyze function.                  | Meters per second. Default is [ ].             |

| Property    | Description                                                                                                                                                | Units, Values                                                                                 |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------|
| RFdata      | rfdata.data object describing the specified coplanar waveguide transmission line. The analyze function creates this object and sets the value of 'RFdata'. | Handle. Default is [].                                                                        |
| SigmaCond   | Conductivity in the conductor                                                                                                                              | Siemens per meter (S/m). Default is Inf.                                                      |
| SlotWidth   | Physical width of the slot                                                                                                                                 | Meters. Default is 0.2e-4.                                                                    |
| StubMode    | Type of stub                                                                                                                                               | String.<br>'None' (default),<br>'Series', or<br>'Shunt'                                       |
| Termination | Termination for stub modes 'Shunt' and 'Series'.                                                                                                           | String.<br>'None' (default),<br>'Open', or 'Short'.<br>Use 'None' when<br>StubMode is 'None'. |
| Thickness   | Physical thickness of the conductor.                                                                                                                       | Meters. Default is 0.005e-6.                                                                  |
| Z0          | Characteristic impedance. Read-only; set by the analyze function.                                                                                          | Ohms. Default is [].                                                                          |

## References

[1] Gupta, K. C., Ramesh Garg, Inder Bahl, and Prakash Bhartia, *Microstrip Lines and Slotlines*, 2nd Edition, Artech House, Inc., Norwood, MA, 1996.

## See Also

analyze, rfckt, rfckt.coaxial, rfckt.microstrip, rfckt.parallelplate, rfckt.twowire, rfckt.txline

**Purpose**

Construct a circuit object from a data file

```
h = rfckt.datafile('Property1',value1,'Property2',value2,...)
h = rfckt.datafile
```

**Description**

`h = rfckt.datafile('Property1',value1,'Property2',value2,...)` returns a circuit object, `h`, based on the specified properties. Use the 'File' property to specify a source `.snp`, `.ynp`, `.znp`, `.hnp`, or `.amp` file that describes an  $n$ -port circuit. Properties you do not specify retain their default values. See Chapter 3, “AMP File Format,” for information about the `.amp` format.

`h = rfckt.datafile` returns a circuit object whose properties all have their default values.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis**

After you create the `datafile` circuit object, use the `analyze` function to calculate the network parameters and noise figure at specified frequencies. For `rfckt.datafile` objects, `freq` must be nonnegative.

```
analyze(h, freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

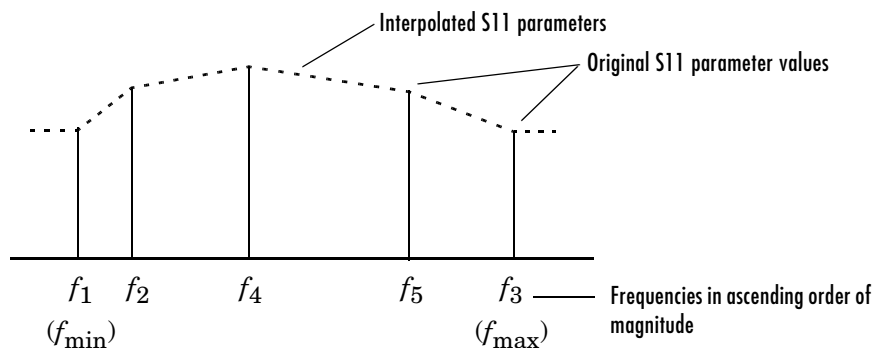
```
hd = getdata(h);
```

**Network Parameters**

If the file you specify with the 'File' property contains network Y- or Z-parameters, `analyze` first converts these parameters, as they exist in the `rfckt.datafile` object, to S-parameters. Using the interpolation method you specify with the 'IntpType' property, `analyze` interpolates the S-parameters to determine the S-parameters at the specified frequencies.

Specifically, `analyze` orders the S-parameters according to the ascending order of their frequencies,  $f_n$ . It then interpolates the S-parameters, using the MATLAB `interp1` function. For example, the curve in the following diagram

illustrates the result of interpolating the S11 parameters at five different frequencies.



You can specify the interpolation method as `cubic`, `linear` (default), or `spline`. For more information, see “One-Dimensional Interpolation” and the `interp1` reference page in the MATLAB documentation.

As shown in the diagram above, `analyze` uses the parameter values at  $f_{\min}$ , the minimum input frequency, for all simulation frequencies smaller than  $f_{\min}$ . It uses the parameters values at  $f_{\max}$ , the maximum input frequency, for all simulation frequencies greater than  $f_{\max}$ . In both cases, the results may not be accurate.

## Properties

This table lists properties useful to `rfckt.datafile` objects along with units, valid values, and property descriptions.

| Property | Description                                                                         | Units, Values                            |
|----------|-------------------------------------------------------------------------------------|------------------------------------------|
| File     | .snp, .ynp, .znp, or .hnp file describing a circuit, where n is the number of ports | String. Default is 'passive.s2p'.        |
| IntpType | Interpolation method                                                                | 'linear' (default), 'spline', or 'cubic' |
| Name     | Object name (read only)                                                             | String. 'Data File'                      |



| Property | Description                                                                                                               | Units, Values                         |
|----------|---------------------------------------------------------------------------------------------------------------------------|---------------------------------------|
| nPort    | Number of ports.                                                                                                          | Integer. Default is 2.                |
| RFdata   | rfdata.data object describing a passive circuit. The analyze function creates this object and sets the value of 'RFdata'. | Handle. Default is [1x1 rfdata.data]. |

## References

[1] EIA/IBIS Open Forum, “Touchstone File Format Specification,” Rev. 1.1, 2002 ([http://www.eda.org/pub/ibis/connector/touchstone\\_spec11.pdf](http://www.eda.org/pub/ibis/connector/touchstone_spec11.pdf)).

## See Also

analyze, rfckt, rfckt.amplifier, rfckt.cascade, rfckt.hybrid, rfckt.parallel, rfckt.series, rfckt.seriesrlc, rfckt.shuntrlc

# rfckt.hybrid

---

**Purpose** Construct a hybrid connected network object

**Syntax**  
`h = rfckt.hybrid('Property1',value1,'Property2',value2,...)`  
`h = rfckt.hybrid`

**Description** `h = rfckt.hybrid('Property1',value1,'Property2',value2,...)` returns a hybrid connected network object, `h`, based on the specified properties. Use the 'Ckts' property to specify the `rfckt` objects to be connected. Properties you do not specify retain their default values.

`h = rfckt.hybrid` returns a hybrid connected network object whose properties all have their default values.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the hybrid network object, use the `analyze` function to calculate the network parameters and noise figure at specified frequencies. For `rfckt.hybrid` objects, `freq` must be strictly positive.

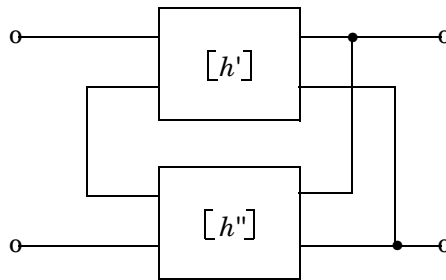
```
analyze(h, freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain from the circuit object's 'RFdata' property with the statement

```
hd = get(h, 'RFdata');
```

## Network Parameters

The `analyze` function first calculates the  $h$  matrix of the hybrid network. It starts by converting each component network's parameters to an  $h$  matrix. The figure shows a hybrid connected network consisting of two 2-port networks, each represented by its  $h$  matrix.



where  $[h'] = \begin{bmatrix} h_{11}' & h_{12}' \\ h_{21}' & h_{22}' \end{bmatrix}$  and  $[h''] = \begin{bmatrix} h_{11}'' & h_{12}'' \\ h_{21}'' & h_{22}'' \end{bmatrix}$

The analyze function then calculates the  $h$  matrix for the hybrid network by calculating the sum of the  $h$  matrices of the individual networks. The following equation illustrates the calculations for two 2-port networks.

$$[h] = \begin{bmatrix} h_{11}' + h_{11}'' & h_{12}' + h_{12}'' \\ h_{21}' + h_{21}'' & h_{22}' + h_{22}'' \end{bmatrix}$$

Finally, analyze converts the  $h$  matrix of the hybrid network to S-parameters at the frequencies specified in the analyze input argument freq.

## Properties

This table lists properties useful to rfckt.hybrid objects along with units, valid values, and property descriptions.

| Property | Description                                                                                                          | Units, Values                            |
|----------|----------------------------------------------------------------------------------------------------------------------|------------------------------------------|
| Ckts     | Cell array containing all circuit objects in the network, in order from source to load. All circuits must be 2-port. | Handles to rfckt objects. Default is {}. |
| Name     | Object name (read only)                                                                                              | String. 'Hybrid Connected Network'       |

# rfckt.hybrid

---

| Property | Description                                                                                                                        | Units, Values                   |
|----------|------------------------------------------------------------------------------------------------------------------------------------|---------------------------------|
| nPort    | Number of ports (read only)                                                                                                        | Integer. The value is always 2. |
| RFdata   | rfdata.data object describing a hybrid connected network. The analyze function creates this object and sets the value of 'RFdata'. | Handle. Default is [].          |

## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

## See Also

analyze, rfckt, rfckt.amplifier, rfckt.cascade, rfckt.datafile, rfckt.hybrid, rfckt.parallel, rfckt.seriesrlc, rfckt.shunrlc

**Purpose** Construct an LC bandpass pi network object

**Syntax** `h = rfckt.lcbandpasspi('Property1',value1,'Property2',value2,...)`  
`h = rfckt.lcbandpasspi`

**Description** `h = rfckt.lcbandpasspi('Property1',value1,'Property2',value2,...)` returns an LC bandpass pi network object, `h`, based on the specified properties. Properties you do not specify retain their default values.

`h = rfckt.lcbandpasspi` returns an LC bandpass pi network object whose properties all have their default values.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `lcbandpasspi` circuit object, use the `analyze` function to calculate the network parameters and noise figure at specified frequencies. For `rfckt.lcbandpasspi` objects, `freq` must be strictly positive.

```
analyze(h, freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

### Network Parameters

For each inductor and capacitor pair in the network, the `analyze` function first calculates the ABCD-parameters for each frequency in the input vector, `freq`. For each series pair,  $A = 1$ ,  $B = Z$ ,  $C = 0$ , and  $D = 1$ , where  $Z$  is the impedance of the series pair. For each shunt pair,  $A = 1$ ,  $B = 0$ ,  $C = Y$ , and  $D = 1$ , where  $Y$  is the admittance of the shunt pair.

The `analyze` function cascades the ABCD-parameters for each series and shunt pair, then converts the cascaded parameters to S-parameters using the `abcd2s` function.

# rfckt.lcbandpasspi

## Properties

This table lists properties useful to `rfckt.lcbandpasspi` objects along with units, valid values, and property descriptions.

| Property | Description                                                                                                                                                                                                         | Units, Values                                               |
|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------|
| C        | Vector containing the capacitances, in order from source to load, of all capacitors in the network. Its length must be equal to the length of the vector you provide for 'L'. All values must be strictly positive. | Farads.<br>Default is [0.3579e-10, 0.0118e-10, 0.3579e-10]. |
| L        | Vector containing the inductances, in order from source to load, of all inductors in the network. The inductance vector must contain at least three elements. All values must be strictly positive.                 | Henrys.<br>Default is [0.0144e-7, 0.4395e-7, 0.0144e-7].    |
| Name     | Object name (read only)                                                                                                                                                                                             | String. 'LC Bandpass Pi'                                    |
| nPort    | Number of ports (read only)                                                                                                                                                                                         | Integer. The value is always 2.                             |
| RFdata   | <code>rfdata.data</code> object describing an LC bandpass pi circuit. The <code>analyze</code> function creates this object and sets the value of 'RFdata'.                                                         | Handle.<br>Default is [].                                   |

## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

[2] Zverev, Anatol I., *Handbook of Filter Synthesis*, John Wiley & Sons, 1967.

## See Also

`analyze`, `rfckt`, `rfckt.lcbandpasstee`, `rfckt.lcbandstoppi`, `rfckt.lcbandstoptee`, `rfckt.lchighpasspi`, `rfckt.lchighpasstee`, `rfckt.lclowpasspi`, `rfckt.lclowpasstee`

**Purpose** Construct an LC bandpass tee network object

**Syntax** `h = rfckt.lcbandpasstee('Property1',value1,'Property2',value2,...)`  
`h = rfckt.lcbandpasstee`

**Description** `h = rfckt.lcbandpasstee('Property1',value1,'Property2',value2,...)` returns an LC bandpass tee network object, `h`, based on the specified properties. Properties you do not specify retain their default values.

`h = rfckt.lcbandpasstee` returns an LC bandpass tee network object whose properties all have their default values.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `lcbandpasstee` circuit object, use the `analyze` function to calculate the network parameters and noise figure at specified frequencies. For `rfckt.lcbandpasstee` objects, `freq` must be strictly positive.

```
analyze(h, freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

### Network Parameters

For each inductor and capacitor pair in the network, the `analyze` function first calculates the ABCD-parameters for each frequency in the input vector, `freq`. For each series pair,  $A = 1$ ,  $B = Z$ ,  $C = 0$ , and  $D = 1$ , where  $Z$  is the impedance of the series pair. For each shunt pair,  $A = 1$ ,  $B = 0$ ,  $C = Y$ , and  $D = 1$ , where  $Y$  is the admittance of the shunt pair.

The `analyze` function cascades the ABCD-parameters for each series and shunt pair, then converts the cascaded parameters to S-parameters using the `abcd2s` function.

# rfckt.lcbandpasstee

## Properties

This table lists properties useful to `rfckt.lcbandpasstee` objects along with units, valid values, and property descriptions.

| Property | Description                                                                                                                                                                                                         | Units, Values                                               |
|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------|
| C        | Vector containing the capacitances, in order from source to load, of all capacitors in the network. Its length must be equal to the length of the vector you provide for 'L'. All values must be strictly positive. | Farads.<br>Default is [0.0186e-10, 0.1716e-10, 0.0186e-10]. |
| L        | Vector containing the inductances, in order from source to load, of all inductors in the network. The inductance vector must contain at least three elements. All values must be strictly positive.                 | Henrys.<br>Default is [0.2781e-7, 0.0301e-7, 0.2781e-7].    |
| Name     | Object name (read only)                                                                                                                                                                                             | String. 'LC Bandpass Tee'                                   |
| nPort    | Number of ports (read only)                                                                                                                                                                                         | Integer. The value is always 2.                             |
| RFdata   | <code>rfdata.data</code> object describing an LC bandpass tee circuit. The <code>analyze</code> function creates this object and sets the value of 'RFdata'.                                                        | Handle.<br>Default is [].                                   |

## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

[2] Zverev, Anatol I., *Handbook of Filter Synthesis*, John Wiley & Sons, 1967.

## See Also

`analyze`, `rfckt`, `rfckt.lcbandpasspi`, `rfckt.lcbandstoppi`, `rfckt.lcbandstoptee`, `rfckt.lchighpasspi`, `rfckt.lchighpasstee`, `rfckt.lclowpasspi`, `rfckt.lclowpasstee`



**Purpose** Construct an LC bandstop pi network object

**Syntax**  
`h = rfckt.lcbandstoppi('Property1',value1,'Property2',value2,...)`  
`h = rfckt.lcbandstoppi`

**Description**  
`h = rfckt.lcbandstoppi('Property1',value1,'Property2',value2,...)` returns an LC bandstop pi network object, `h`, based on the specified properties. Properties you do not specify retain their default values.

`h = rfckt.lcbandstoppi` returns an LC bandstop pi network object whose properties all have their default values.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `lcbandstoppi` circuit object, use the `analyze` function to calculate the network parameters and noise figure at specified frequencies. For `rfckt.lcbandstoppi` objects, `freq` must be strictly positive.

```
analyze(h, freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

### Network Parameters

For each inductor and capacitor pair in the network, the `analyze` function first calculates the ABCD-parameters for each frequency in the input vector, `freq`. For each series pair,  $A = 1$ ,  $B = Z$ ,  $C = 0$ , and  $D = 1$ , where  $Z$  is the impedance of the series pair. For each shunt pair,  $A = 1$ ,  $B = 0$ ,  $C = Y$ , and  $D = 1$ , where  $Y$  is the admittance of the shunt pair.

The `analyze` function cascades the ABCD-parameters for each series and shunt pair, then converts the cascaded parameters to S-parameters using the `abcd2s` function.

# rfckt.lcbandstoppi

## Properties

This table lists properties useful to `rfckt.lcbandstoppi` objects along with units, valid values, and property descriptions.

| Property | Description                                                                                                                                                                                                         | Units, Values                                                        |
|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------|
| C        | Vector containing the capacitances, in order from source to load, of all capacitors in the network. Its length must be equal to the length of the vector you provide for 'L'. All values must be strictly positive. | Farads.<br>Default is<br>[0.0184e-10,<br>0.2287e-10,<br>0.0184e-10]. |
| L        | Vector containing the inductances, in order from source to load, of all inductors in the network. The inductance vector must contain at least three elements. All values must be strictly positive.                 | Henrys.<br>Default is<br>[0.2809e-7,<br>0.0226e-7,<br>0.2809e-7].    |
| Name     | Object name (read only)                                                                                                                                                                                             | String. 'LC<br>Bandstop Pi'                                          |
| nPort    | Number of ports (read only)                                                                                                                                                                                         | Integer. The<br>value is always<br>2.                                |
| RFdata   | <code>rfdata.data</code> object describing an LC bandstop pi circuit. The <code>analyze</code> function creates this object and sets the value of 'RFdata'.                                                         | Handle.<br>Default is [].                                            |

## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

[2] Zverev, Anatol I., *Handbook of Filter Synthesis*, John Wiley & Sons, 1967.

## See Also

`analyze`, `rfckt`, `rfckt.lcbandpasspi`, `rfckt.lcbandpasstee`,  
`rfckt.lcbandstoptee`, `rfckt.lchighpasspi`, `rfckt.lchighpasstee`,  
`rfckt.lclowpasspi`, `rfckt.lclowpasstee`

**Purpose** Construct an LC bandstop tee network object

**Syntax**  
`h = rfckt.lcbandstoptee('Property1',value1,'Property2',value2,...)`  
`h = rfckt.lcbandstoptee`

**Description**  
`h = rfckt.lcbandstoptee('Property1',value1,'Property2',value2,...)` returns an LC bandstop tee network object, `h`, based on the specified properties. Properties you do not specify retain their default values.

`h = rfckt.lcbandstoptee` returns an LC bandstop tee network object whose properties all have their default values.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `lcbandstoptee` circuit object, use the `analyze` function to calculate the network parameters and noise figure at specified frequencies. For `rfckt.lcbandstoptee` objects, `freq` must be strictly positive.

```
analyze(h, freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

### Network Parameters

For each inductor and capacitor pair in the network, the `analyze` function first calculates the ABCD-parameters for each frequency in the input vector, `freq`. For each series pair,  $A = 1$ ,  $B = Z$ ,  $C = 0$ , and  $D = 1$ , where  $Z$  is the impedance of the series pair. For each shunt pair,  $A = 1$ ,  $B = 0$ ,  $C = Y$ , and  $D = 1$ , where  $Y$  is the admittance of the shunt pair.

The `analyze` function cascades the ABCD-parameters for each series and shunt pair, then converts the cascaded parameters to S-parameters using the `abcd2s` function.

# rfckt.lcbandstoptee

## Properties

This table lists properties useful to `rfckt.lcbandstoptee` objects along with units, valid values, and property descriptions.

| Property | Description                                                                                                                                                                                                         | Units, Values                                                        |
|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------|
| C        | Vector containing the capacitances, in order from source to load, of all capacitors in the network. Its length must be equal to the length of the vector you provide for 'L'. All values must be strictly positive. | Farads.<br>Default is<br>[0.1852e-10,<br>0.0105e-10,<br>0.1852e-10]. |
| L        | Vector containing the inductances, in order from source to load, of all inductors in the network. The inductance vector must contain at least three elements. All values must be strictly positive.                 | Henrys.<br>Default is<br>[0.0279e-7,<br>0.4932e-7,<br>0.0279e-7].    |
| Name     | Object name (read only)                                                                                                                                                                                             | String. 'LC<br>Bandstop Tee'                                         |
| nPort    | Number of ports (read only)                                                                                                                                                                                         | Integer. The<br>value is always<br>2.                                |
| RFdata   | <code>rfdata.data</code> object describing an LC bandstop tee circuit. The <code>analyze</code> function creates this object and sets the value of 'RFdata'.                                                        | Handle.<br>Default is [].                                            |

## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

[2] Zverev, Anatol I., *Handbook of Filter Synthesis*, John Wiley & Sons, 1967.

## See Also

`analyze`, `rfckt`, `rfckt.lcbandpasspi`, `rfckt.lcbandpasstee`,  
`rfckt.lcbandstoppi`, `rfckt.lchighpasspi`, `rfckt.lchighpasstee`,  
`rfckt.lclowpasspi`, `rfckt.lclowpasstee`

**Purpose** Construct an LC highpass pi network object

**Syntax** `h = rfckt.lchighpasspi('Property1',value1,'Property2',value2,...)`  
`h = rfckt.lchighpasspi`

**Description** `h = rfckt.lchighpasspi('Property1',value1,'Property2',value2,...)` returns an LC highpass pi network object, `h`, based on the specified properties. Properties you do not specify retain their default values.

`h = rfckt.lchighpasspi` returns an LC highpass pi network object whose properties all have their default values.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `lchighpasspi` circuit object, use the `analyze` function to calculate the network parameters and noise figure at specified frequencies. For `rfckt.lchighpasspi` objects, `freq` must be strictly positive.

```
analyze(h, freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

### Network Parameters

For each inductor and capacitor in the network, the `analyze` function first calculates the ABCD-parameters for each frequency in the input vector, `freq`. For each series element,  $A = 1$ ,  $B = Z$ ,  $C = 0$ , and  $D = 1$ , where  $Z$  is the impedance of the series element. For each shunt element,  $A = 1$ ,  $B = 0$ ,  $C = Y$ , and  $D = 1$ , where  $Y$  is the admittance of the shunt element.

The `analyze` function cascades the ABCD-parameters for each circuit element, then converts the cascaded parameters to S-parameters using the `abcd2s` function.

# rfckt.lchighpasspi

## Properties

This table lists properties useful to `rfckt.lchighpasspi` objects along with units, valid values, and property descriptions.

| Property | Description                                                                                                                                                                                                                          | Units, Values                                 |
|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------|
| L        | Vector containing the inductances, in order from source to load, of all inductors in the network. The inductance vector must contain at least two elements. All values must be strictly positive.                                    | Henrys.<br>Default is [2.2363e-9].            |
| C        | Vector containing the capacitances, in order from source to load, of all capacitors in the network. Its length must be equal to or one less than the length of the vector you provide for 'L'. All values must be strictly positive. | Farads.<br>Default is [0.1188e-5, 0.1188e-5]. |
| Name     | Object name (read only)                                                                                                                                                                                                              | String. 'LC Highpass Pi'                      |
| nPort    | Number of ports (read only)                                                                                                                                                                                                          | Integer. The value is always 2.               |
| RFdata   | <code>rfdata.data</code> object describing an LC highpass pi circuit. The <code>analyze</code> function creates this object and sets the value of 'RFdata'.                                                                          | Handle.<br>Default is [].                     |

## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

[2] Zverev, Anatol I., *Handbook of Filter Synthesis*, John Wiley & Sons, 1967.

## See Also

`analyze`, `rfckt`, `rfckt.lcbandpasspi`, `rfckt.lcbandpasstee`, `rfckt.lcbandstoppi`, `rfckt.lcbandstoptee`, `rfckt.lchighpasstee`, `rfckt.lclowpasspi`, `rfckt.lclowpasstee`

**Purpose** Construct an LC highpass tee network object

**Syntax** `h = rfckt.lchighpasstee('Property1',value1,'Property2',value2,...)`  
`h = rfckt.lchighpasstee`

**Description** `h = rfckt.lchighpasstee('Property1',value1,'Property2',value2,...)` returns an LC highpass tee network object, `h`, with the specified properties. Properties you do not specify retain their default values.

`h = rfckt.lchighpasstee` returns an LC highpass tee network object whose properties all have their default values.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `lchighpasstee` circuit object, use the `analyze` function to calculate the network parameters and noise figure at specified frequencies. For `rfckt.lchighpasstee` objects, `freq` must be strictly positive.

```
analyze(h, freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

### Network Parameters

For each inductor and capacitor in the network, the `analyze` function first calculates the ABCD-parameters for each frequency in the input vector, `freq`. For each series element,  $A = 1$ ,  $B = Z$ ,  $C = 0$ , and  $D = 1$ , where  $Z$  is the impedance of the series element. For each shunt element,  $A = 1$ ,  $B = 0$ ,  $C = Y$ , and  $D = 1$ , where  $Y$  is the admittance of the shunt element.

The `analyze` function cascades the ABCD-parameters for each circuit element, then converts the cascaded parameters to S-parameters using the `abcd2s` function.

# rfckt.lchighpasstee

## Properties

This table lists properties useful to `rfckt.lchighpasstee` objects along with units, valid values, and property descriptions.

| Property | Description                                                                                                                                                                                                                                                                                        | Units, Values                                 |
|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------|
| C        | Vector containing the capacitances, in order from source to load, of all capacitors in the network. The capacitance vector must contain at least two elements. Its length must be equal to or one greater than the length of the vector you provide for 'L'. All values must be strictly positive. | Farads.<br>Default is [0.4752e-9, 0.4752e-9]. |
| L        | Vector containing the inductances, in order from source to load, of all inductors in the network. All values must be strictly positive. The vector cannot be empty.                                                                                                                                | Henrys.<br>Default is [5.5907e-6].            |
| Name     | Object name (read only)                                                                                                                                                                                                                                                                            | String. 'LC Highpass Tee'                     |
| nPort    | Number of ports (read only)                                                                                                                                                                                                                                                                        | Integer. The value is always 2.               |
| RFdata   | <code>rfdata.data</code> object describing an LC highpass tee circuit. The <code>analyze</code> function creates this object and sets the value of 'RFdata'.                                                                                                                                       | Handle.<br>Default is [].                     |

## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

[2] Zverev, Anatol I., *Handbook of Filter Synthesis*, John Wiley & Sons, 1967.

## See Also

`analyze`, `rfckt`, `rfckt.lcbandpasspi`, `rfckt.lcbandpasstee`, `rfckt.lcbandstoppi`, `rfckt.lcbandstoptee`, `rfckt.lchighpasspi`, `rfckt.lclowpasspi`, `rfckt.lclowpasstee`



**Purpose** Construct an LC lowpass pi network object

**Syntax**  
`h = rfckt.lclowpasspi('Property1',value1,'Property2',value2,...)`  
`h = rfckt.lclowpasspi`

**Description**  
`h = rfckt.lclowpasspi('Property1',value1,'Property2',value2,...)` returns an LC lowpass pi network object, `h`, based on the specified properties. Properties you do not specify retain their default values.

`h = rfckt.lclowpasspi` returns an LC lowpass pi network object whose properties all have their default values.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `lclowpasspi` circuit object, use the `analyze` function to calculate the network parameters and noise figure at specified frequencies. For `rfckt.lclowpasspi` objects, `freq` must be strictly positive.

```
analyze(h, freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

### Network Parameters

For each inductor and capacitor in the network, the `analyze` function first calculates the ABCD-parameters for each frequency in the input vector, `freq`. For each series element,  $A = 1$ ,  $B = Z$ ,  $C = 0$ , and  $D = 1$ , where  $Z$  is the impedance of the series element. For each shunt element,  $A = 1$ ,  $B = 0$ ,  $C = Y$ , and  $D = 1$ , where  $Y$  is the admittance of the shunt element.

The `analyze` function cascades the ABCD-parameters for each circuit element, then converts the cascaded parameters to S-parameters using the `abcd2s` function.

# rfckt.lclowpasspi

## Properties

This table lists properties useful to `rfckt.lclowpasspi` objects along with units, valid values, and property descriptions.

| Property | Description                                                                                                                                                                                                                                                                                        | Units, Values                              |
|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------|
| C        | Vector containing the capacitances, in order from source to load, of all capacitors in the network. The capacitance vector must contain at least two elements. Its length must be equal to or one greater than the length of the vector you provide for 'L'. All values must be strictly positive. | Farads. Default is [0.5330e-8, 0.5330e-8]. |
| L        | Vector containing the inductances, in order from source to load, of all inductors in the network. All values must be strictly positive. The vector cannot be empty.                                                                                                                                | Henrys. Default is [2.8318e-6].            |
| Name     | Object name (read only)                                                                                                                                                                                                                                                                            | String. 'LC Lowpass Pi'                    |
| nPort    | Number of ports (read only)                                                                                                                                                                                                                                                                        | Integer. The value is always 2.            |
| RFdata   | <code>rfdata.data</code> object describing an LC lowpass pi circuit. The <code>analyze</code> function creates this object and sets the value of 'RFdata'.                                                                                                                                         | Handle. Default is [].                     |

## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

[2] Zverev, Anatol I., *Handbook of Filter Synthesis*, John Wiley & Sons, 1967.

## See Also

`analyze`, `rfckt`, `rfckt.lcbandpasspi`, `rfckt.lcbandpasstee`, `rfckt.lcbandstoppi`, `rfckt.lcbandstoptee`, `rfckt.lchighpasspi`, `rfckt.lchighpasstee`, `rfckt.lclowpasstee`

**Purpose** Construct an LC lowpass tee network object

**Syntax**  
`h = rfckt.lclowpasstee('Property1',value1,'Property2',value2,...)`  
`h = rfckt.lclowpasstee`

**Description**  
`h = rfckt.lclowpasstee('Property1',value1,'Property2',value2,...)` returns an LC lowpass tee network object, `h`, based on the specified properties. Properties you do not specify retain their default values.

`h = rfckt.lclowpasstee` returns an LC lowpass tee network object whose properties all have their default values.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `lclowpasstee` circuit object, use the `analyze` function to calculate the network parameters and noise figure at specified frequencies. For `rfckt.lclowpasstee` objects, `freq` must be strictly positive.

```
analyze(h, freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

### Network Parameters

For each inductor and capacitor in the network, the `analyze` function first calculates the ABCD-parameters for each frequency in the input vector, `freq`. For each series element,  $A = 1$ ,  $B = Z$ ,  $C = 0$ , and  $D = 1$ , where  $Z$  is the impedance of the series element. For each shunt element,  $A = 1$ ,  $B = 0$ ,  $C = Y$ , and  $D = 1$ , where  $Y$  is the admittance of the shunt element.

The `analyze` function cascades the ABCD-parameters for each circuit element, then converts the cascaded parameters to S-parameters using the `abcd2s` function.

# rfckt.lclowpasstee

## Properties

This table lists properties useful to `rfckt.lclowpasstee` objects along with units, valid values, and property descriptions.

| Property | Description                                                                                                                                                                                                                          | Units, Values                                 |
|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------|
| C        | Vector containing the capacitances, in order from source to load, of all capacitors in the network. Its length must be equal to or one less than the length of the vector you provide for 'L'. All values must be strictly positive. | Farads.<br>Default is [1.1327e-9].            |
| L        | Vector containing the inductances, in order from source to load, of all inductors in the network. The inductance vector must contain at least two elements. All values must be strictly positive.                                    | Henrys.<br>Default is [0.1332e-4, 0.1332e-4]. |
| Name     | Object name (read only)                                                                                                                                                                                                              | String. 'LC Lowpass Tee'                      |
| nPort    | Number of ports (read only)                                                                                                                                                                                                          | Integer. The value is always 2.               |
| RFdata   | <code>rfdata.data</code> object describing an LC lowpass tee circuit. The <code>analyze</code> function creates this object and sets the value of 'RFdata'.                                                                          | Handle.<br>Default is [].                     |

## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

[2] Zverev, Anatol I., *Handbook of Filter Synthesis*, John Wiley & Sons, 1967.

## See Also

`analyze`, `rfckt`, `rfckt.lcbandpasspi`, `rfckt.lcbandpasstee`, `rfckt.lcbandstoppi`, `rfckt.lcbandstoptee`, `rfckt.lchighpasspi`, `rfckt.lchighpasstee`, `rfckt.lclowpasspi`

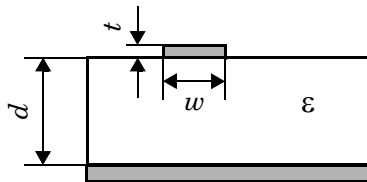
**Purpose** Construct a microstrip transmission line object

**Syntax**  
`h = rfckt.microstrip('Property1',value1,'Property2',value2,...)`  
`h = rfckt.microstrip`

**Description**  
`h = rfckt.microstrip('Property1',value1,'Property2',value2,...)` returns a microstrip transmission line object, `h`, with the specified properties. Properties you do not specify retain their default values.

`h = rfckt.microstrip` returns a microstrip transmission line object whose properties all have their default values.

A microstrip transmission line is shown here in cross-section. Its physical characteristics include the microstrip width ( $w$ ), the microstrip thickness ( $t$ ), the substrate height ( $d$ ), and the relative permittivity constant ( $\epsilon$ ).




---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the microstrip circuit object, use the `analyze` function to calculate the network parameters and noise figure at specified frequencies. For `rfckt.microstrip` objects, `freq` must be strictly positive.

```
analyze(h, freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

## Network Parameters

A microstrip transmission line object enables you to model the transmission line as a stub or as a stubless line.

**Stubless Transmission Line.** If you model the transmission line as a stubless line, the analyze function calculates the S-parameters for the specified frequencies, based on the physical length of the transmission line,  $D$ , and the complex propagation constant,  $k$ .

$$S_{11} = 0$$

$$S_{12} = e^{-kD}$$

$$S_{21} = e^{-kD}$$

$$S_{22} = 0$$

$k = \alpha_a + i\beta$ , where  $\alpha_a$  is the attenuation coefficient and  $\beta$  is the wave number. The attenuation coefficient  $\alpha_a$  is related to the loss,  $\alpha$ , by

$$\alpha_a = -\ln 10^{-\frac{\alpha}{20}}$$

where  $\alpha$  is the reduction in signal strength, in dB, per unit length.  $\alpha$  combines both conductor loss and dielectric loss and is derived from the rfckt.microstrip object properties.

The wave number  $\beta$  is related to the phase velocity,  $V_P$ , by

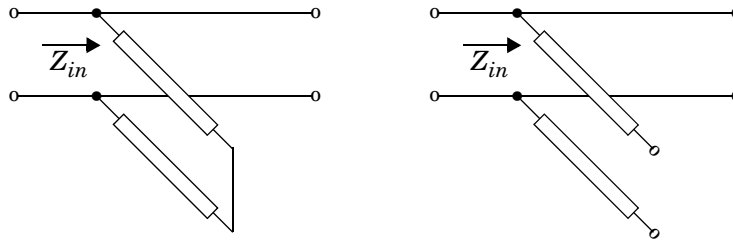
$$\beta = \frac{2\pi f}{V_P}$$

$V_P = c/\sqrt{\epsilon_{\text{eff}}}$  where  $\epsilon_{\text{eff}}$  is the frequency dependent effective dielectric constant.  $f$  is the frequency range specified in the analyze input argument freq.  $V_P$  and  $\epsilon_{\text{eff}}$  are derived from the rfckt.microstrip object properties.

The phase velocity  $V_p$  is also known as the wave propagation velocity.

**Shunt and Series Stubs.** If you model the transmission line as a shunt or series stub, the analyze function first calculates the ABCD-parameters at the specified frequencies. It then uses the abcd2s function to convert the ABCD-parameters to S-parameters.

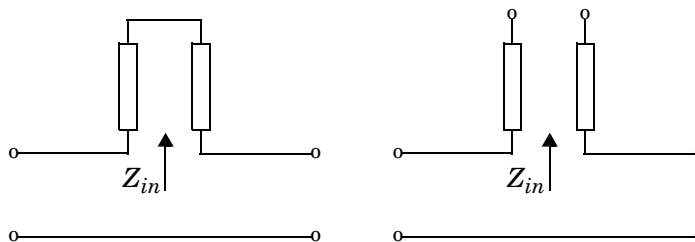
When you set the StubMode property to 'Shunt', the 2-port network consists of a stub transmission line that you can terminate with either a short circuit or an open circuit as shown here.



$Z_{in}$  is the input impedance of the shunt circuit. The ABCD-parameters for the shunt stub are calculated as

$$\begin{aligned} A &= 1 \\ B &= 0 \\ C &= 1/Z_{in} \\ D &= 1 \end{aligned}$$

When you set the StubMode property to 'Series', the 2-port network consists of a series transmission line that you can terminate with either a short circuit or an open circuit as shown here.



$Z_{in}$  is the input impedance of the series circuit. The ABCD-parameters for the series stub are calculated as

# rfckt.microstrip

$$\begin{aligned}A &= 1 \\B &= Z_{in} \\C &= 0 \\D &= 1\end{aligned}$$

## Properties

This table lists properties useful to `rfckt.microstrip` objects along with units, valid values, and property descriptions.

| Property    | Description                                                                                                                                     | Units, Values                          |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------|
| EpsilonR    | Relative permittivity of the dielectric expressed as the ratio of the permittivity of the dielectric to permittivity in free space $\epsilon_0$ | Default is 9.8.                        |
| Height      | Thickness of the dielectric on which the microstrip resides                                                                                     | Meters. Default is 6.35e-4.            |
| LineLength  | Physical length of the transmission line                                                                                                        | Meters. Default is 0.01.               |
| Loss        | Reduction in strength of the signal as it travels over the transmission line. Read-only; set by the <code>analyze</code> function.              | Decibels per meter. Default is [].     |
| LossTangent | Loss angle tangent of the dielectric                                                                                                            | Default is 0.                          |
| Name        | Object name (read only)                                                                                                                         | String. 'Microstrip Transmission Line' |
| nPort       | Number of ports (read only)                                                                                                                     | Integer. The value is always 2.        |
| PV          | Phase velocity. Propagation velocity of a uniform plane wave on the transmission line. Read-only; set by the <code>analyze</code> function.     | Meters per second. Default is [].      |



| Property    | Description                                                                                                                            | Units, Values                                                                              |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------|
| RFdata      | rfdata.data object describing a microstrip transmission line. The analyze function creates this object and sets the value of 'RFdata'. | Handle. Default is [].                                                                     |
| SigmaCond   | Conductivity in the conductor                                                                                                          | Siemens per meter (S/m). Default is Inf.                                                   |
| StubMode    | Type of stub                                                                                                                           | String.<br>'None' (default),<br>'Series', or<br>'Shunt'                                    |
| Termination | Termination for stub modes 'Shunt' and 'Series'.                                                                                       | String.<br>'None' (default),<br>'Open', or 'Short'.<br>Use 'None' when StubMode is 'None'. |
| Thickness   | Physical thickness of the microstrip                                                                                                   | Meters. Default is 5.0e-6.                                                                 |
| Width       | Physical width of the parallel-plate                                                                                                   | Meters. Default is 6.0e-4.                                                                 |
| Z0          | Characteristic impedance. Read-only; set by the analyze function.                                                                      | Ohms. Default is [].                                                                       |

## References

[1] Gupta, K.C., G. Ramesh, I. Bahl, and P. Bhartia, *Microstrip Lines and Slotlines*, Second Edition, Artech House, 1996. pp. 102-109.

## See Also

analyze, rfckt, rfckt.coaxial, rfckt.cpw, rfckt.parallelplate, rfckt.twowire, rfckt.txline

# rfckt.mixer

---

**Purpose** Construct a mixer object from a data file

**Syntax** `h = rfckt.mixer('Property1',value1,'Property2',value2,...)`  
`h = rfckt.mixer`

**Description** `h = rfckt.mixer('Property1',value1,'Property2',value2,...)` returns an `rfckt.mixer` circuit object, `h`, based on the specified properties. Use the 'File' property to specify a source `.amp`, `.s2p`, `.y2p`, or `.z2p` file that describes a mixer. Properties you do not specify retain their default values. See Chapter 3, “AMP File Format,” for information about the `.amp` format.

`h = rfckt.mixer` returns a mixer circuit object whose properties all have their default values.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `rfckt.mixer` circuit object, use the `analyze` function to calculate the network parameters, output power intercept point, and noise figure at specified frequencies. For `rfckt.mixer` objects, `freq` must be nonnegative.

```
analyze(h, freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

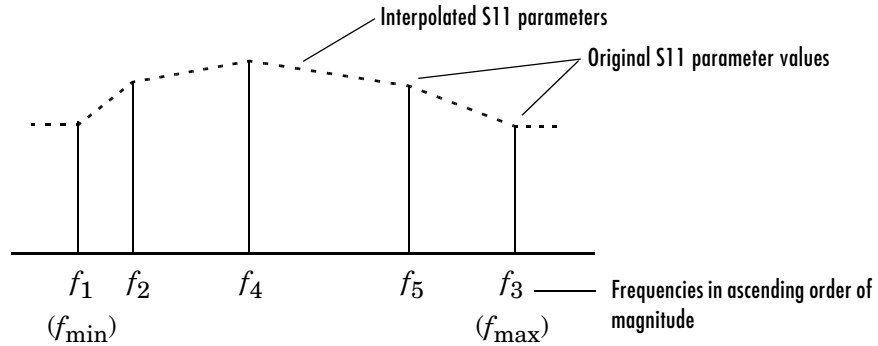
```
hd = getdata(h);
```

## Network Parameters

If the file you specify with the 'File' property contains network Y- or Z-parameters, `analyze` first converts these parameters, as they exist in the `rfckt.mixer` object, to S-parameters. Using the interpolation method you specify with the 'IntpType' property, `analyze` interpolates the S-parameters to determine the S-parameters at the specified frequencies.

Specifically, `analyze` orders the S-parameters according to the ascending order of their frequencies,  $f_n$ . It then interpolates the S-parameters, using the

MATLAB `interp1` function. For example, the curve in the following diagram illustrates the result of interpolating the S11 parameters at five different frequencies.



You can specify the interpolation method as `cubic`, `linear` (default), or `spline`. For more information, see “One-Dimensional Interpolation” and the `interp1` reference page in the MATLAB documentation.

As shown in the diagram above, `analyze` uses the parameter values at  $f_{\min}$ , the minimum input frequency, for all simulation frequencies smaller than  $f_{\min}$ . It uses the parameters values at  $f_{\max}$ , the maximum input frequency, for all simulation frequencies greater than  $f_{\max}$ . In both cases, the results may not be accurate.

### OIP3

If your file contains power data, the `analyze` function uses it to calculate OIP3. If your file contains no power data, set the value of the 'OIP3' property to an appropriate value.

### Noise Figure

If active spot noise data exists in the file you specify with the 'File' property, the `analyze` function uses the data to calculate the noise figure. It first interpolates the noise data for the simulation frequencies, using the interpolation method specified by the 'IntpType' property. It then calculates the noise figure using the resulting values.

If the file you specify contains no noise data, set the value of the 'NF' property to an appropriate value.

## Properties

This table lists properties useful to `rfckt.mixer` objects along with units, valid values, and property descriptions.

| Property   | Description                                                                                                                                             | Units, Values                              |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------|
| File       | .amp, .s2p, .y2p, or .z2p file describing a mixer                                                                                                       | Default is 'default.s2p'.                  |
| FLO        | Local oscillator frequency. For MixerType = 'downconverter', $f_{out} = f_{in} - f_{lo}$ . For MixerType = 'upconverter', $f_{out} = f_{in} + f_{lo}$ . | Hertz. Default is 1.0e+9                   |
| FreqOffset | Vector specifying the frequency offset for the phase noise level                                                                                        | Hertz. Default is [ ].                     |
| IntpType   | Interpolation method                                                                                                                                    | 'linear' (default), 'spline', or 'cubic'   |
| MixerType  | Type of mixer                                                                                                                                           | 'downconverter' (default) or 'upconverter' |
| Name       | Object name (read only)                                                                                                                                 | String. 'Mixer'                            |
| NF         | Noise figure. The amount of noise relative to a noise temperature of 290 degrees Kelvin. 0 dB indicates a noiseless system                              | Decibels. Default is 0.                    |
| nPort      | Number of ports (read only)                                                                                                                             | Integer. The value is always 2.            |
| OIP3       | Third order output intercept point of the mixer                                                                                                         | dBm. Default is Inf.                       |

| Property        | Description                                                                                                     | Units, Values                         |
|-----------------|-----------------------------------------------------------------------------------------------------------------|---------------------------------------|
| PhaseNoiseLevel | Vector specifying the phase noise level                                                                         | dBc/Hz. Default is [ ].               |
| RFdata          | rfdata.data object describing a mixer. The analyze function creates this object and sets the value of 'RFdata'. | Handle. Default is [1x1 rfdata.data]. |

## References

[1] EIA/IBIS Open Forum, “Touchstone File Format Specification,” Rev. 1.1, 2002 ([http://www.eda.org/pub/ibis/connector/touchstone\\_spec11.pdf](http://www.eda.org/pub/ibis/connector/touchstone_spec11.pdf)).

## See Also

analyze, rfckt, rfckt.amplifier, rfckt.cascade, rfckt.datafile, rfckt.hybrid, rfckt.parallel, rfckt.series, rfckt.seriesrlc, rfckt.shuntrlc

# rfckt.parallel

---

**Purpose** Construct a parallel connected network object

**Syntax** `h = rfckt.parallel('Property1',value1,'Property2',value2,...)`  
`h = rfckt.parallel`

**Description** `h = rfckt.parallel('Property1',value1,'Property2',value2,...)` returns a parallel connected network object, `h`, based on the specified properties. Use the 'Ckts' property to specify the 2-port `rfckt` objects to be connected. Properties you do not specify retain their default values.

`h = rfckt.parallel` returns a parallel connected network object whose properties all have their default values.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `parallel` network object, use the `analyze` function calculate the network parameters and noise figure at specified frequencies. For `rfckt.parallel` objects, `freq` must be strictly positive.

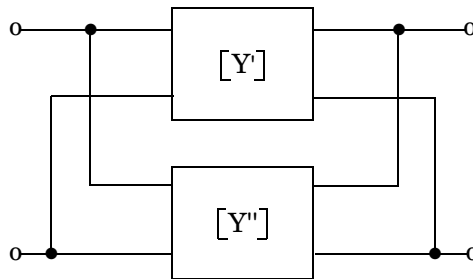
```
analyze(h,freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

## Network Parameters

The `analyze` function first calculates the admittance matrix of the parallel connected network. It starts by converting each component network's parameters to an admittance matrix. The figure shows a parallel connected network consisting of two 2-port networks, each represented by its admittance matrix.



where  $[Y'] = \begin{bmatrix} Y_{11}' & Y_{12}' \\ Y_{21}' & Y_{22}' \end{bmatrix}$  and  $[Y''] = \begin{bmatrix} Y_{11}'' & Y_{12}'' \\ Y_{21}'' & Y_{22}'' \end{bmatrix}$

The analyze function then calculates the admittance matrix for the parallel network by calculating the sum of the individual admittances. The following equation illustrates the calculations for two 2-port circuits.

$$[Y] = [Y'] + [Y''] = \begin{bmatrix} Y_{11}' + Y_{11}'' & Y_{12}' + Y_{12}'' \\ Y_{21}' + Y_{21}'' & Y_{22}' + Y_{22}'' \end{bmatrix}$$

Finally, analyze converts the admittance matrix of the parallel network to S-parameters at the frequencies specified in the analyze input argument freq.

## Properties

This table lists properties useful to rfckt.parallel objects along with units, valid values, and property descriptions.

| Property | Description                                                                                                          | Units, Values                            |
|----------|----------------------------------------------------------------------------------------------------------------------|------------------------------------------|
| Ckts     | Cell array containing all circuit objects in the network, in order from source to load. All circuits must be 2-port. | Handles to rfckt objects. Default is {}. |
| Name     | Object name (read only)                                                                                              | String. 'Parallel Connected Network'     |

# rfckt.parallel

---

| Property | Description                                                                                                                          | Units, Values                   |
|----------|--------------------------------------------------------------------------------------------------------------------------------------|---------------------------------|
| nPort    | Number of ports (read only)                                                                                                          | Integer. The value is always 2. |
| RFdata   | rfdata.data object describing a parallel connected network. The analyze function creates this object and sets the value of 'RFdata'. | Handle. Default is [].          |

## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

## See Also

analyze, rfckt, rfckt.amplifier, rfckt.cascade, rfckt.datafile, rfckt.hybrid, rfckt.series, rfckt.seriesrlc, rfckt.shuntrlc



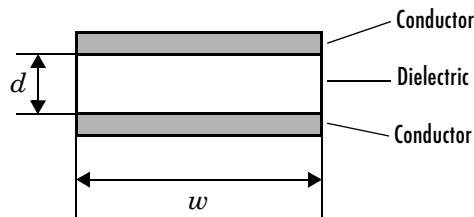
**Purpose** Construct a parallel-plate transmission line object

**Syntax**  
`h = rfckt.parallelplate('Property1',value1,'Property2',value2,...)`  
`h = rfckt.parallelplate`

**Description**  
`h = rfckt.parallelplate('Property1',value1,'Property2',value2,...)` returns a parallel-plate transmission line object, `h`, with the specified properties. Properties you do not specify retain their default values.

`h = rfckt.parallelplate` returns a parallel-plate transmission line object whose properties all have their default values.

A parallel-plate transmission line is shown here in cross-section. Its physical characteristics include the plate width  $w$  and the plate separation  $d$ .




---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `parallelplate` circuit object, use the `analyze` function to calculate the network parameters and noise figure at specified frequencies. For `rfckt.parallelplate` objects, `freq` must be strictly positive.

```
analyze(h, freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

## Network Parameters

A parallel-plate transmission line object enables you to model the transmission line as a stub or as a stubless line.

**Stubless Transmission Line.** If you model the transmission line as a stubless line, the analyze function calculates the S-parameters for the specified frequencies, based on the physical length of the transmission line,  $D$ , and the complex propagation constant,  $k$ .

$$S_{11} = 0$$

$$S_{12} = e^{-kD}$$

$$S_{21} = e^{-kD}$$

$$S_{22} = 0$$

$k$  is a vector whose elements correspond to the elements of the input vector freq.  $k$  can be expressed in terms of the resistance ( $R$ ), inductance ( $L$ ), conductance ( $G$ ), and capacitance ( $C$ ) per unit length (meters) as

$$k = k_r + jk_i = \sqrt{(R + j2\pi fL)(G + j2\pi fC)}$$

where  $f$  is the frequency range specified in the analyze input argument freq, and

$$R = \frac{2}{w\sigma_{\text{cond}}\delta}$$

$$L = \mu \frac{d}{w}$$

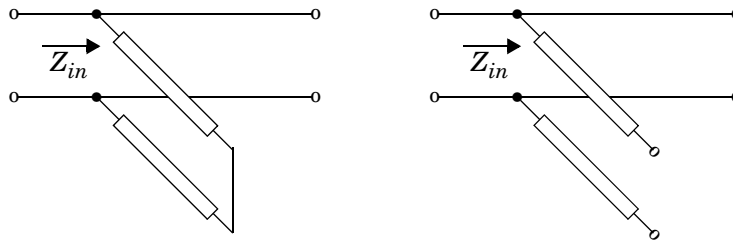
$$G = \sigma_{\text{diel}} \frac{w}{d}$$

$$C = \varepsilon \frac{w}{d}$$

In these equations,  $\sigma_{\text{cond}}$  is the conductivity in the conductor and  $\sigma_{\text{diel}}$  is the conductivity in the dielectric.  $\mu$  is the relative permeability of the dielectric,  $\varepsilon$  is its permittivity as derived from the EpsilonR property, and skin depth  $\delta$  is calculated as  $1/\sqrt{\pi f \mu \sigma_{\text{cond}}}$ .

**Shunt and Series Stubs.** If you model the transmission line as a shunt or series stub, the analyze function first calculates the ABCD-parameters at the specified frequencies. It then uses the `abcd2s` function to convert the ABCD-parameters to S-parameters.

When you set the `StubMode` property to 'Shunt', the 2-port network consists of a stub transmission line that you can terminate with either a short circuit or an open circuit as shown here.



$Z_{in}$  is the input impedance of the shunt circuit. The ABCD-parameters for the shunt stub are calculated as

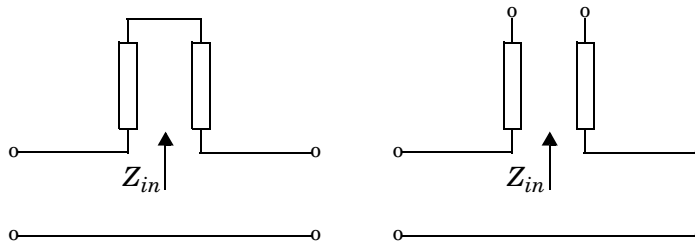
$$A = 1$$

$$B = 0$$

$$C = 1/Z_{in}$$

$$D = 1$$

When you set the `StubMode` property to 'Series', the 2-port network consists of a series transmission line that you can terminate with either a short circuit or an open circuit as shown here.



$Z_{in}$  is the input impedance of the series circuit. The ABCD-parameters for the series stub are calculated as

$$A = 1$$

$$B = Z_{in}$$

$$C = 0$$

$$D = 1$$

## Properties

This table lists properties useful to `rfckt.parallelplate` objects along with units, valid values, and property descriptions.

| Property   | Description                                                                                                                                     | Units, Values                                    |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------|
| EpsilonR   | Relative permittivity of the dielectric expressed as the ratio of the permittivity of the dielectric to permittivity in free space $\epsilon_0$ | Default is 2.3.                                  |
| LineLength | Physical length of the transmission line                                                                                                        | Meters. Default is 0.01.                         |
| Loss       | Reduction in strength of the signal as it travels over the transmission line. Read-only; set by the analyze function.                           | Decibels per meter. Default is [].               |
| MuR        | Relative permeability of the dielectric expressed as the ratio of the permeability of the dielectric to permeability in free space $\mu_0$      | Default is 1.                                    |
| Name       | Object name (read only)                                                                                                                         | String.<br>'Parallel-Plate<br>Transmission Line' |
| nPort      | Number of ports (read only)                                                                                                                     | Integer. The value is always 2.                  |

| Property    | Description                                                                                                                                | Units, Values                                                                              |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------|
| PV          | Phase velocity. Propagation velocity of a uniform plane wave on the transmission line. Read-only; set by the analyze function.             | Meters per second. Default is [ ].                                                         |
| RFdata      | rfdata.data object describing a parallel-plate transmission line. The analyze function creates this object and sets the value of 'RFdata'. | Handle. Default is [ ].                                                                    |
| Separation  | Thickness of the dielectric separating the plates                                                                                          | Meters. Default is 1.0e-3.                                                                 |
| SigmaCond   | Conductivity in the conductor                                                                                                              | Siemens per meter (S/m). Default is Inf.                                                   |
| SigmaDiel   | Conductivity in the dielectric                                                                                                             | Siemens per meter (S/m). Default is 0.                                                     |
| StubMode    | Type of stub                                                                                                                               | String.<br>'None' (default),<br>'Series', or 'Shunt'                                       |
| Termination | Termination for stub modes 'Shunt' and 'Series'.                                                                                           | String.<br>'None' (default),<br>'Open', or 'Short'.<br>Use 'None' when StubMode is 'None'. |
| Width       | Physical width of the parallel-plate transmission line                                                                                     | Meters. Default is .005.                                                                   |
| Z0          | Characteristic impedance. Read-only; set by the analyze function.                                                                          | Ohms. Default is [ ].                                                                      |

## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

# rfckt.parallelplate

---

## See Also

analyze, rfckt, rfckt.coaxial, rfckt.cpw, rfckt.microstrip,  
rfckt.twowire, rfckt.txline

**Purpose** Construct a series connected network object

**Syntax** `h = rfckt.series('Property1',value1,'Property2',value2,...)`  
`h = rfckt.series`

**Description** `h = rfckt.series('Property1',value1,'Property2',value2,...)` returns a series connected network object, `h`, based on the specified properties. Use the 'Ckts' property to specify the 2-port `rfckt` objects to be connected. Properties you do not specify retain their default values.

`h = rfckt.series` returns a series connected network object whose properties all have their default values.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the series network object, use the `analyze` function calculate the network parameters and noise figure at specified frequencies. For `rfckt.series` objects, `freq` must be strictly positive.

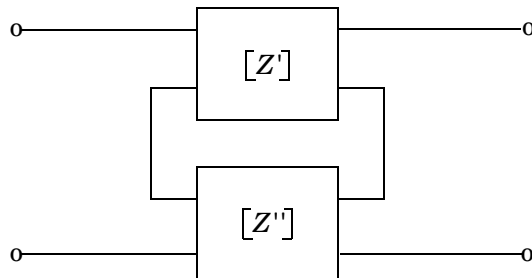
```
analyze(h,freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

### Network Parameters

The `analyze` function first calculates the impedance matrix of the series connected network. It starts by converting each component network's parameters to an impedance matrix. The figure shows a series connected network consisting of two 2-port networks, each represented by its impedance matrix.



where  $[Z'] = \begin{bmatrix} Z_{11}' & Z_{12}' \\ Z_{21}' & Z_{22}' \end{bmatrix}$  and  $[Z''] = \begin{bmatrix} Z_{11}'' & Z_{12}'' \\ Z_{21}'' & Z_{22}'' \end{bmatrix}$

The analyze function then calculates the impedance matrix for the series network by calculating the sum of the individual impedances. The following equation illustrates the calculations for two 2-port circuits.

$$[Z] = [Z'] + [Z''] = \begin{bmatrix} Z_{11}' + Z_{11}'' & Z_{12}' + Z_{12}'' \\ Z_{21}' + Z_{21}'' & Z_{22}' + Z_{22}'' \end{bmatrix}$$

Finally, analyze converts the impedance matrix of the series network to S-parameters at the frequencies specified in the analyze input argument freq.

## Properties

This table lists properties useful to rfckt.series objects along with units, valid values, and property descriptions.

| Property | Description                                                                                                          | Units, Values                            |
|----------|----------------------------------------------------------------------------------------------------------------------|------------------------------------------|
| Ckts     | Cell array containing all circuit objects in the network, in order from source to load. All circuits must be 2-port. | Handles to rfckt objects. Default is {}. |
| Name     | Object name (read only)                                                                                              | String. 'Series Connected Network'       |



| Property | Description                                                                                                                        | Units, Values                   |
|----------|------------------------------------------------------------------------------------------------------------------------------------|---------------------------------|
| nPort    | Number of ports (read only)                                                                                                        | Integer. The value is always 2. |
| RFdata   | rfdata.data object describing a series connected network. The analyze function creates this object and sets the value of 'RFdata'. | Handle. Default is [].          |

## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

## See Also

analyze, rfckt, rfckt.amplifier, rfckt.cascade, rfckt.datafile, rfckt.hybrid, rfckt.parallel, rfckt.seriesrlc, rfckt.shuntrlc

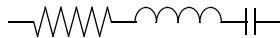
# rfckt.seriesrlc

---

**Purpose** Construct a series RLC network object

**Syntax**  
`h = rfckt.seriesrlc('R',Rvalue,'L',Lvalue,'C',Cvalue)`  
`h = rfckt.seriesrlc`

**Description** The series RLC network object is a two-port network as shown in the circuit diagram below.



`h = rfckt.seriesrlc('R',Rvalue,'L',Lvalue,'C',Cvalue)` returns a series RLC network object, `h`, based on the specified resistance (R), inductance (L), and capacitance (C) values. Properties you do not specify retain their default values, allowing you to specify a network of a single resistor, inductor, or capacitor.

`h = rfckt.seriesrlc` returns a series RLC network object whose properties all have their default values. This is equivalent to a pass-through two port network, i.e., the resistor, inductor, and capacitor are each replaced by a short circuit.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `seriesrlc` circuit object, use the `analyze` function to calculate the network parameters and noise correlation matrix at specified frequencies. For `rfckt.seriesrlc` objects, `freq` must be strictly positive.

```
analyze(h, freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

### Network Parameters

The analyze function first calculates the ABCD-parameters for the circuit, then converts the ABCD-parameters to S-parameters using the abcd2s function. For this circuit,  $A = 1$ ,  $B = Z$ ,  $C = 0$ , and  $D = 1$ , where

$$Z = \frac{-LC\omega^2 + jRC\omega + 1}{jC\omega}$$

where  $\omega = 2\pi f$ .

### Properties

This table lists properties useful to rfckt.seriesrlc objects along with units, valid values, and property descriptions.

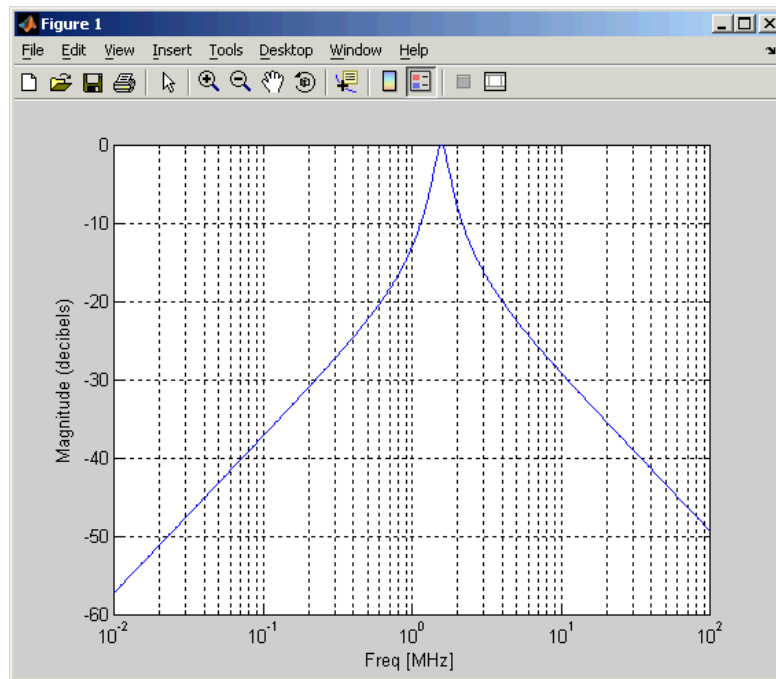
| Property | Description                                                                                                                  | Units, Values                   |
|----------|------------------------------------------------------------------------------------------------------------------------------|---------------------------------|
| C        | Scalar value for the capacitance                                                                                             | Farads. Default is Inf.         |
| L        | Scalar value for the inductance                                                                                              | Henries. Default is 0.          |
| Name     | Object name (read only)                                                                                                      | String, 'Series RLC'.           |
| nPort    | Number of ports (read only)                                                                                                  | Integer. The value is always 2. |
| R        | Scalar value for the resistance                                                                                              | Ohms. Default is 0.             |
| RFdata   | rfdata.data object describing a series RLC network. The analyze function creates this object and sets the value of 'RFdata'. | Handle. Default is [].          |

### Examples

This example creates a series LC resonator and examines its frequency response. It first creates the circuit object then uses the analyze function to calculate its frequency response. Finally, it plots the results – first, the magnitude in decibels (dB).

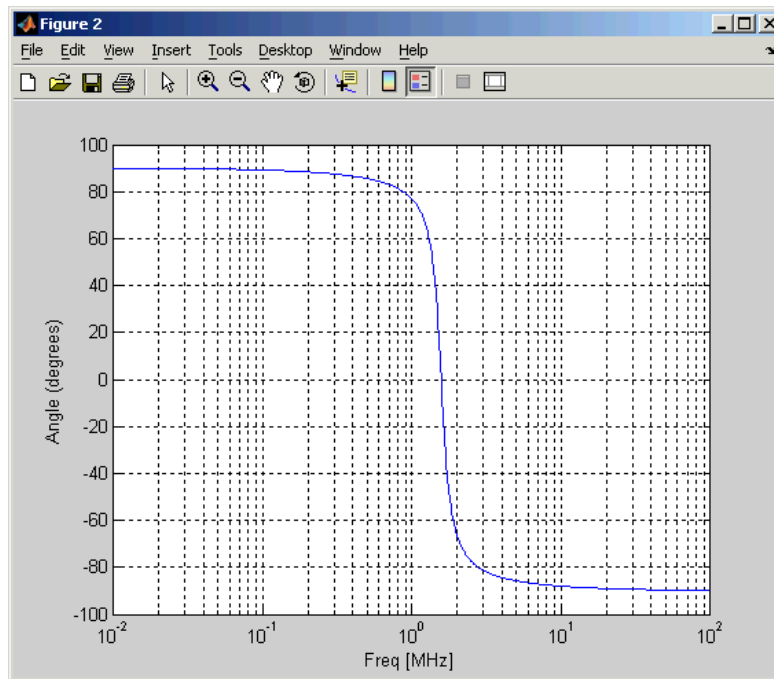
# rfckt.seriesrlc

```
h = rfckt.seriesrlc('L',4.7e-5,'C',2.2e-10);
analyze(h,logspace(4,8,1000));
plot(h,'s21','dB')
set(gca,'Xscale','log')
```



The example then plots the phase, in degrees

```
figure
plot(h,'s21','angle')
set(gca,'Xscale','log')
```



## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

## See Also

analyze, rfckt, rfckt.amplifier, rfckt.cascade, rfckt.hybrid, rfckt.mixer, rfckt.parallel, rfckt.series, rfckt.shuntrlc

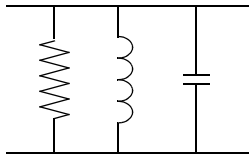
# rfckt.shuntrlc

---

**Purpose** Construct a shunt RLC network object

**Syntax**  
`h = rfckt.shuntrlc('R',Rvalue,'L',Lvalue,'C',Cvalue)`  
`h = rfckt.shuntrlc`

**Description** The shunt RLC network object is a two-port network as shown in the circuit diagram below.



`h = rfckt.shuntrlc('R',Rvalue,'L',Lvalue,'C',Cvalue)` returns a shunt RLC network object, `h`, based on the specified resistance (`R`), inductance (`L`), and capacitance (`C`) values. Properties you do not specify retain their default values, allowing you to specify a network of a single resistor, inductor, or capacitor.

`h = rfckt.shuntrlc` returns a shunt RLC network object whose properties all have their default values. This is equivalent to a pass-through two port network, i.e., the resistor, inductor, and capacitor are each replaced by an open circuit.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `shuntrlc` circuit object, use the `analyze` function to calculate the network parameters and noise correlation matrix at specified frequencies. For `rfckt.shuntrlc` objects, `freq` must be strictly positive.

```
analyze(h, freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain from the circuit object's `'RFdata'` property with the statement

```
hd = get(h, 'RFdata')
```

### Network Parameters

The analyze function first calculates the ABCD-parameters for the circuit, then converts the ABCD-parameters to S-parameters using the abcd2s function. For this circuit,  $A = 1$ ,  $B = 0$ ,  $C = Y$ , and  $D = 1$ , where

$$Y = \frac{-LC\omega^2 + j(L/R)\omega + 1}{jL\omega}$$

and  $\omega = 2\pi f$ .

### Properties

This table lists properties useful to rfckt.shuntrlc objects along with units, valid values, and property descriptions.

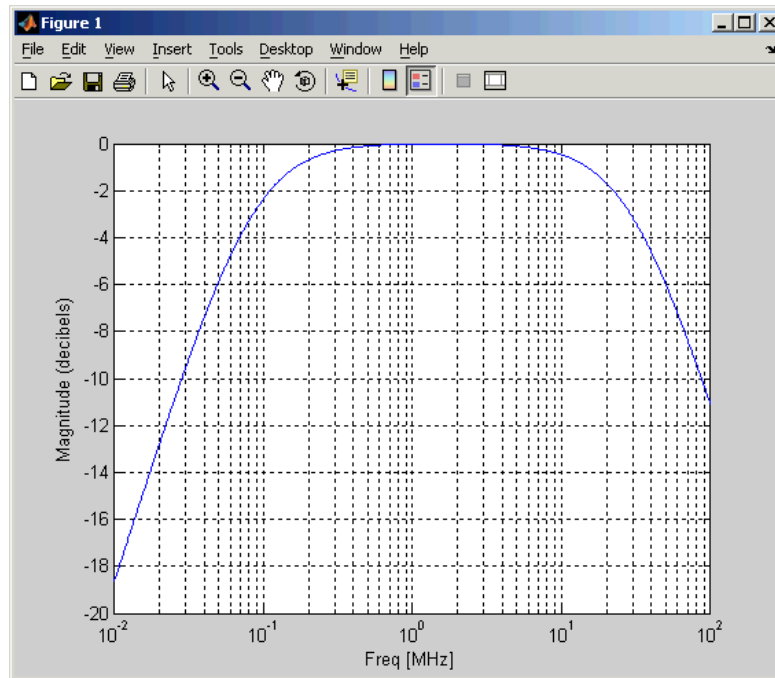
| Property | Description                                                                                                                  | Units, Values                   |
|----------|------------------------------------------------------------------------------------------------------------------------------|---------------------------------|
| C        | Scalar value for the capacitance                                                                                             | Farads. Default is 0.           |
| L        | Scalar value for the inductance                                                                                              | Henries. Default is Inf.        |
| Name     | Object name (read only)                                                                                                      | String. 'Shunt RLC'.            |
| nPort    | Number of ports (read only)                                                                                                  | Integer. The value is always 2. |
| R        | Scalar value for the resistance                                                                                              | Ohms. Default is Inf.           |
| RFdata   | rfdata.data object describing a series RLC network. The analyze function creates this object and sets the value of 'RFdata'. | Handle. Default is [].          |

### Examples

This example creates a shunt LC resonator and examines its frequency response. It first creates the circuit object then uses the analyze function to calculate its frequency response. Finally, it plots the results – first, the magnitude in decibels (dB).

# rfckt.shuntrlc

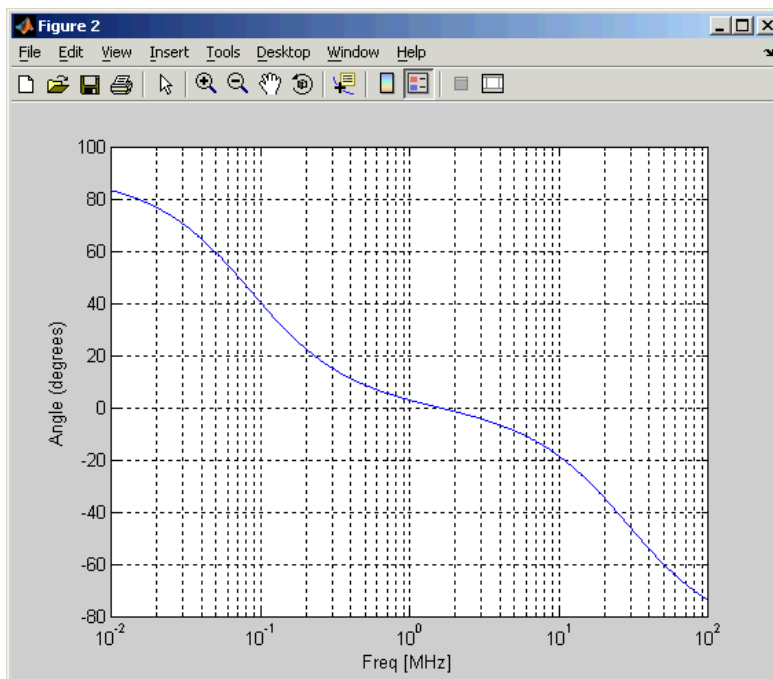
```
h = rfckt.shuntrlc('L',4.7e-5,'C',2.2e-10);
analyze(h,logspace(4,8,1000));
plot(h,'s21','dB')
set(gca,'Xscale','log')
```



The example then plots the phase, in degrees

```
figure
plot(h,'s21','angle')
set(gca,'Xscale','log')
```





## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

## See Also

analyze, rfckt, rfckt.amplifier, rfckt.cascade, rfckt.hybrid, rfckt.mixer, rfckt.parallel, rfckt.series, rfckt.seriesrlc

# rfckt.twowire

---

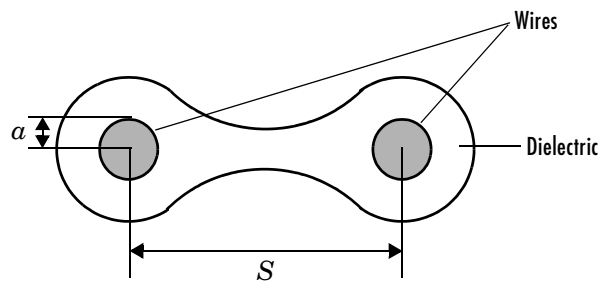
**Purpose** Construct a two-wire transmission line object

**Syntax** `h = rfckt.twowire('Property1',value1,'Property2',value2,...)`  
`h = rfckt.twowire`

**Description** `h = rfckt.twowire('Property1',value1,'Property2',value2,...)` returns a two-wire transmission line object, `h`, with the specified properties. Properties you do not specify retain their default values.

`h = rfckt.twowire` returns a two-wire transmission line object whose properties all have their default values.

A two-wire transmission line is shown here in cross-section. Its physical characteristics include the radius of the wires  $a$ , and the separation or physical distance between the wire centers  $S$ .



---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `twowire` circuit object, use the `analyze` function to calculate the network parameters and noise figure at specified frequencies. For `rfckt.twowire` objects, `freq` must be strictly positive.

`analyze(h, freq)`

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

### Network Parameters

A two-wire transmission line object enables you to model the transmission line as a stub or as a stubless line.

**Stubless Transmission Line.** If you model the transmission line as a stubless line, the `analyze` function calculates the S-parameters for the specified frequencies, based on the physical length of the transmission line,  $D$ , and the complex propagation constant,  $k$ .

$$S_{11} = 0$$

$$S_{12} = e^{-kD}$$

$$S_{21} = e^{-kD}$$

$$S_{22} = 0$$

$k$  is a vector whose elements correspond to the elements of the input vector `freq`.  $k$  can be expressed in terms of the resistance ( $R$ ), inductance ( $L$ ), conductance ( $G$ ), and capacitance ( $C$ ) per unit length (meters) as

$$k = k_r + jk_i = \sqrt{(R + j2\pi fL)(G + j2\pi fC)}$$

where  $f$  is the frequency range specified in the `analyze` input argument `freq`, and

$$R = \frac{1}{\pi a \sigma_{\text{cond}} \delta}$$

$$L = \frac{\mu}{\pi} \text{acosh}\left(\frac{D}{2a}\right)$$

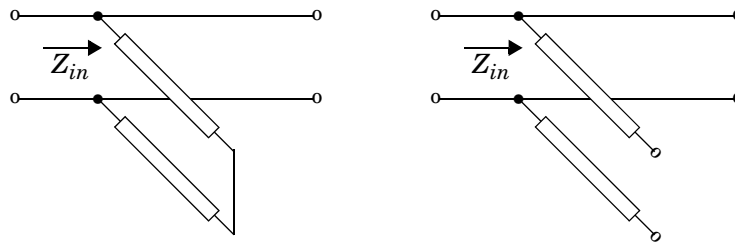
$$G = \frac{\pi \sigma_{\text{diel}}}{\text{acosh}(D/(2a))}$$

$$C = \frac{\pi \epsilon}{\text{acosh}(D/(2a))}$$

In these equations,  $\sigma_{\text{cond}}$  is the conductivity in the conductor and  $\sigma_{\text{diel}}$  is the conductivity in the dielectric.  $\mu$  is the relative permeability of the dielectric,  $\epsilon$  is its permittivity as derived from the EpsilonR property, and skin depth  $\delta$  is calculated as  $1/\sqrt{\pi f \mu \sigma_{\text{cond}}}$ .

**Shunt and Series Stubs.** If you model the transmission line as a shunt or series stub, the analyze function first calculates the ABCD-parameters at the specified frequencies. It then uses the abcd2s function to convert the ABCD-parameters to S-parameters.

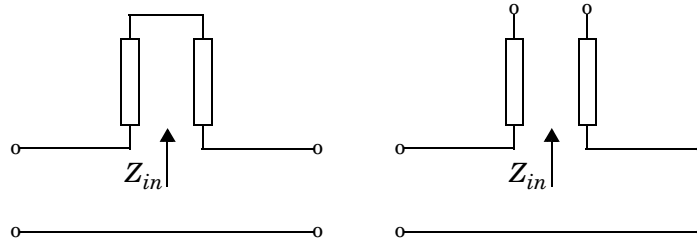
When you set the StubMode property to 'Shunt', the 2-port network consists of a stub transmission line that you can terminate with either a short circuit or an open circuit as shown here.



$Z_{in}$  is the input impedance of the shunt circuit. The ABCD-parameters for the shunt stub are calculated as

$$\begin{aligned} A &= 1 \\ B &= 0 \\ C &= 1/Z_{in} \\ D &= 1 \end{aligned}$$

When you set the StubMode property to 'Series', the 2-port network consists of a series transmission line that you can terminate with either a short circuit or an open circuit as shown here.



$Z_{in}$  is the input impedance of the series circuit. The ABCD-parameters for the series stub are calculated as

$$\begin{aligned}
 A &= 1 \\
 B &= Z_{in} \\
 C &= 0 \\
 D &= 1
 \end{aligned}$$

### Properties

This table lists properties useful to `rfckt.twowire` objects along with units, valid values, and property descriptions.

| Property   | Description                                                                                                                                     | Units, Values                      |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------|
| EpsilonR   | Relative permittivity of the dielectric expressed as the ratio of the permittivity of the dielectric to permittivity in free space $\epsilon_0$ | Default is 2.3.                    |
| LineLength | Physical length of the transmission line                                                                                                        | Meters. Default is 0.01.           |
| Loss       | Reduction in strength of the signal as it travels over the transmission line. Read-only; set by the <code>analyze</code> function.              | Decibels per meter. Default is []. |

| Property   | Description                                                                                                                                | Units, Values                                 |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------|
| MuR        | Relative permeability of the dielectric expressed as the ratio of the permeability of the dielectric to permeability in free space $\mu_0$ | Default is 1.                                 |
| Name       | Object name (read only)                                                                                                                    | String.<br>'Two-Wire<br>Transmission<br>Line' |
| nPort      | Number of ports (read only)                                                                                                                | Integer. The value is always 2.               |
| PV         | Phase velocity. Propagation velocity of a uniform plane wave on the transmission line. Read-only; set by the analyze function.             | Meters per second.<br>Default is [ ].         |
| Radius     | Radius of the conducting wires                                                                                                             | Meters. Default is $6.7e-4$ .                 |
| RFdata     | rfdata.data object describing a two-wire transmission line. The analyze function creates this object and sets the value of 'RFdata'.       | Handle. Default is [ ].                       |
| Separation | Physical distance between the wires                                                                                                        | Meters. Default is 0.0016.                    |
| SigmaCond  | Conductivity in conductor                                                                                                                  | Siemens per meter (S/m). Default is Inf.      |
| SigmaDiel  | Conductivity in dielectric                                                                                                                 | Siemens per meter (S/m). Default is 0.        |

| Property    | Description                                                             | Units, Values                                                                                    |
|-------------|-------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------|
| StubMode    | Type of stub                                                            | String.<br>'None' (default),<br>'Series', or<br>'Shunt'                                          |
| Termination | Termination for stub modes<br>'Shunt' and 'Series'.                     | String.<br>'None' (default),<br>'Open', or 'Short'.<br>Use 'None' when<br>StubMode is<br>'None'. |
| Z0          | Characteristic impedance.<br>Read-only; set by the analyze<br>function. | Ohms. Default<br>is [].                                                                          |

## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

## See Also

analyze, rfckt, rfckt.coaxial, rfckt.cpw, rfckt.microstrip,  
rfckt.parallelplate, rfckt.txline

# rfckt.txline

---

**Purpose** Construct a general transmission line object

**Syntax** `h = rfckt.txline('Property1',value1,'Property2',value2,...)`  
`h = rfckt.txline`

**Description** `h = rfckt.txline('Property1',value1,'Property2',value2,...)` returns a general transmission line object, `h`, with the specified properties. Properties you do not specify retain their default values.

`h = rfckt.txline` returns a general transmission line object whose properties all have their default values.

---

**Note** See the `rfckt` reference page for a list of functions that act on circuit (`rfckt`) objects.

---

**Circuit Analysis** After you create the `txline` circuit object, use the `analyze` function to calculate the network parameters and noise figure at specified frequencies. For `rfckt.txline` objects, `freq` must be strictly positive.

```
analyze(h,freq)
```

The `analyze` function stores the results of the analysis in an `rfdata.data` object whose handle you can obtain with the `getdata` function.

```
hd = getdata(h);
```

## Network Parameters

A general transmission line object enables you to model the transmission line as a stub or as a stubless line. The transmission line, which can be lossy or lossless, is treated as a 2-port linear network.

**Stubless Transmission Line.** If you model the transmission line as a stubless line, the `analyze` function calculates the S-parameters for the specified frequencies, based on the physical length of the transmission line,  $D$ , and the complex propagation constant,  $k$ .



$$S_{11} = 0$$

$$S_{12} = e^{-kD}$$

$$S_{21} = e^{-kD}$$

$$S_{22} = 0$$

$k$  is a vector whose elements correspond to the elements of the input vector freq.  $k = \alpha_a + i\beta$ , where  $\alpha_a$  is the attenuation coefficient and  $\beta$  is the wave number. The attenuation coefficient  $\alpha_a$  is related to the loss,  $\alpha$ , by

$$\alpha_a = -\ln 10^{-\frac{\alpha}{20}}$$

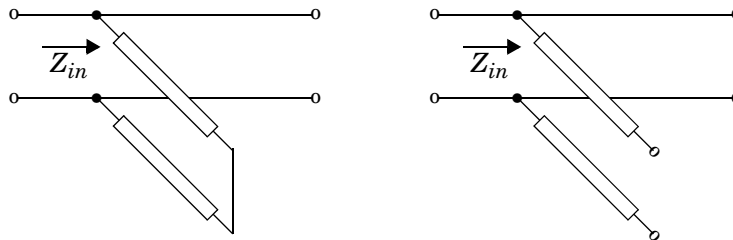
and the wave number  $\beta$  is related to the phase velocity,  $V_p$ , by

$$\beta = \frac{2\pi f}{V_p}$$

where  $f$  is the frequency range specified in the analyze input argument freq. The phase velocity  $V_p$  is derived from the rfckt.txline object properties. It is also known as the wave propagation velocity.

**Shunt and Series Stubs.** If you model the transmission line as a shunt or series stub, the analyze function first calculates the ABCD-parameters at the specified frequencies. It then uses the abcd2s function to convert the ABCD-parameters to S-parameters.

When you set the StubMode property to 'Shunt', the 2-port network consists of a stub transmission line that you can terminate with either a short circuit or an open circuit as shown here.



$Z_{in}$  is the input impedance of the shunt circuit. The ABCD-parameters for the shunt stub are calculated as

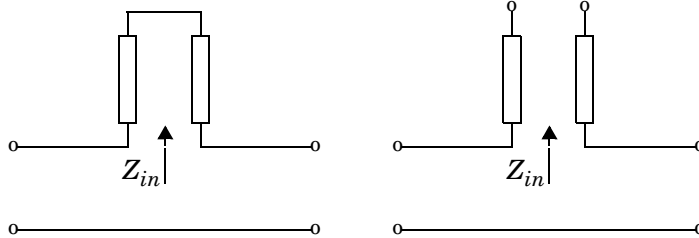
$$A = 1$$

$$B = 0$$

$$C = 1/Z_{in}$$

$$D = 1$$

When you set the StubMode property to 'Series', the 2-port network consists of a series transmission line that you can terminate with either a short circuit or an open circuit as shown here.



$Z_{in}$  is the input impedance of the series circuit. The ABCD-parameters for the series stub are calculated as

$$A = 1$$

$$B = Z_{in}$$

$$C = 0$$

$$D = 1$$

**Properties**

This table lists properties useful to `rfckt.txline` objects along with units, valid values, and property descriptions.

| <b>Property</b> | <b>Description</b>                                                                                                                                            | <b>Units, Values</b>                                    |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------|
| LineLength      | Physical length of the transmission line                                                                                                                      | Meters. Default is 0.01.                                |
| Loss            | Reduction in strength of the signal as it travels over the transmission line                                                                                  | Decibels per meter. Must be positive. Default is 0.     |
| Name            | Object name (read only)                                                                                                                                       | String.<br>'Transmission Line'                          |
| nPort           | Number of ports (read only)                                                                                                                                   | Integer. The value is always 2.                         |
| PV              | Phase velocity. Propagation velocity of a uniform plane wave on the transmission line                                                                         | Meters per second. Default is 299792458.                |
| RFdata          | <code>rfdata.data</code> object describing a general transmission line. The <code>analyze</code> function creates this object and sets the value of 'RFdata'. | Handle. Default is [].                                  |
| StubMode        | Type of stub                                                                                                                                                  | String.<br>'None' (default),<br>'Series', or<br>'Shunt' |

# rfckt.txline

---

| Property    | Description                                      | Units, Values                                                                     |
|-------------|--------------------------------------------------|-----------------------------------------------------------------------------------|
| Termination | Termination for stub modes 'Shunt' and 'Series'. | String. 'None' (default), 'Open', or 'Short'. Use 'None' when StubMode is 'None'. |
| Z0          | Characteristic impedance                         | Ohms. Default is 50.                                                              |

## References

[1] Ludwig, Reinhold and Pavel Bretchko, *RF Circuit Design: Theory and Applications*, Prentice-Hall, 2000.

## See Also

analyze, rfckt, rfckt.coaxial, rfckt.cpw, rfckt.microstrip, rfckt.parallelplate, rfckt.twowire

**Purpose** Construct an RF data object

**Description** An `rfdata` object contains network parameter data. Only the read and analyze functions can create an `rfdata` object.

See the individual `rfdata` object reference pages for information about a specific data object and its properties. See Chapter 2, “Working with Objects,” for additional information.

**Objects** The following table lists the available objects.

| <code>rfdata.type</code> | Description                               |
|--------------------------|-------------------------------------------|
| <code>rfdata.data</code> | Data object containing network parameters |

**Functions** The following functions act on `rfdata` objects.

| Function                | Purpose                                                                    |
|-------------------------|----------------------------------------------------------------------------|
| <code>analyze</code>    | Analyze a circuit or data object in the frequency domain.                  |
| <code>calculate</code>  | Calculate specified network parameters for a circuit or data object        |
| <code>copy</code>       | Copy a circuit or data object                                              |
| <code>extract</code>    | Extract network parameters from a data object                              |
| <code>listformat</code> | List valid formats for a network parameter                                 |
| <code>listparam</code>  | List valid network parameter for a circuit or data object                  |
| <code>plot</code>       | Plot network parameters from a circuit or data object on an X-Y plane      |
| <code>polar</code>      | Plot network parameters from a circuit or data object on polar coordinates |
| <code>read</code>       | Read RF network parameters from a file to a new or existing data object.   |

| Function | Purpose                                                                |
|----------|------------------------------------------------------------------------|
| smith    | Plot network parameters from a circuit or data object on a Smith chart |
| write    | Write RF data from a data object to a file.                            |

## Properties

Properties vary for each type of object. See the individual object reference pages for information about properties.

### Viewing Object Properties

You can use `get` to view an `rfdata` object's properties. To see a specific property, use

```
get(h, 'PropertyName')
```

To see all properties for an object, use

```
get(h)
```

### Changing Object Properties

To see the `rfdata` properties whose values you can change use

```
set(h)
```

To change specific properties, use

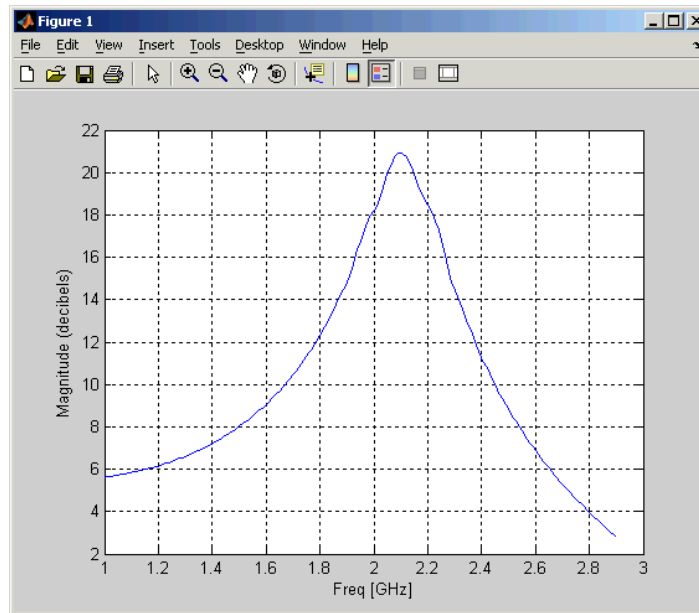
```
set(h, 'PropertyName1', value1, 'PropertyName2', value2, ...)
```

Note that you must use single quotation marks around the property name.

## Examples

Construct an RF data object from a `.s2p` data file.

```
file = 'default.s2p';
h = read(rfdata.data, file); % Read file into rfdata.data object.
figure
plot(h, 's21', 'db'); % Plot dB(S21) in XY plane.
```



You can also use other RF Toolbox functions such as `polar` and `smith` to visualize results.

### See Also

`analyze`, `calculate`, `copy`, `extract`, `listformat`, `listparam`, `plot`, `polar`, `read`, `rfckt`, `smith`, `write`

# rfdata.data

---

**Purpose** Network parameters data object

**Description** An `rfdata.data` object contains network parameter data. Only the `read` function and the `rfckt/analyze` and `rfdata/analyze` functions can create an `rfdata.data` object.

- `read` reads network parameters from a data file and writes those parameters to an `rfdata.data` object.
- `analyze` stores the results of its analysis in an `rfdata.data` object.

---

**Note** See the `rfdata` reference page for a list of functions that act on `rfdata.data` objects.

---

Use `get` and `set` to view and change `rfdata.data` object properties. To see a specific property, use

```
get(h, 'PropertyName')
```

To change specific properties, use

```
set(h, 'PropertyName1', value1, 'PropertyName2', value2, ...)
```

**Properties** This table lists properties useful to `rfdata.data` objects along with units, valid values, and property descriptions.

| Property | Description                                                                                                                                                            | Units, Values                            |
|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------|
| Freq     | Frequencies of the S-parameters as an M-element vector. The order of the frequencies must correspond to the order of 'S-parameters'. All frequencies must be positive. | Hertz. Default is [].                    |
| IntpType | Interpolation method                                                                                                                                                   | 'linear' (default), 'spline', or 'cubic' |



| <b>Property</b> | <b>Description</b>                                                                                                         | <b>Units, Values</b>               |
|-----------------|----------------------------------------------------------------------------------------------------------------------------|------------------------------------|
| Name            | Object name (read only)                                                                                                    | String.<br>'rfdata.data<br>object' |
| NF              | Noise figure. The amount of noise relative to a noise temperature of 290 degrees Kelvin. 0 dB indicates a noiseless system | Decibels. Default is 1.            |
| OIP3            | Third order output intercept point                                                                                         | dBm. Default is Inf.               |
| S_Parameters    | S-parameters of the circuit described by the rfdata.data object in a 2-by-2-by-M array. M is the number of S-parameters.   | Default is [].                     |
| Z0              | Reference impedance                                                                                                        | Ohms. Default is 50.               |
| ZL              | Load impedance                                                                                                             | Ohms. Default is 50.               |
| ZS              | Source impedance                                                                                                           | Ohms. Default is 50.               |

**See Also**

analyze, rfckt, rfdata

# rftool

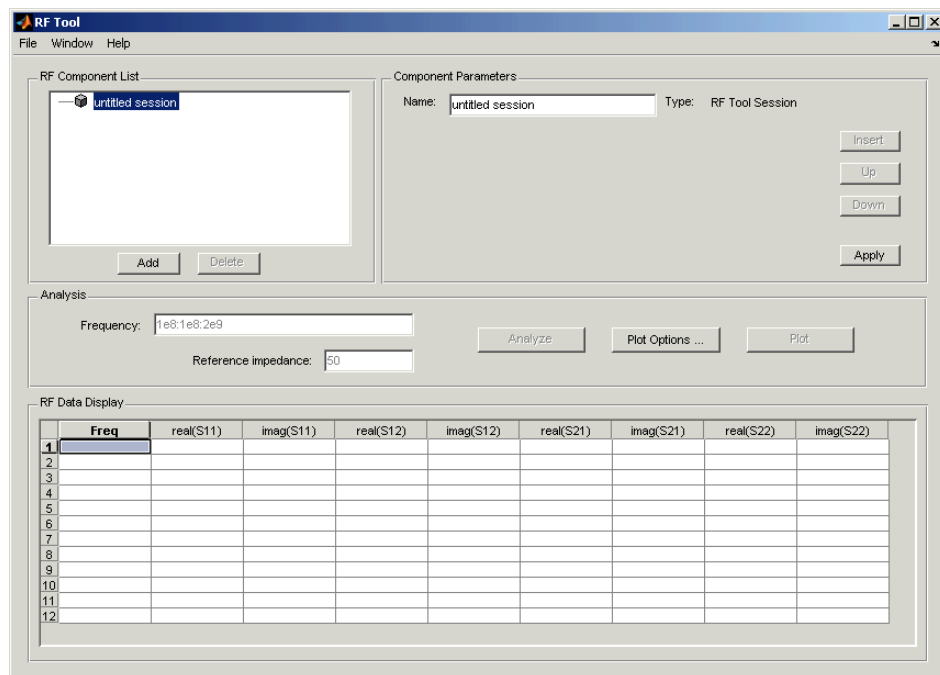
**Purpose** Open the RF Analysis Tool (RFTool)

**Syntax** rftool

**Description** rftool opens RFTool. Use this tool to:

- Create circuit components and set their parameters
- Analyze components over a specified frequency range and step size
- Plot the analysis results
- Import component objects to and export them from the MATLAB workspace
- Save RFTool sessions for later use

See Chapter 4, “RF Tool: An RF Analysis GUI” for more information



**Purpose** Convert S-parameters to ABCD-parameters

**Syntax** `abcd_params = s2abcd(s_params, z0)`

**Description** `abcd_params = s2abcd(s_params, z0)` converts the scattering parameters `s_params` into the ABCD parameters `abcd_params`. The `s_params` input is a complex 2-by-2-by-`m` array, representing `m` two-port S-parameters. `z0` is the reference impedance; its default is 50 ohms. `abcd_params` is a complex 2-by-2-by-`m` array, representing `m` two-port ABCD-parameters.

**See Also** `abcd2s`, `h2abcd`, `s2y`, `s2z`, `s2h`, `y2abcd`, `z2abcd`

# s2h

---

**Purpose** Convert S-parameters to h-parameters

**Syntax** `h_params = s2h(s_params, z0)`

**Description** `h_params = s2h(s_params, z0)` converts the scattering parameters `s_params` into the hybrid parameters `h_params`. The `s_params` input is a complex 2-by-2-by-`m` array, representing `m` two-port S-parameters. `z0` is the reference impedance; its default is 50 ohms. `h_params` is a complex 2-by-2-by-`m` array, representing `m` two-port h-parameters.

**See Also** `abcd2h`, `h2s`, `s2abcd`, `s2y`, `s2z`, `y2h`, `z2h`

**Purpose** Convert S-parameters to S-parameters with different impedance

**Syntax**  
`s_params_new = s2s(s_params, z0)`  
`s_params_new = s2s(s_params, z0, z0_new)`

**Description** `s_params_new = s2s(s_params, z0)` converts the scattering parameters `s_params` with reference impedance `z0` into the scattering parameters `s_params_new` with reference impedance 50 ohms. `s_params_new` is a complex `n`-by-`n`-by-`m` array, representing `m` `n`-port S-parameters. `s_params` is a complex `n`-by-`n`-by-`m` array, representing `m` `n`-port S-parameters. `z0` is the reference impedance of the input S-parameters.

`s_params_new = s2s(s_params, z0, z0_new)` converts the scattering parameters `s_params` with reference impedance `z0` into the scattering parameters `s_params_new` with the reference impedance `z0_new`.

**See Also** `abcd2s`, `h2s`, `s2abcd`, `s2h`, `s2y`, `s2z`, `y2s`, `z2s`

# s2t

---

**Purpose** Convert S-parameters to T-parameters

**Syntax** `t_params = s2t(s_params)`

**Description** `t_params = s2t(s_params)` converts the scattering parameters `s_params` into the chain scattering parameters `t_params`. The `s_params` input is a complex 2-by-2-by-`m` array, representing `m` two-port S-parameters. `t_params` is a complex 2-by-2-by-`m` array, representing `m` two-port T-parameters.

**See Also** `s2abcd`, `s2h`, `s2y`, `s2z`, `t2s`

**Purpose** Convert S-parameters to Y-parameters

**Syntax** `y_params = s2y(s_params, z0)`

**Description** `y_params = s2y(s_params, z0)` converts the scattering parameters `s_params` into the admittance parameters `y_params`. The `s_params` input is a complex `n`-by-`n`-by-`m` array, representing `m` `n`-port S-parameters. `z0` is the reference impedance; its default is 50 ohms. `y_params` is a complex `n`-by-`n`-by-`m` array, representing `m` `n`-port Y-parameters.

**See Also** `abcd2y`, `h2y`, `s2abcd`, `s2h`, `s2z`, `y2s`, `z2y`

## s2z

---

**Purpose** Convert S-parameters to Z-parameters

**Syntax** `z_params = s2h(s_params, z0)`

**Description** `z_params = s2h(s_params, z0)` converts the scattering parameters `s_params` into the impedance parameters `z_params`. The `s_params` input is a complex  $n$ -by- $n$ -by- $m$  array, representing  $m$   $n$ -port S-parameters. `z0` is the reference impedance; its default is 50 ohms. `z_params` is a complex  $n$ -by- $n$ -by- $m$  array, representing  $m$   $n$ -port Z-parameters.

**See Also** `abcd2z`, `h2z`, `s2abcd`, `s2h`, `s2y`, `y2z`, `z2s`



---

|                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Purpose</b>     | Plot network parameters from a circuit or data object on a Smith chart                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Syntax</b>      | <code>[lineseries,hsm] = smith(h,parameter1,...,parameterN,type)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>Description</b> | <code>[lineseries,hsm] = smith(h,parameter1,...,parameterN,type)</code> plots the network parameters <code>parameter1,...,parameterN</code> from the object <code>h</code> on a Smith chart. <code>h</code> is the handle of a circuit ( <code>rfckt</code> ) or data ( <code>rfdata</code> ) object. <code>type</code> is a string, 'z' (default), 'y', or 'zy', specifying the type of Smith chart.<br><br>Type <code>listparam(h)</code> to get a list of valid network parameters for a circuit or data object <code>h</code> . |

---

**Note** For all circuit and data objects except those that contain data from a data file, you must use the `analyze` function to perform a frequency domain analysis before calling `smith`.

---

---

**Note** Use the `smithchart` function to plot network parameters that are not part of a circuit (`rfckt`) or data (`rfdata`) object, but are specified as vector data.

---

### Changing Properties of the Plotted Lines

The `smith` function returns `lineseries`, a column vector of handles to `lineseries` objects, one handle per plotted line. Use the MATLAB `lineseries` properties to change the properties of these lines.

### Changing Properties of the Smith Chart

The `smith` function returns the handle `hsm` of the Smith chart. Use the properties listed below to change the properties of the chart itself.

|                   |                                                                                                                                                                                                                                                        |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Properties</b> | <code>smith</code> creates the plot using the default property values of a Smith chart. Use <code>set(hsm,'PropertyName1',PropertyValue1,...)</code> to change the property values of the chart. Use <code>get(hsm)</code> to get the property values. |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

This section lists all properties you can specify for a Smith chart object along with units, valid values, and a descriptions of their use.

| <b>Property Name</b> | <b>Description</b>                                                           | <b>Units, Values</b>                               |
|----------------------|------------------------------------------------------------------------------|----------------------------------------------------|
| Color                | Line color for a Z or Y Smith chart. For a ZY Smith chart, the Z line color. | ColorSpec. Default is [0.4 0.4 0.4] (dark grey).   |
| LabelColor           | Color of the line labels.                                                    | ColorSpec. Default is [0 0 0] (black).             |
| LabelSize            | Size of the line labels.                                                     | FontUnits. Default is 10.                          |
| LabelVisible         | Visibility of the line labels.                                               | 'on' (default) or 'off'                            |
| LineType             | Line spec for a Z or Y Smith chart. For a ZY Smith chart, the Z line spec.   | LineSpec. Default is '-' (solid line).             |
| LineWidth            | Line width for a Z or Y Smith chart. For a ZY Smith chart, the Z line width. | Number of points. Default is 0.5.                  |
| SubColor             | The Y line color for a ZY Smith chart.                                       | ColorSpec. Default is [0.8 0.8 0.8] (medium grey). |
| SubLineType          | The Y line spec for a ZY Smith chart.                                        | LineSpec. Default is ':' (dotted line).            |
| SubLineWidth         | The Y line width for a ZY Smith chart.                                       | Number of points. Default is 0.5.                  |

| Property Name | Description                                                                                                  | Units, Values                                                                                       |
|---------------|--------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------|
| Type          | Type of Smith chart                                                                                          | 'z' (default), 'y', or 'zy'                                                                         |
| Value         | Two-row matrix. Row 1 specifies the constant resistance lines. Row 2 specifies the constant reactance lines. | 2-by-n matrix. Default is [0.2000 0.5000 1.0000 2.0000 5.0000; 1.0000 2.0000 5.0000 5.0000 30.0000] |

**See Also**

get, listformat, listparam, rfckt, rfddata, set, smithchart

# smithchart

---

**Purpose** Plot complex vector on a Smith chart

**Syntax** `[lineseries,hsm] = smithchart(y)`  
`[lineseries,hsm] = smithchart`

**Description** `[lineseries,hsm] = smithchart(y)` plots the complex vector `y` on a Smith chart and returns the handle `h` of the Smith chart object. Change the Smith chart properties to customize the chart.

The `smithchart` function returns `lineseries`, a column vector of handles to `lineseries` objects, one handle per plotted line. Use the `lineseries` properties to change the properties of these lines.

The `smithchart` function also returns the handle `hsm` to the Smith chart. Use the properties listed below to change the properties of the chart itself.

`[lineseries,hsm] = smithchart` draws a blank Smith chart.

---

**Note** To plot network parameters from a circuit (`rfckt`) or data (`rfdata`) object on a Smith chart, use the `smith` function.

---

**Properties** `smithchart` creates the plot using default property values of a Smith chart. Use `set(h, 'PropertyName1', PropertyValue1, ...)` to change the property values. Use `get(h)` to get the property values.

This section lists all properties you can specify for `smithchart` objects along with units, valid values, and a descriptions of their use.

| Property Name | Description                                                                  | Units, Values                                    |
|---------------|------------------------------------------------------------------------------|--------------------------------------------------|
| Color         | Line color for a Z or Y Smith chart. For a ZY Smith chart, the Z line color. | ColorSpec. Default is [0.4 0.4 0.4] (dark grey). |
| LabelColor    | Color of the line labels.                                                    | ColorSpec. Default is [0 0 0] (black).           |

| Property Name | Description                                                                                                  | Units, Values                                                                                       |
|---------------|--------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------|
| LabelSize     | Size of the line labels.                                                                                     | FontUnits. Default is 10.                                                                           |
| LabelVisible  | Visibility of the line labels.                                                                               | 'on' (default) or 'off'                                                                             |
| LineType      | Line spec for a Z or Y Smith chart. For a ZY Smith chart, the Z line spec.                                   | LineStyle. Default is '-' (solid line).                                                             |
| LineWidth     | Line width for a Z or Y Smith chart. For a ZY Smith chart, the Z line width.                                 | Number of points. Default is 0.5.                                                                   |
| SubColor      | The Y line color for a ZY Smith chart.                                                                       | ColorSpec. Default is [0.8 0.8 0.8] (medium grey).                                                  |
| SubLineType   | The Y line spec for a ZY Smith chart.                                                                        | LineStyle. Default is ':' (dotted line).                                                            |
| SubLineWidth  | The Y line width for a ZY Smith chart.                                                                       | Number of points. Default is 0.5.                                                                   |
| Type          | Type of Smith chart                                                                                          | 'z' (default), 'y', or 'zy'                                                                         |
| Value         | Two-row matrix. Row 1 specifies the constant resistance lines. Row 2 specifies the constant reactance lines. | 2-by-n matrix. Default is [0.2000 0.5000 1.0000 2.0000 5.0000; 1.0000 2.0000 5.0000 5.0000 30.0000] |

**See Also**

get, rfckt, rfddata, set, smith

## t2s

---

**Purpose** Convert T-parameters to S-parameters

**Syntax** `s_params = t2s(t_params)`

**Description** `s_params = t2s(t_params)` converts the chain scattering parameters `t_params` into the scattering parameters `s_params`. The `t_params` input is a complex 2-by-2-by-`m` array, representing `m` two-port T-parameters. `s_params` is a complex 2-by-2-by-`m` array, representing `m` two-port S-parameters.

**See Also** `abcd2s`, `h2s`, `s2t`, `y2s`, `z2s`

**Purpose**                      Calculates the VSWR at the given reflection coefficient gamma

**Syntax**                      `result = vswr(gamma)`

**Description**                `result = vswr(gamma)` calculates the voltage standing-wave ratio (VSWR) at the given reflection coefficient gamma as

$$\text{VSWR} = \frac{1 + |\Gamma|}{1 - |\Gamma|}$$

where  $\Gamma$  is the given reflection coefficient gamma. The input gamma is a complex vector. `result` is a real vector of the same length as gamma.

**See Also**                      `gammain`, `gammaout`

# write

---

**Purpose** Create a formatted RF network data file

**Syntax** `status = write(data,filename,dataformat,funit,printfmt,freqformat)`

**Description** `status = write(data,filename,dataformat,funit,printfmt,freqformat)` writes information from `data` to the specified file. `data` is an `rfdata.data` object that contains sufficient information to write the specified file. `filename` is a string representing the filename of a `.snp`, `.ynp`, `.znp`, `.hnp`, or `.amp` file, where `n` is the number of ports. The default filename extension is `.snp`. See Chapter 3, “AMP File Format,” for information about the `.amp` format. `write` returns `True` if the operation is successful and returns `False` otherwise.

`dataformat` specifies the format of the data to be written. It must be one of the case-insensitive strings in the following table.

| Format | Description                                                         |
|--------|---------------------------------------------------------------------|
| 'DB'   | Data is given in (dB-magnitude, angle) pairs with angle in degrees. |
| 'MA'   | Data is given in (magnitude, angle) pairs with angle in degrees.    |
| 'RI'   | Data is given in (real, imaginary) pairs (default).                 |

`funit` specifies the frequency units of the data to be written. It must be `'GHz'`, `'MHz'`, `'KHz'` or `'Hz'`. If you do not specify `funit`, its value is taken from the object `data`. All values are case insensitive.

`printfmt` is a string that specifies the precision of the network and noise parameters. See the Precision specification for `fprintf`.

`freqformat` is a string that specifies the precision of the frequency. See the Precision specification for `fprintf`.

**References** [1] EIA/IBIS Open Forum, “Touchstone File Format Specification,” Rev. 1.1, 2002 ([http://www.eda.org/pub/ibis/connector/touchstone\\_spec11.pdf](http://www.eda.org/pub/ibis/connector/touchstone_spec11.pdf)).

**See Also** `read`, `rfckt`, `rfdata`



**Purpose** Convert Y-parameters to ABCD-parameters

**Syntax** `abcd_params = y2abcd(y_params)`

**Description** `abcd_params = y2abcd(y_params)` converts the admittance parameters `y_params` into the ABCD parameters `abcd_params`. The `y_params` input is a complex 2-by-2-by-`m` array, representing `m` two-port Y-parameters. `abcd_params` is a complex 2-by-2-by-`m` array, representing `m` two-port ABCD-parameters.

**See Also** `abcd2y`, `h2abcd`, `s2abcd`, `y2h`, `y2s`, `y2z`, `z2abcd`

# y2h

---

**Purpose** Convert Y-parameters to h-parameters

**Syntax** `h_params = y2h(y_params)`

**Description** `h_params = y2h(y_params)` converts the admittance parameters `y_params` into the hybrid parameters `h_params`. The `y_params` input is a complex 2-by-2-by-`m` array, representing `m` two-port Y-parameters. `h_params` is a complex 2-by-2-by-`m` array, representing `m` two-port h-parameters.

**See Also** `abcd2h`, `h2y`, `s2h`, `y2abcd`, `y2s`, `y2z`, `z2h`

**Purpose** Convert Y-parameters to S-parameters

**Syntax** `s_params = y2s(y_params, z0)`

**Description** `s_params = y2s(y_params, z0)` converts the admittance parameters `y_params` into the scattering parameters `s_params`. The `y_params` input is a complex `n`-by-`n`-by-`m` array, representing `m` `n`-port Y-parameters. `z0` is the reference impedance; its default is 50 ohms. `s_params` is a complex `n`-by-`n`-by-`m` array, representing `m` `n`-port S-parameters.

**See Also** `abcd2s`, `h2s`, `s2y`, `y2abcd`, `y2h`, `y2s`, `y2z`, `z2s`

# y2z

---

**Purpose** Convert Y-parameters to Z-parameters

**Syntax** `z_params = y2z(y_params)`

**Description** `z_params = y2z(y_params)` converts the admittance parameters `y_params` into the impedance parameters `z_params`. The `y_params` input is a complex `n`-by-`n`-by-`m` array, representing `m` `n`-port Y-parameters. `z_params` is a complex `n`-by-`n`-by-`m` array, representing `m` `n`-port Z-parameters.

**See Also** `abcd2z`, `h2z`, `y2abcd`, `y2h`, `y2s`, `y2z`, `z2s`, `z2y`

**Purpose** Convert Z-parameters to ABCD-parameters

**Syntax** `abcd_params = z2abcd(z_params)`

**Description** `abcd_params = z2abcd(z_params)` converts the impedance parameters `z_params` into the ABCD parameters `abcd_params`. The `z_params` input is a complex 2-by-2-by-`m` array, representing `m` two-port Z-parameters. `abcd_params` is a complex 2-by-2-by-`m` array, representing `m` two-port ABCD-parameters.

**See Also** `abcd2z`, `h2abcd`, `s2abcd`, `y2abcd`, `z2h`, `z2s`, `z2y`

# z2h

---

**Purpose** Convert Z-parameters to h-parameters

**Syntax** `h_params = z2h(z_params)`

**Description** `h_params = z2h(z_params)` converts the impedance parameters `z_params` into the hybrid parameters `h_params`. The `z_params` input is a complex 2-by-2-by-`m` array, representing `m` two-port Z-parameters. `h_params` is a complex 2-by-2-by-`m` array, representing `m` two-port h-parameters.

**See Also** `abcd2h`, `h2z`, `s2h`, `y2h`, `z2abcd`, `z2s`, `z2y`

**Purpose** Convert Z-parameters to S-parameters

**Syntax** `s_params = z2s(z_params, z0)`

**Description** `s_params = z2s(z_params, z0)` converts the impedance parameters `z_params` into the scattering parameters `s_params`. The `z_params` input is a complex `n`-by-`n`-by-`m` array, representing `m` `n`-port Z-parameters. `z0` is the reference impedance; its default is 50 ohms. `s_params` is a complex `n`-by-`n`-by-`m` array, representing `m` `n`-port S-parameters.

**See Also** `abcd2s`, `h2s`, `s2z`, `y2s`, `z2abcd`, `z2h`, `z2y`

# **z2y**

---

**Purpose** Convert Z-parameters to Y-parameters

**Syntax** `y_params = z2y(z_params)`

**Description** `y_params = z2y(z_params)` converts the impedance parameters `z_params` into the admittance parameters `y_params`. The `z_params` input is a complex `n`-by-`n`-by-`m` array, representing `m` `n`-port Z-parameters. `y_params` is a complex `n`-by-`n`-by-`m` array, representing `m` `n`-port Y-parameters.

**See Also** `abcd2y`, `h2y`, `s2y`, `y2z`, `z2abcd`, `z2h`, `z2s`



## A

- abcd2h function 5-8
- abcd2s function 5-9
- abcd2y function 5-10
- abcd2z function 5-11
- ABCD-parameters
  - converting to h-parameters 5-8
  - converting to S-parameters 5-9
  - converting to Y-parameters 5-10
  - converting to Z-parameters 5-11
- accessing
  - object properties 2-8
- AMP file format 3-1
  - comments 3-3
  - data sections 3-4
  - noise parameters 3-8
  - overview 3-2
  - power data 3-6
  - S, Y, Z Network Parameters 3-4
- amplifier
  - circuit analysis 5-36
  - properties 5-38
  - rfckt object 5-36
- analyze function 5-12

## C

- calculate function 5-14
  - example 5-14
- calculations
  - cascaded S-parameters 5-16
  - circuit analysis 5-12
  - input reflection coefficient 5-20
  - output reflection coefficient 5-21
  - specified network parameters 5-14
- cascade network
  - circuit analysis 5-39

- properties 5-41
- rfckt object 5-39
- cascadesparams function 5-16
- chart properties 5-127, 5-130
- circuit 5-12
- circuit analysis
  - amplifier 5-36
  - analyze function 5-12
  - cascade network 5-39
  - coaxial transmission line 5-42
  - coplanar waveguide transmission line 5-48
  - general circuit 5-53
  - general transmission line 5-110
  - hybrid network 5-56
  - LC bandpass pi filter 5-59
  - LC bandpass tee filter 5-61
  - LC bandstop pi filter 5-63
  - LC bandstop tee filter 5-65
  - LC highpass pi filter 5-67
  - LC highpass tee filter 5-69
  - LC lowpass pi filter 5-71
  - LC lowpass tee filter 5-73
  - microstrip transmission line 5-75
  - mixer 5-80
  - parallel network 5-84
  - parallel-plate transmission line 5-87
  - series network 5-93
  - series RLC filter 5-96
  - shunt RLC filter 5-100
  - two-wire transmission line 5-104
- circuit objects
  - constructing 5-32
  - constructing from a data file 5-53
  - copying 5-17
  - list of 5-32

- coaxial transmission line
    - circuit analysis 5-42
    - properties 5-45
    - rfckt object 5-42
    - shunt and series stubs 5-44
    - stubless 5-43
  - constructing new objects 2-3
  - conversion
    - ABCD-parameters to h-parameters 5-8
    - ABCD-parameters to S-parameters 5-9
    - ABCD-parameters to Y-parameters 5-10
    - ABCD-parameters to Z-parameters 5-11
    - h-parameters to ABCD-parameters 5-23
    - h-parameters to S-parameters 5-24
    - h-parameters to Y-parameters 5-25
    - h-parameters to Z-parameters 5-26
    - S-parameters to ABCD-parameters 5-121
    - S-parameters to h-parameters 5-122
    - S-parameters to S-parameters 5-123
    - S-parameters to T-parameters 5-124
    - S-parameters to Y-parameters 5-125
    - S-parameters to Z-parameters 5-126
    - T-parameters to S-parameters 5-132
    - Y-parameters to ABCD-parameters 5-135
    - Y-parameters to h-parameters 5-136
    - Y-parameters to S-parameters 5-137
    - Y-parameters to Z-parameters 5-138
    - Z-parameters to ABCD-parameters 5-139
    - Z-parameters to h-parameters 5-140
    - Z-parameters to S-parameters 5-141
    - Z-parameters to Y-parameters 5-142
  - coplanar waveguide transmission line
    - circuit analysis 5-48
    - properties 5-51
    - rfckt object 5-48
    - shunt and series stubs 5-49
    - stubless 5-49
  - copying objects 2-5
  - copythis function 5-17
- D**
- data I/O
    - constructing circuit object from a file 5-53
    - updating data object from a file 5-31
    - writing file from a data object 5-134
  - data objects
    - copying 5-17
    - corresponding to circuit object 5-22
    - list of 5-115
  - data visualization
    - polar plane 5-30
    - Smith chart from complex vector 5-130
    - Smith chart from object 5-127
    - using RF Tool 4-15
    - X-Y plane 5-29
  - datafile
    - rfckt object 5-53
  - deembedding
    - S-parameters
      - calculations
        - deembedding S-parameters 5-18
  - deembedsparams function 5-18
- E**
- extract function 5-19
  - extracting
    - network parameters 5-19
- F**
- file formats
    - AMP 3-1

## file formats (continued)

- supported 1-4

- See also.* AMP file format

## file I/O

- constructing circuit object from a file 5-53

- updating data object from a file 5-31

- writing file from a data object 5-134

## filter objects

- rfckt.lcbandpasspi 5-59

- rfckt.lcbandpasstee 5-61

- rfckt.lcbandstoppi 5-63

- rfckt.lcbandstoptee 5-65

- rfckt.lchighpasspi 5-67

- rfckt.lchighpasstee 5-69

- rfckt.lclowpasspi 5-71

- rfckt.lclowpasstee 5-73

- rfckt.seriesrlc 5-96

- rfckt.shuntrlc 5-100

## functions

- listed by category 5-2

- that act on objects 2-10

**G**

- gammain function 5-20

- gammaout function 5-21

## general circuits

- circuit analysis 5-53

- constructing from a data file 5-53

- properties 5-54

- rfckt object 5-53

## general transmission line

- circuit analysis 5-110

- properties 5-113

- rfckt object 5-110

- shunt and series stubs 5-111

- stubless 5-110

- getdata function 5-22

**H**

- h2abcd function 5-23

- h2s function 5-24

- h2y function 5-25

- h2z function 5-26

## h-parameters

- converting to ABCD-parameters 5-23

- converting to S-parameters 5-24

- converting to Y-parameters 5-25

- converting to Z-parameters 5-26

## hybrid network

- circuit analysis 5-56

- properties 5-57

- rfckt object 5-56

**I**

- input reflection coefficient

- calculating 5-20

**L**

- LC bandpass pi filter

- circuit analysis 5-59

- properties 5-60

- rfckt object 5-59

- LC bandpass tee filter

- circuit analysis 5-61

- properties 5-62

- rfckt object 5-61

- LC bandstop pi filter

- circuit analysis 5-63

- properties 5-64

- rfckt object 5-63

- LC bandstop tee filter
    - circuit analysis 5-65
    - properties 5-66
    - rfckt object 5-65
  - LC filter objects
    - rfckt.lcbandpasspi 5-59
    - rfckt.lcbandpasstee 5-61
    - rfckt.lcbandstoppi 5-63
    - rfckt.lcbandstoptee 5-65
    - rfckt.lchighpasspi 5-67
    - rfckt.lchighpasstee 5-69
    - rfckt.lclowpasspi 5-71
    - rfckt.lclowpasstee 5-73
  - LC highpass pi filter
    - circuit analysis 5-67
    - properties 5-68
    - rfckt object 5-67
  - LC highpass tee filter
    - circuit analysis 5-69
    - properties 5-70
    - rfckt object 5-69
  - LC lowpass pi filter
    - circuit analysis 5-71
    - properties 5-72
    - rfckt object 5-71
  - LC lowpass tee filter
    - circuit analysis 5-73
    - properties 5-74
    - rfckt object 5-73
  - listformat function 5-27
    - example 5-27
  - listparam function 5-28
    - example 5-28
- M**  
methods
- getting help 1-6
    - rfckt and rfddata objects 2-10
  - microstrip transmission line
    - circuit analysis 5-75
    - properties 5-78
    - rfckt object 5-75
    - shunt and series stubs 5-76
    - stubless 5-76
  - mixer
    - circuit analysis 5-80
    - properties 5-82
    - rfckt object 5-80
- N**
- network parameters
    - calculating 5-14
    - cascading 5-16
    - deembedding 5-18
    - extracting 5-19
    - listing valid formats 5-27
    - rfddata.data object 5-118
    - updating from a file 5-31
    - valid for a circuit or data object 5-28
    - writing to a file 5-134
- O**  
objects
- analysis 5-12
  - constructing new 2-3
  - copying 2-5
  - example using rfckt objects 2-12
  - functions that act on 2-10
  - getting help 1-6
  - introduction to their use 2-2
  - methods 2-10

objects (continued)

properties 2-6

output reflection coefficient

calculating 5-21

## P

parallel network

circuit analysis 5-84

properties 5-85

rfckt object 5-84

parallel-plate transmission line

circuit analysis 5-87

properties 5-90

rfckt object 5-87

shunt and series stubs 5-89

stubless 5-88

plot function 5-29

plots

polar plane 5-30

Smith chart from complex vector 5-130

Smith chart from object 5-127

using RF Tool 4-15

X-Y plane 5-29

polar function 5-30

properties of objects 2-6

retrieving 2-8

setting 2-6

## R

read function 5-31

reflection coefficient

calculating input 5-20

calculating output 5-21

retrieving

object properties 2-8

RF circuit objects

constructing 5-32

list of 5-32

RF data objects

constructing 5-115

list of 5-115

RF Tool

adding a component 4-5

adding a network 4-7

analyzing circuits 4-14

available components 4-2

available networks 4-3

deleting circuits 4-11

exporting RF objects 4-21

getting help 4-3

importing RF objects 4-18

opening the tool 4-4

overview of use 4-2

plotting network parameters 4-15

populating a network 4-8

reordering circuits 4-10

sessions 4-2

setting component parameters 4-13

working with sessions 4-24

rfckt function 5-32

example 5-34

rfckt methods

analyze 5-12

calculate 5-14

copy 5-17

getdata 5-22

listformat 5-27

listparam 5-28

plot 5-29

polar 5-30

smith 5-127

- rfckt objects
  - changing properties 5-34
  - viewing properties 5-34
- rfckt.amplifier object 5-36
  - circuit analysis 5-36
  - properties 5-38
- rfckt.cascade object 5-39
  - circuit analysis 5-39
  - properties 5-41
- rfckt.coaxial object 5-42
  - circuit analysis 5-42
  - properties 5-45
- rfckt.cpw object 5-48
  - circuit analysis 5-48
  - properties 5-51
- rfckt.datafile object 5-53
  - circuit analysis 5-53
  - properties 5-54
- rfckt.hybrid object 5-56
  - circuit analysis 5-56
  - properties 5-57
- rfckt.lcbandpasspi object 5-59
  - circuit analysis 5-59
  - properties 5-60
- rfckt.lcbandpasstee object 5-61
  - circuit analysis 5-61
  - properties 5-62
- rfckt.lcbandstoppi object 5-63
  - circuit analysis 5-63
  - properties 5-64
- rfckt.lcbandstoptee object 5-65
  - circuit analysis 5-65
  - properties 5-66
- rfckt.lchighpasspi object 5-67
  - circuit analysis 5-67
  - properties 5-68
- rfckt.lchighpasstee object 5-69
  - circuit analysis 5-69
  - properties 5-70
- rfckt.lclowpasspi object 5-71
  - circuit analysis 5-71
  - properties 5-72
- rfckt.lclowpasstee object 5-73
  - circuit analysis 5-73
  - properties 5-74
- rfckt.microstrip object 5-75
  - circuit analysis 5-75
  - properties 5-78
- rfckt.mixer object 5-80
  - circuit analysis 5-80
  - properties 5-82
- rfckt.parallel object 5-84
  - circuit analysis 5-84
  - properties 5-85
- rfckt.parallelplate object 5-87
  - circuit analysis 5-87
  - properties 5-90
- rfckt.series object 5-93
  - circuit analysis 5-93
  - properties 5-94
- rfckt.seriesrlc object 5-96
  - circuit analysis 5-96
  - properties 5-97
- rfckt.shuntrlc object 5-100
  - circuit analysis 5-100
  - properties 5-101
- rfckt.twowire object 5-104
  - circuit analysis 5-104
  - properties 5-107
- rfckt.txline object 5-110
  - circuit analysis 5-110
  - properties 5-113
- rfdata function 5-115
  - example 5-116

- rfddata methods
    - analyze 5-12
    - calculate 5-14
    - copy 5-17
    - extract 5-19
    - listformat 5-27
    - listparam 5-28
    - plot 5-29
    - polar 5-30
    - read 5-31
    - smith 5-127
    - write 5-134
  - rfddata objects
    - changing properties 5-116
    - viewing properties 5-116
  - rfddata.data object 5-118
    - properties 5-118
  - rftool function 5-120
  - RLC filter objects
    - rfckt.seriesRLC 5-96
    - rfckt.shuntRLC 5-100
    - properties 5-97
    - rfckt object 5-96
  - setting
    - object properties 2-6
  - shunt RLC filter
    - circuit analysis 5-100
    - example 5-101
    - properties 5-101
    - rfckt object 5-100
  - Smith chart from complex vector 5-130
  - Smith chart from object 5-127
  - smith function 5-127
  - smithchart function 5-130
  - S-parameters
    - changing impedance 5-123
    - converting to ABCD-parameters 5-121
    - converting to h-parameters 5-122
    - converting to S-parameters 5-123
    - converting to T-parameters 5-124
    - converting to Y-parameters 5-125
    - converting to Z-parameters 5-126
    - deembedding 5-18
  - stubless transmission lines
    - coaxial 5-43
    - coplanar waveguide 5-49
    - general 5-110
    - microstrip 5-76
    - parallel-plate 5-88
    - two-wire 5-105
  - stubs (shunt and series)
    - coaxial transmission line 5-44
    - coplanar waveguide transmission line 5-49
    - general transmission line 5-111
    - microstrip transmission line 5-76
    - parallel-plate transmission line 5-89
    - two-wire transmission line 5-106
- S**
- s2abcd function 5-121
  - s2h function 5-122
  - s2s function 5-123
  - s2t function 5-124
  - s2y function 5-125
  - s2z function 5-126
  - series network
    - circuit analysis 5-93
    - properties 5-94
    - rfckt object 5-93
  - series RLC filter
    - circuit analysis 5-96
    - example 5-97

**T**

- t2s function 5-132
- T-parameters
  - converting to S-parameters 5-132
- transmission line objects
  - coaxial 5-42
  - coplanar waveguide 5-48
  - general 5-110
  - microstrip 5-75
  - parallel-plate 5-87
  - two-wire 5-104
- two-wire transmission line
  - circuit analysis 5-104
  - properties 5-107
  - rfckt object 5-104
  - shunt and series stubs 5-106
  - stubless 5-105

**V**

- visualization
  - polar plane 5-30
  - Smith chart from complex vector 5-130
  - Smith chart from object 5-127
  - using RF Tool 4-15
  - X-Y plane 5-29
- voltage standing-wave ratio (VSWR)
  - calculating 5-133
- VSWR
  - calculating 5-133
- vswr function 5-133

**W**

- write function 5-134

**Y**

- y2abcd function 5-135
- y2h function 5-136
- y2s function 5-137
- y2z function 5-138
- Y-parameters
  - converting to ABCD-parameters 5-135
  - converting to h-parameters 5-136
  - converting to S-parameters 5-137
  - converting to Z-parameters 5-138

**Z**

- z2abcd function 5-139
- z2h function 5-140
- z2s function 5-141
- z2y function 5-142
- Z-parameters
  - converting to ABCD-parameters 5-139
  - converting to h-parameters 5-140
  - converting to S-parameters 5-141
  - converting to Y-parameters 5-142