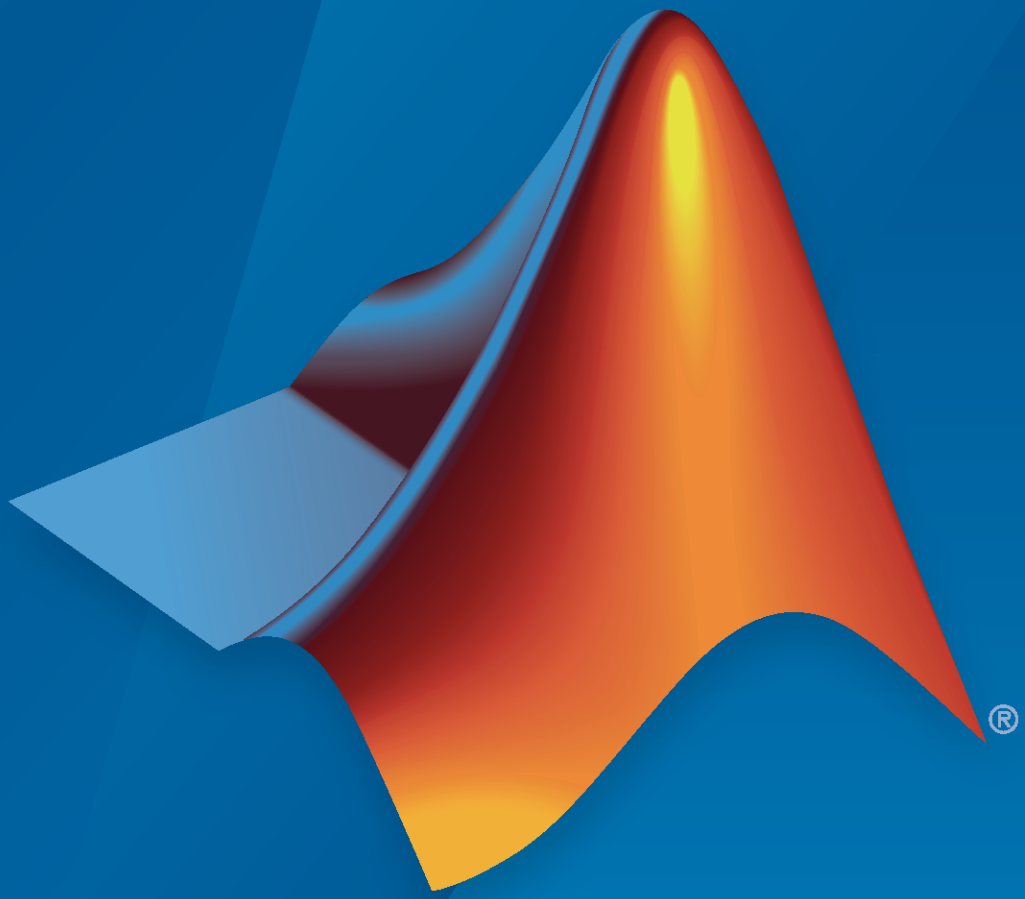
RF Toolbox™

Getting Started Guide



MATLAB®

R2022b

MathWorks®

How to Contact MathWorks

Latest news: `www.mathworks.com`

Sales and services: `www.mathworks.com/sales_and_services`

User community: `www.mathworks.com/matlabcentral`

Technical support: `www.mathworks.com/support/contact_us`

Phone: 508-647-7000

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

RF Toolbox™ Getting Started

Revision History

June 2004	Online only	New for Version 1.0 (Release 14)
August 2004	Online only	Revised for Version 1.0.1 (Release 14+)
March 2005	Online only	Revised for Version 1.1 (Release 14SP2)
September 2005	Online only	Revised for Version 1.2 (Release 14SP3)
March 2006	Online only	Revised for Version 1.3 (Release 2006a)
September 2006	Online only	Revised for Version 2.0 (Release 2006b)
March 2007	Online only	Revised for Version 2.1 (Release 2007a)
September 2007	Online only	Revised for Version 2.2 (Release 2007b)
March 2008	Online only	Revised for Version 2.3 (Release 2008a)
October 2008	Online only	Revised for Version 2.4 (Release 2008b)
March 2009	Online only	Revised for Version 2.5 (Release 2009a)
September 2009	Online only	Revised for Version 2.6 (Release 2009b)
March 2010	Online only	Revised for Version 2.7 (Release 2010a)
September 2010	Online only	Revised for Version 2.8 (Release 2010b)
April 2011	Online only	Revised for Version 2.8.1 (Release 2011a)
September 2011	Online only	Revised for Version 2.9 (Release 2011b)
March 2012	Online only	Revised for Version 2.10 (Release 2012a)
September 2012	Online only	Revised for Version 2.11 (Release 2012b)
March 2013	Online only	Revised for Version 2.12 (Release 2013a)
September 2013	Online only	Revised for Version 2.13 (Release 2013b)
March 2014	Online only	Revised for Version 2.14 (Release 2014a)
October 2014	Online only	Revised for Version 2.15 (Release 2014b)
March 2015	Online only	Revised for Version 2.16 (Release 2015a)
September 2015	Online only	Revised for Version 2.17 (Release 2015b)
March 2016	Online only	Revised for Version 3.0 (Release 2016a)
September 2016	Online only	Revised for Version 3.1 (Release 2016b)
March 2017	Online only	Revised for Version 3.2 (Release 2017a)
September 2017	Online only	Revised for Version 3.3 (Release 2017b)
March 2018	Online only	Revised for Version 3.4 (Release 2018a)
September 2018	Online only	Revised for Version 3.5 (Release 2018b)
March 2019	Online only	Revised for Version 3.6 (Release 2019a)
September 2019	Online only	Revised for Version 3.7 (Release 2019b)
March 2020	Online only	Revised for Version 3.8 (Release 2020a)
September 2020	Online only	Revised for Version 4.0 (Release 2020b)
March 2021	Online only	Revised for Version 4.1 (Release 2021a)
September 2021	Online only	Revised for Version 4.2 (Release 2021b)
March 2022	Online only	Revised for Version 4.3 (Release 2022a)
September 2022	Online only	Revised for Version 4.4 (Release 2022b)

Contents

1 Getting Started

Getting Started

RF Toolbox Product Description

Design, model, and analyze networks of RF components

RF Toolbox provides functions, objects, and apps for designing, modeling, analyzing, and visualizing networks of radio frequency (RF) components. The toolbox supports wireless communications, radar, and signal integrity projects.

RF Toolbox lets you build networks of RF components such as filters, transmission lines, matching networks, amplifiers, and mixers. Components can be specified using measurement data such as Touchstone files, network parameters, or physical properties. The toolbox provides functions for analyzing, manipulating, and visualizing RF data. You can analyze S-parameters; convert among S, Y, Z, T, and other network parameters; and visualize RF data using rectangular and polar plots and Smith® Charts. You can also de-embed, check, and enforce passivity, and compute group and phase delay.

The RF Budget Analyzer app lets you analyze transceiver chains in terms of noise, power, and nonlinearity and generate RF Blockset™ models for circuit envelope simulation. Using the rational function fitting method, you can model backplanes, interconnects, and linear components, and export them as Simulink® blocks, SPICE netlists, or Verilog®-A modules for time-domain simulation.

Key Features

- RF filters, transmission lines, amplifiers, and mixers specified by measurement data, network parameters, or physical properties
- S-parameter calculation for RF component networks
- RF Budget Analyzer app for calculating noise figure, gain, and IP3 of RF transceivers and for generating RF Blockset testbenches
- Rational function fitting method for building models and exporting them as Simulink blocks or Verilog-A modules
- De-embedding of N-port S-parameters measurement data
- Conversion among S, Y, Z, ABCD, h, g, and T network parameters
- RF data visualization using rectangular and polar plots and Smith Charts

Related Products

Several MathWorks® products are especially relevant to the kinds of tasks you can perform with RF Toolbox software. The following table summarizes the related products and describes how they complement the features of the toolbox.

Product	Description
“Communications Toolbox”	Simulink blocks and MATLAB® functions for time-domain simulation of modulation and demodulation of a wireless communications signal.
“DSP System Toolbox”	Simulink blocks and MATLAB functions for time-domain simulation of filtering the modulated communication signal.
“RF Blockset”	Circuit-envelope and equivalent-baseband simulation of RF components in Simulink.
“Signal Processing Toolbox”	MATLAB functions for filtering the modulated communication signal.

RF Objects

RF Toolbox software uses objects to represent RF components and networks. You create an object using the object's *constructor*. Every object has predefined fields called *properties*. The properties define the characteristics of the object. Each property associated with an object is assigned a value. Every object has a set of *methods*, which are operations that you can perform on the object. Methods are similar to functions except that they only act on an object.

The following table summarizes the types of objects that are available in the toolbox and describes the uses of each one. For more information on a particular type of object, including a list of the available objects and methods, follow the link in the table to the documentation for that object type.

Object Type	Name	Description
“RF Data Objects”	`rfdata`	Stores data for use by other RF objects or for plotting and network parameter conversion.
“RF Circuit Objects”	`rfckt`	Represents RF components and networks using network parameters and physical properties for frequency-domain simulation.
“RF Model Objects”	`rfmodel`	Represents RF components and networks mathematically for computing time-domain behavior and exporting models.

Each name in the preceding table is the prefix to the names of all object constructors of that type. The constructors use *dot notation* that consists of the object type, followed by a dot and then the component name. The component name is also called the *class*. For information on how to construct an RF object from the command line using dot notation, see “Create RF Objects”.

You use a different form of dot notation to specify object properties, as described in “Reference Properties Directly Using Dot Notation”. This is just one way to define component data. For more information on object properties, see “Specify or Import Component Data”.

You use object methods to perform frequency-domain analysis and visualize the results. For more information, see “Analyze and Plot RF Components”.

Note The toolbox also provides a graphical interface for creating and analyzing circuit objects. For more information, see “The RF Design and Analysis Tool”.

See Also

More About

- “RF Data Objects”
- “RF Circuit Objects”
- “RF Model Objects”

S-Parameter Notation

In this section...

Define S-Parameters

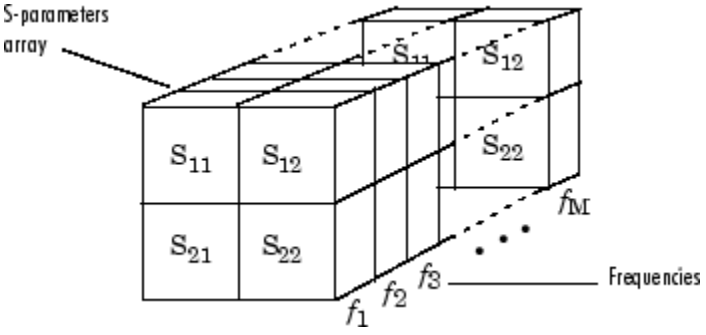
RF Toolbox software uses matrix notation to specify S-parameters. The indices of an S-parameter matrix correspond to the port numbers of the network that the data represent. For example, to define a matrix of 50-ohm, 2-port S-parameters, type:

```
s11 = 0.61*exp(j*165/180*pi);
s21 = 3.72*exp(j*59/180*pi);
s12 = 0.05*exp(j*42/180*pi);
s22 = 0.45*exp(j*(-48/180)*pi);
s_params = [s11 s12; s21 s22];
```

RF Toolbox functions that operate on `s_params` assume:

- `s_params(1,1)` corresponds to the reflection coefficient at port 1, S_{11}.
- `s_params(2,1)` corresponds to the transmission coefficient from port 1 to port 2, S_{21}.
- `s_params(1,2)` corresponds to the transmission coefficient from port 2 to port 1, S_{12}.
- `s_params(2,2)` corresponds to the reflection coefficient at port 2, S_{22}.

RF Toolbox software also supports three-dimensional arrays of S-parameters. The third dimension of an S-parameter array corresponds to S-parameter data at different frequencies. The following figure illustrates this convention.



Refer to S-Parameters Using Character Vector

RF Toolbox software uses character vector to refer to S-parameters in plotting and calculation methods, such as `plot`. These character vector have one of the following two forms:

- `'S`*nm*`'` — Use this syntax if *n* and *m* are both less than 10.
- `'S`*n*`,`*m*`'` — Use this syntax if one or both are greater than 10. `'S`*n*`,`*m*`'` is not a valid syntax when both *n* and *m* are less than 10.

The indices *n* and *m* are the port numbers for the S-parameters.

Most toolbox objects only analyze 2-port S-parameters. The following objects analyze S-parameters with more than two ports:

- `rfckt.passive`
- `rfckt.datafile`
- `rfdata.data`

You can get 2-port parameters from S-parameters with an arbitrary number of ports using one or more of the following steps:

- Extract 2-port S-parameters from N-port S-parameters.

  See “Extract M-Port S-Parameters from N-Port S-Parameters”.
- Convert single-ended 4-port parameters to differential 2-port parameters.

  See “Convert Single-Ended S-Parameters to Mixed-Mode S-Parameters”.

See Also

More About

- “RF Network Parameter Objects”
- “AMP File Data Sections”
- “Determining Parameter Formats”

RF Analysis

When you analyze an RF circuit using RF Toolbox, your workflow may include the following tasks:

1. Select RF circuit objects to represent the components of your RF network. To learn how to create RF objects, see “Create RF Objects”.
2. Define component data by:
   - Specifying network parameters or physical properties (see “Set Property Values”).
   - Importing data from an industry-standard Touchstone file, a MathWorks AMP file, an Agilent® P2D or S2D file, or the MATLAB workspace (see “Import Property Values from Data Files”).
   - Where applicable, selecting operating condition values (see “Specify Operating Conditions”).
3. Where applicable, perform network parameter conversions on imported file data. To understand network parameter conversion, see “Process File Data for Analysis”.
4. Integrate components to form a cascade, hybrid, parallel, or series network. To combine a set of RF components and existing networks to form an RF network, see “Construct Networks of Specified Components”.
5. Analyze the network in the frequency domain. To analyze your network, see “Analyze Networks in Frequency Domain”.
6. Generate plots to gain insight into network behavior.

   The following plots and charts are available in the toolbox:

   - Rectangular plots
   - Polar plots
   - Smith Chart
   - Budget plots (for cascaded S-parameters)

   To learn how to visualize your component and network data, see “Visualize Component and Network Data”.
7. Compute the network transfer function. To learn how to compute the network transfer function, see “Compute Network Transfer Function”.
8. Create an RF model object that describes the transfer function analytically. To analytically describe transfer function, see “Fit Model Object to Circuit Object Data”.
9. Plot the time-domain response of the transfer function of the RF model object. To compute and plot time-domain response of your RF model objects, see “Compute and Plot Time-Domain Response”.
10. Export a Verilog-A description of the network. To export Verilog-A description of the network, see “Export Verilog-A Model”.

You can also use **RF Budget Analyzer** app to:

- Build a cascade of RF elements.
- Calculate the per-stage and cascade output power, gain, noise figure, SNR, and IP3 of the system.
- Compute nonlinear effects such as output power, IP2, NF, and SNR using harmonic balance analysis.
- Plot `rfbudget` results across bandwidths and over stages.

- Plot S-parameters of the RF System on a Smith chart and a polar plot.
- Plot magnitude, phase and real, and imaginary parts of S-parameters of the RF System and over stages.
- Export per-stage and cascade values to the MATLAB workspace.
- Export the system design to RF Blockset for simulation.
- Export the system design to the RF Blockset Testbench as a device under test (DUT) subsystem and verify the results using simulation.
- Visualize budget results and S-parameters over stages and frequencies.
- Compare Friis and harmonic balance budget results.

To learn how to use **RF Budget Analyzer** app to build and analyze superhetrodyne receiver, see “Superheterodyne Receiver Using RF Budget Analyzer App”.

See Also

RF Budget Analyzer

More About

- “RF Data Objects”
- “RF Circuit Objects”
- “RF Model Objects”
- “RF Network Parameter Objects”

Model Cascaded Network

This example shows you how to model a cascaded network, analyze the network in the frequency domain, and plot the results. The network that you use in this example consists of an amplifier and two transmission lines.

Create RF Components

Create three circuit (`rfckt`) objects with the default property values. These circuit objects represent the two transmission lines and the amplifier.

```
FirstCkt = rfckt.txline;
SecondCkt = rfckt.amplifier;
ThirdCkt = rfckt.txline;
```

Specify Component Data

In this part of the example, you specify the transmission line and amplifier properties.

Transmission Line Properties

Set the line length of the first transmission line, `FirstCkt`, to `12`.

```
FirstCkt.LineLength = 12;
```

Set the line length of the second transmission line, `ThirdCkt`, to `0.025` and the phase velocity to `2.0e8`.

```
ThirdCkt.LineLength = 0.025;
ThirdCkt.PV = 2.0e8;
```

Amplifier Properties

Import network parameters, noise data, and power data from the `default.amp` file into the amplifier, `SecondCkt`.

```
read(SecondCkt,'default.amp');
```

Set the interpolation method of the amplifier, `SecondCkt`, to `cubic`.

```
SecondCkt.IntpType = 'cubic';
```

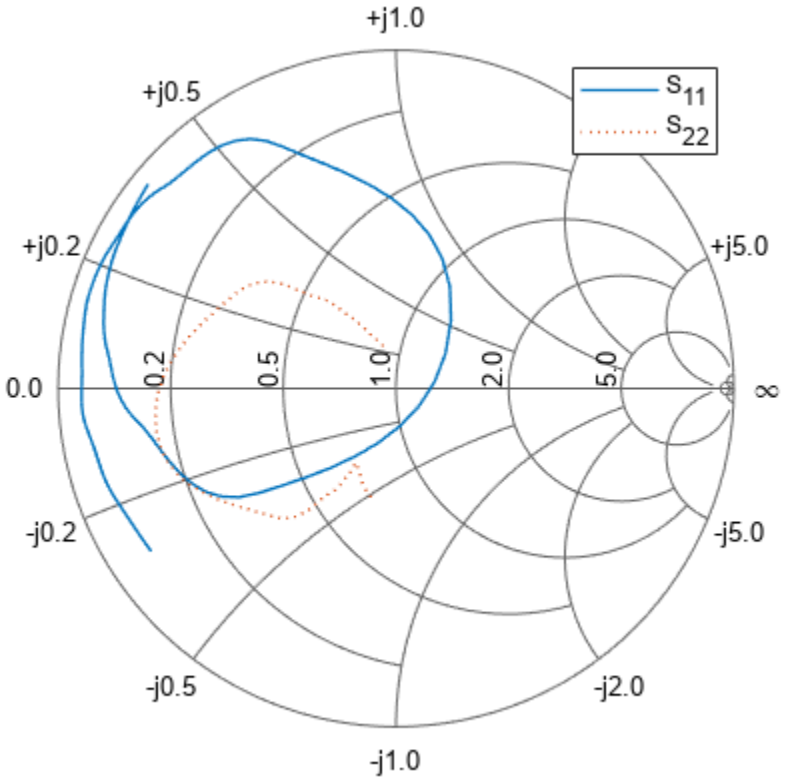
The `IntpType` property tells the RF toolbox how to interpolate the network parameters, noise data, and power data when you analyze the amplifier at frequencies other than those specified in the file.

Validate RF Components

In this part of the example, you plot the network parameters and power data (output power versus input power) to validate the behavior of the amplifier. Use the `smithplot` function to plot the original S11 and S22 parameters of the amplifier (`SecondCkt`) on a Z Smith® Chart.

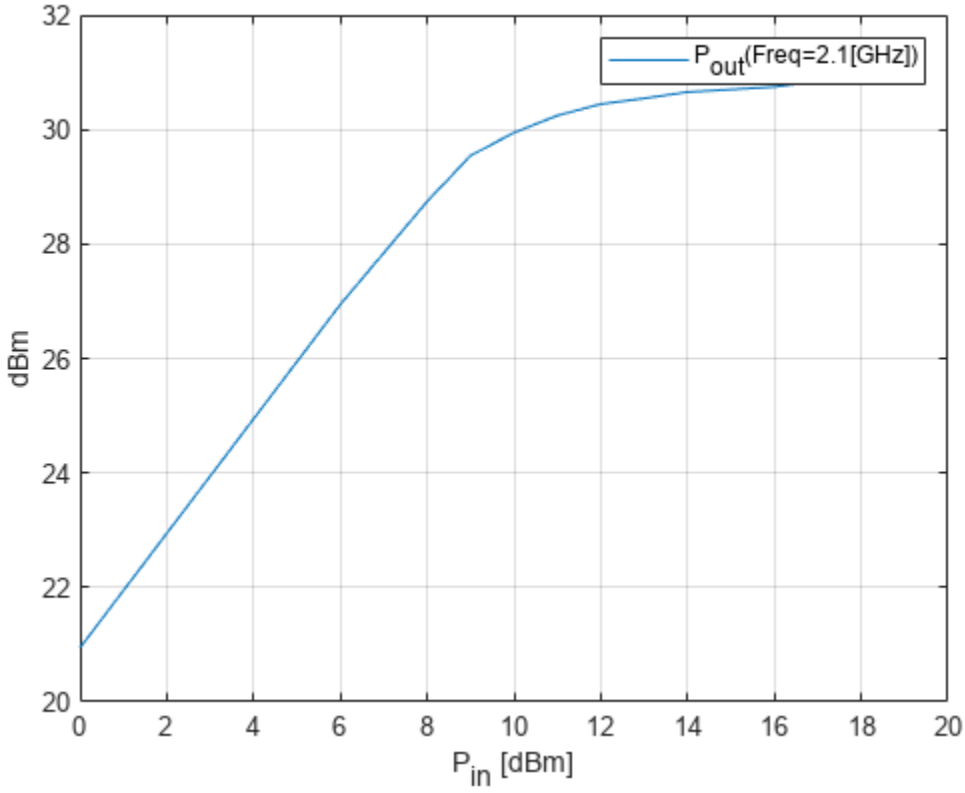
```
figure
legend show
lineseries1 = smith(SecondCkt,'S11','S22');
lineseries1(1).LineStyle = '-';
lineseries1(1).LineWidth = 1;
```

```
lineseries1(2).LineStyle = ':';
lineseries1(2).LineWidth = 1;
```



Plot the amplifier (`SecondCkt`) output power (Pout) as a function of input power (Pin), both in decibels referenced to one milliwatt (dBm), on an X-Y plane plot.

```
figure
legend show
plot(SecondCkt,'Pout','dBm')
```



Build and Simulate Network

In this part of the example, you create a circuit object to represent the cascaded amplifier and analyze the object in the frequency domain. Cascade the three circuit objects to form a new cascaded circuit object, `CascadedCkt`.

```
FirstCkt = rfckt.txline;
SecondCkt = rfckt.amplifier;
ThirdCkt = rfckt.txline;
CascadedCkt = rfckt.cascade('Ckts',{FirstCkt,SecondCkt,...
           ThirdCkt});
```

Define the range of frequencies over which to analyze the cascaded circuit, and then run the analysis.

```
f = (1.0e9:1e7:2.9e9);
analyze(CascadedCkt,f);
```

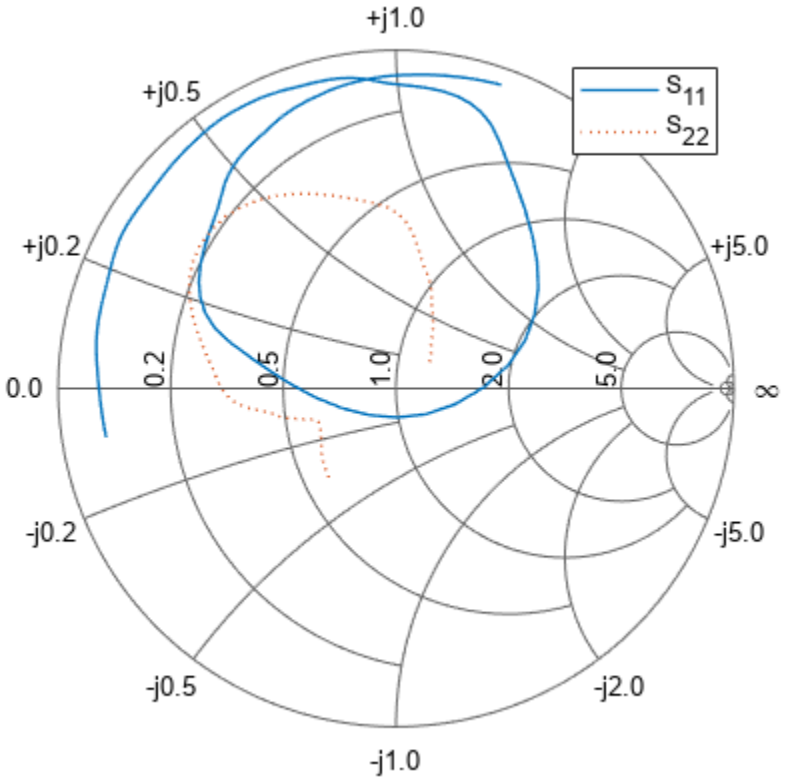
The plot shows the power data at 2.1 GHz because this frequency is the one for which power data is specified in the `default.amp` file.

Analyze Simulation Results

In this part of the example, you analyze the results of the simulation by plotting data for the circuit object that represents the cascaded amplifier network. Use the `smithplot` function to plot the S11 and S22 parameters of the cascaded amplifier network on a Z Smith Chart.

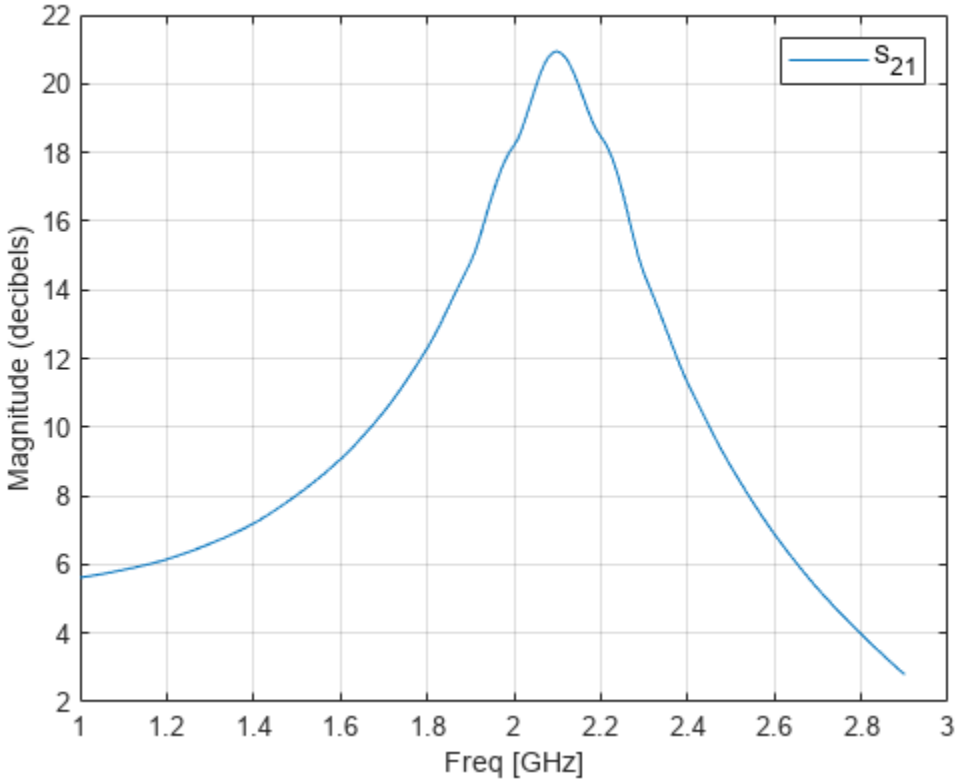
```
figure
legend show
```

```
lineseries2 = smith(CascadedCkt,'S11','S22','z');
lineseries2(1).LineStyle ='-';
lineseries2(1).LineWidth =1;
lineseries2(2).LineStyle = ':';
lineseries2(2).LineWidth = 1;
```



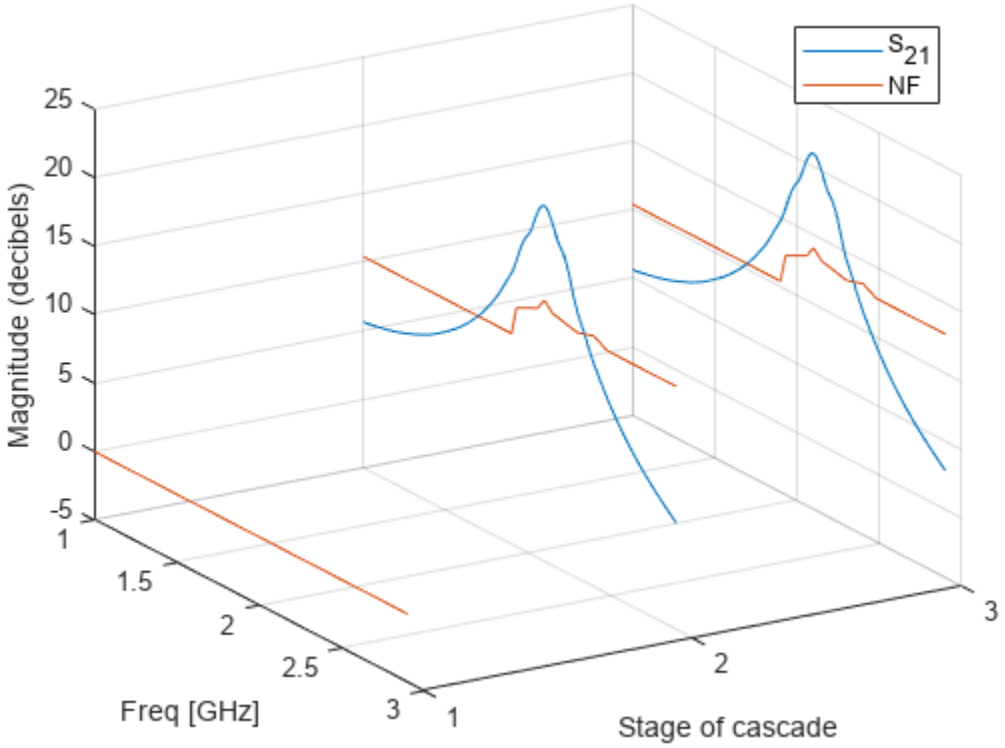
Use the `plot` function to plot the S21 parameter of the cascaded network, which represents the network gain, on an X-Y plane.

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



Use the `plot` function to create a budget plot of the S21 parameter and the noise figure of the amplifier network:

```
figure
legend show
plot(CascadedCkt,'budget', 'S21','NF')
```



The budget plot shows parameters as a function of frequency by circuit index. Components are indexed based on their position in the network. In this example:

- Circuit index one corresponds to `FirstCkt`.
- Circuit index two corresponds to `SecondCkt`.
- Circuit index three corresponds to `ThirdCkt`.

The curve for each index represents the contributions of the RF components up to and including the component at that index.

See Also

More About

- “Superheterodyne Receiver Using RF Budget Analyzer App”
- “Analyze Transmission Line” on page 1-15
- “Basic Operations with RF Objects”

Analyze Transmission Line

In this example, you use the RF Toolbox™ command-line interface to model the time-domain response of a parallel plate transmission line. You analyze the network in the frequency domain, compute and plot the time-domain response of the network, and export a Verilog-A model of the transmission line for use in system-level simulations.

Build and Simulate Transmission Line

Type the following command at the MATLAB® prompt to create a circuit (`rfckt`) object to represent the transmission line, which is 0.1 meters long and 0.05 meters wide:

```
tline = rfckt.parallelplate('LineLength',0.1,'Width',0.05);
```

Type the following set of commands at the MATLAB prompt to define the range of frequencies over which to analyze the transmission line and then run the analysis:

```
f = 1.0e9:1e7:2.9e9;
analyze(tline,f);
```

Compute Transmission Line Transfer Function and Time-Domain Response

This part of the example illustrates how to perform the following tasks:

- Calculate the Transfer Function
- Fit and Validate the Transfer Function Model
- Compute and Plot the Time-Domain Response

Calculate Transfer Function

Type the following command at the MATLAB prompt to extract the computed S-parameter values and the corresponding frequency values for the transmission line:

```
[S_Params, Freq] = extract(tline,'S_Parameters');
```

Type the following command at the MATLAB prompt to compute the transfer function from the frequency response data using the `s2tf` function:

```
TrFunc = s2tf(S_Params);
```

Fit and Validate Transfer Function Model

In this part of the example, you fit a rational function model to the transfer function. The toolbox stores the fitting results in an `rfmodel` object. You use the RF Toolbox `freqresp` method to validate the fit of the rational function model.

Type the following command at the MATLAB prompt to fit a rational function to the computed data and store the result in an `rfmodel` object:

```
RationalFunc = rationalfit(Freq,TrFunc)
```

```
RationalFunc =
   rfmodel.rational with properties:

        A: [7x1 double]
```

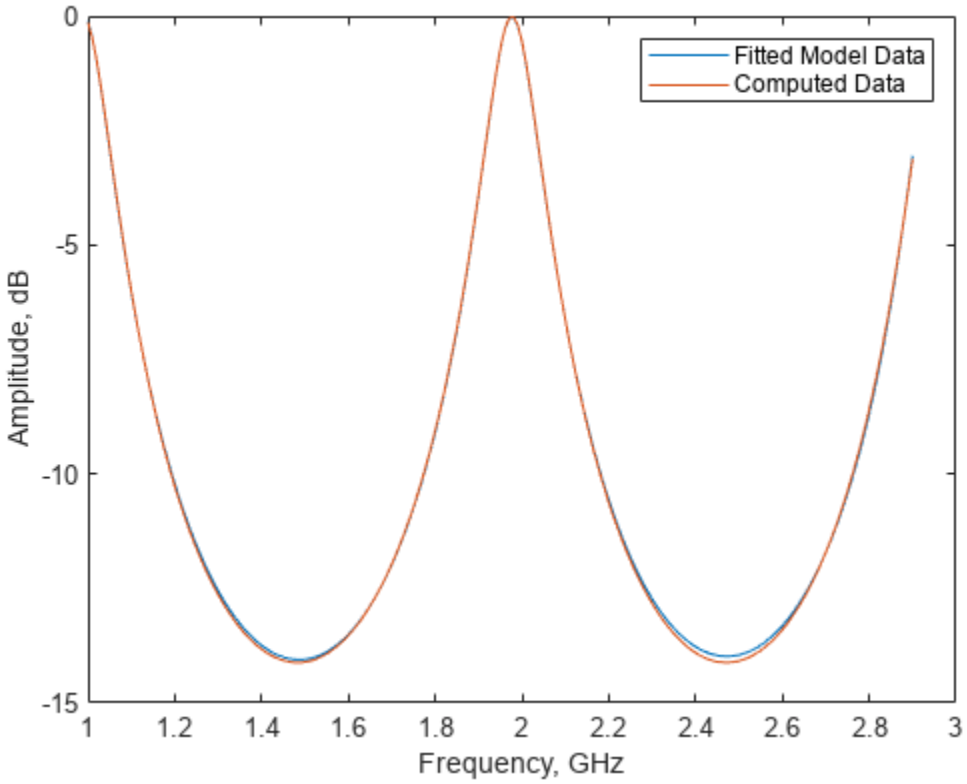
```
        C: [7x1 double]
        D: 0
    Delay: 0
     Name: 'Rational Function'
```

Type the following command at the MATLAB prompt to compute the frequency response of the fitted model data:

```
[fresp,freq] = freqresp(RationalFunc,Freq);
```

Type the following set of commands at the MATLAB prompt to plot the amplitude of the frequency response of the fitted model data and that of the computed data:

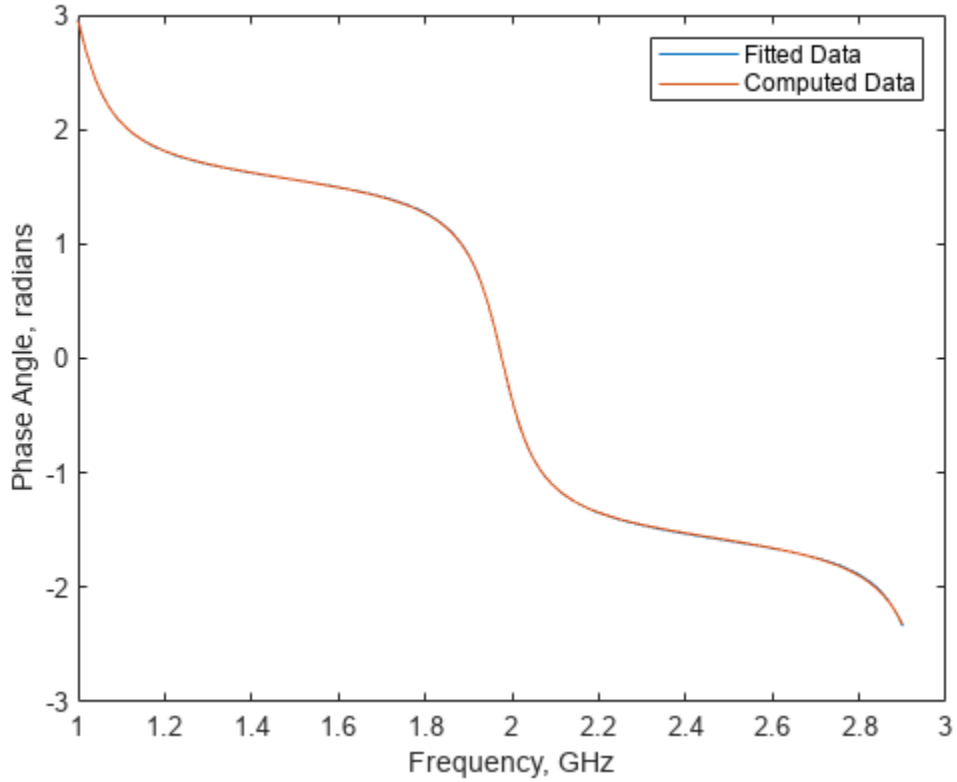
```
figure
plot(freq/1e9,20*log10(abs(fresp)),freq/1e9,20*log10(abs(TrFunc)))
xlabel('Frequency, GHz')
ylabel('Amplitude, dB')
legend('Fitted Model Data','Computed Data')
```



The amplitude of the model data is very close to the amplitude of the computed data. You can control the tradeoff between model accuracy and model complexity by specifying the optional tolerance argument, `tol`, to the `rationalfit` function, as described in “Export Verilog-A Model”.

Type the following set of commands at the MATLAB prompt to plot the phase angle of the frequency response of the fitted model data and that of the computed data:

```
figure
plot(freq/1e9,unwrap(angle(fresp)),...
    freq/1e9,unwrap(angle(TrFunc)))
xlabel('Frequency, GHz')
ylabel('Phase Angle, radians')
legend('Fitted Data','Computed Data')
```



The phase angle of the model data is very close to the phase angle of the computed data.

Compute and Plot Time-Domain Response

In this part of the example, you compute and plot the time-domain response of the transmission line.

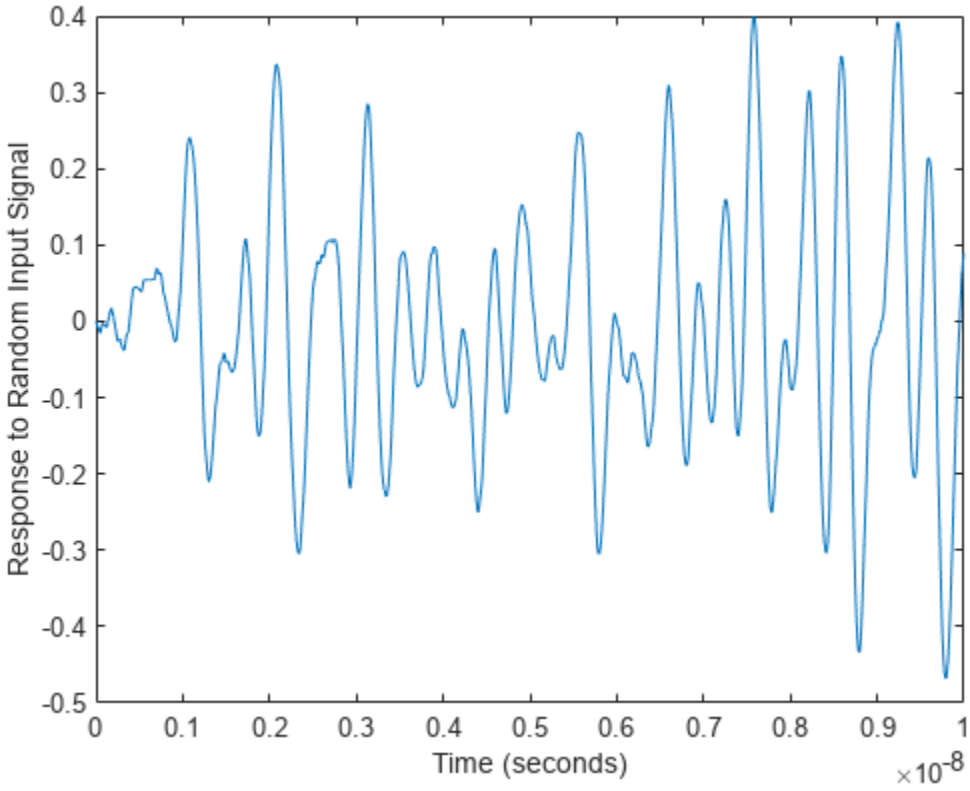
Type the following set of commands at the MATLAB prompt to create a random input signal and compute the time response, `tresp`, of the fitted model data to the input signal:

```
SampleTime = 1e-12;
NumberOfSamples = 1e4;
OverSamplingFactor = 25;
InputTime = double((1:NumberOfSamples)')*SampleTime;
InputSignal = ...
    sign(randn(1, ceil(NumberOfSamples/OverSamplingFactor)));
InputSignal = repmat(InputSignal, [OverSamplingFactor, 1]);
InputSignal = InputSignal(:);

[tresp,t] = timeresp(RationalFunc,InputSignal,SampleTime);
```

Type the following set of commands at the MATLAB prompt to plot the time response of the fitted model data:

```
figure
plot(t,tresp)
xlabel('Time (seconds)')
ylabel('Response to Random Input Signal')
```



Export Verilog-A Model

In this part of the example, you export a Verilog-A model of the transmission line. You can use this model in other simulation tools for detailed time-domain analysis and system simulations.

The following code illustrates how to use the `writeva` method to write a Verilog-A module for `RationalFunc` to the file `tline.va`. The module has one input, `tline_in`, and one output, `tline_out`. The method returns a `status` of `True`, if the operation is successful, and `False` if it is unsuccessful.

```
status = writeva(RationalFunc,'tline','tline_in','tline_out')
```

```
status = logical
   1
```

For more information on the `writeva` method and its arguments, see the `writeva` reference page.

See Also

More About

- “Superheterodyne Receiver Using RF Budget Analyzer App”
- “Model Cascaded Network” on page 1-9
- “Basic Operations with RF Objects”

RF circuit or rfbudget vs. rfckt Objects

This topic helps you to determine when to use RF `circuit`, `rfbudget`, and `rfckt` objects in your RF analysis workflow.

RF Toolbox provides two approaches to construct and analyze RF networks:

1. Networks built using elements based on the RF `circuit` and `rfbudget` objects.
2. Networks built with `rfckt` objects.

To understand when to use RF `circuit`, `rfbudget`, and `rfckt` objects, see this table.

Features	RF `circuit` or `rfbudget`	`rfckt` Objects
Construct networks with N-ports	Yes (RF `circuit`)	No
Import N-port S-parameters	Yes (RF `circuit`)	No
Matching network design	Yes (RF `circuit`)	Yes
Filter design	Yes	Yes
S-parameter analysis	Yes (N-port in RF `circuit`, two-port in `rfbudget`)	Yes (Two-port networks only)
Noise analysis	Yes (`rfbudget`)	Yes
Group delay analysis	Yes (RF `circuit`)	Yes
Nonlinearity analysis	Yes (`rfbudget` — Friis and Harmonic Balance methods)	Yes (Friis method)
AM-PM definitions of nonlinear amplifier	No	Yes
Compatibility	Compatible with RF Blockset Circuit Envelope library blocks (You can import and export `rfbudget` objects to Circuit Envelope library).	Compatible with RF Blockset Equivalent Baseband library blocks.

You can use this table as a reference when designing and analyzing your RF systems. For more on how to design an RF system using these objects, see:

- “RF System Design Using circuit or rfbudget Objects” on page 1-21
- “RF System Design Using rfckt Objects” on page 1-25

The RF chain shown in both examples is identical and provides the same result on analysis.

See Also

`rfbudget` | `circuit` | “RF Filter Design” | “RF Network Construction” | “RF Budget Analysis”

More About

- “RF Analysis” on page 1-7

RF System Design Using circuit or rfbudget Objects

Design an RF system using RF Toolbox™ `circuit` or `rfbudget` objects.

Create RF and IF bandpass filters using `nport` objects.

```
f1 = nport('RFBudget_RF.s2p','RFBandpassFilter');
f2 = nport('RFBudget_IF.s2p','IFBandpassFilter');
```

Create RF and IF amplifiers using the `amplifier` object and specifying the name, power gain, noise figure, and third-order intercept point properties.

```
a1 = amplifier('Name','RFAmplifier','Gain',12,'NF',2,'OIP3',35);
a2 = amplifier('Name','IFAmplifier','Gain',30,'NF',8,'OIP3',37);
```

Create a demodulator and microstrip transmission line using the `modulator` and `txlineMicrostrip` objects, respectively.

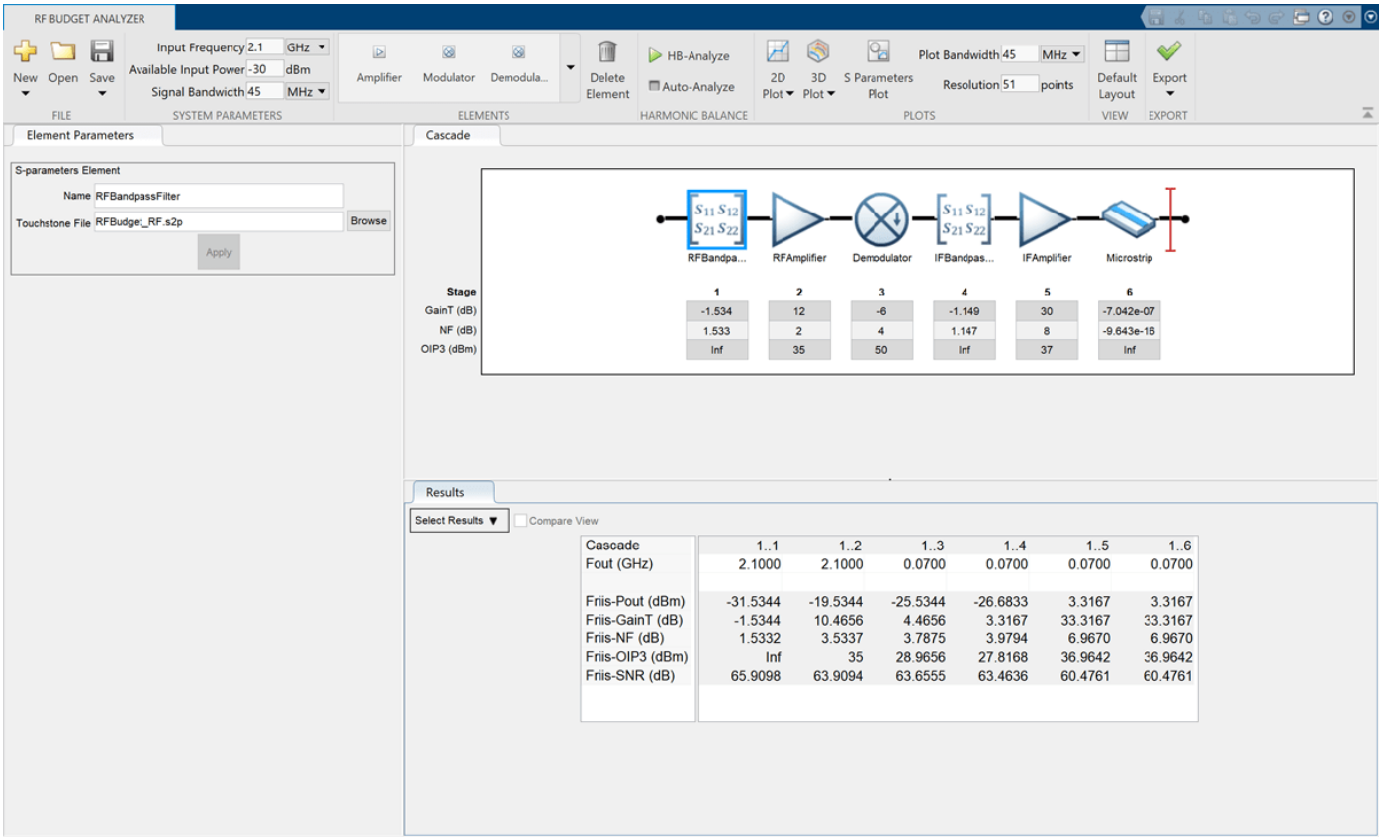
```
d = modulator('Name','Demodulator','Gain',-6,'NF',4,'OIP3',50, ...
        'LO',2.03e9,'ConverterType','Down');
tx = txlineMicrostrip('Thickness',0.0075e-6);
```

Create an `rfbudget` object and compute RF budget results for the chain of two-port elements. Calculate the RF budget of the RF elements at the specified input frequency, power, and signal bandwidth.

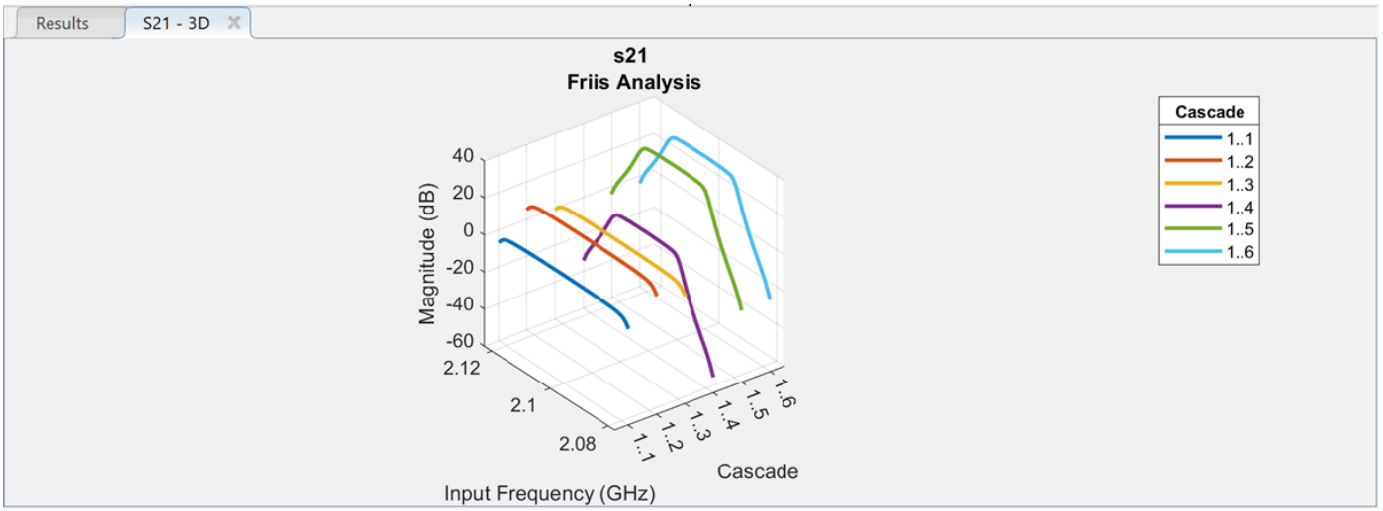
```
b = rfbudget([f1 a1 d f2 a2 tx],2.1e9,-30,45e6);
```

Type this command in the MATLAB® Command Window to analyze your circuit in the **RF Budget Analyzer** app.

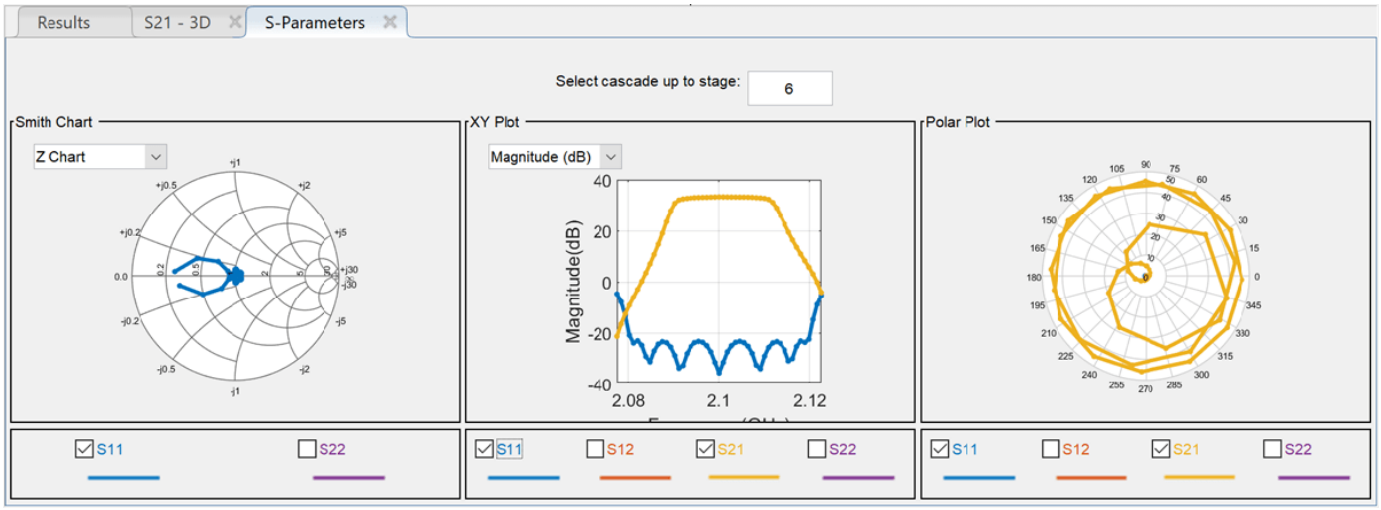
```
show(b)
```



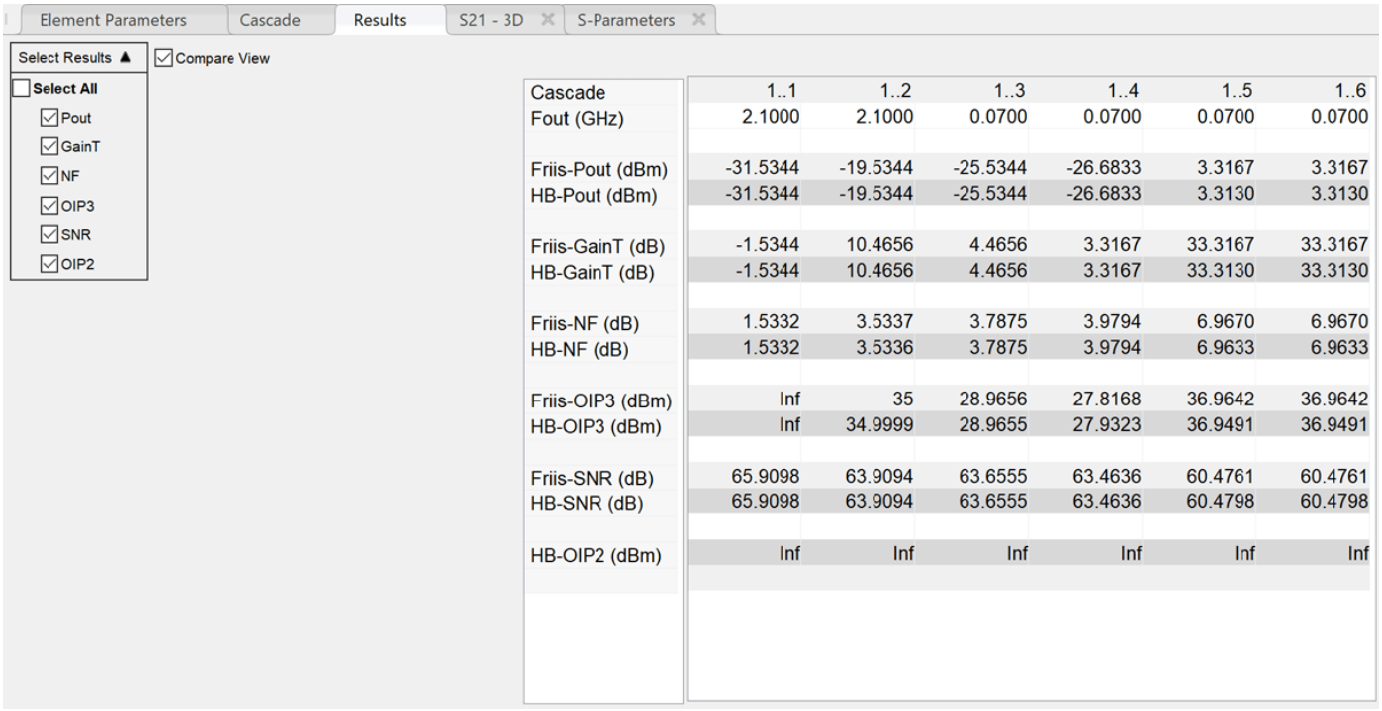
To plot the S21 parameter in the app, select **3D Plot** button in the **Plot** section and choose `S-Parameters` and then `S21`.



Select the **S-Parameters Plot** button. In the `S-Parameters` pane, type `6` in the **Select cascade up to stage** box. This allows you to plot Smith® chart, polar plot, magnitude, phase, real, and imaginary parts of the S-parameters of the RF System over stages. In the **XY Plot** pane, select the **S11** check box, and in the **Plot Plot** pane, select the **S21** check box. This displays the S11 and S21 in the XY plot and polar plot respectively.



To analyze your system using the harmonic balance solver, in the **Harmonic Balance** section, select the **HB-Analyze** button. Harmonic balance results are dispayed in the **Results** pane. To compare the results, select the **Compare View** check box.



Cascade	1..1	1..2	1..3	1..4	1..5	1..6
Fout (GHz)	2.1000	2.1000	0.0700	0.0700	0.0700	0.0700
Friis-Pout (dBm)	-31.5344	-19.5344	-25.5344	-26.6833	3.3167	3.3167
HB-Pout (dBm)	-31.5344	-19.5344	-25.5344	-26.6833	3.3130	3.3130
Friis-GainT (dB)	-1.5344	10.4656	4.4656	3.3167	33.3167	33.3167
HB-GainT (dB)	-1.5344	10.4656	4.4656	3.3167	33.3130	33.3130
Friis-NF (dB)	1.5332	3.5337	3.7875	3.9794	6.9670	6.9670
HB-NF (dB)	1.5332	3.5336	3.7875	3.9794	6.9633	6.9633
Friis-OIP3 (dBm)	Inf	35	28.9656	27.8168	36.9642	36.9642
HB-OIP3 (dBm)	Inf	34.9999	28.9655	27.9323	36.9491	36.9491
Friis-SNR (dB)	65.9098	63.9094	63.6555	63.4636	60.4761	60.4761
HB-SNR (dB)	65.9098	63.9094	63.6555	63.4636	60.4798	60.4798
HB-OIP2 (dBm)	Inf	Inf	Inf	Inf	Inf	Inf

Use **Select Results** list on from the **Results** pane to filter the results and to compare between Friis and harmonic balance solvers. For more information on the **RF Budget Analyzer** app, see RF Budget Analyzer.

See Also

“RF Filter Design” | “RF Network Construction” | “RF Budget Analysis”

More About

- “RF System Design Using rfckt Objects” on page 1-25
- “RF Analysis” on page 1-7
- “RF circuit or rfbudget vs. rfckt Objects” on page 1-20

RF System Design Using rfckt Objects

Design an RF system using RF Toolbox™ `rfckt` objects.

Create RF and IF bandpass filters using `rfckt.passive` objects.

```
fi1 = read(rfckt.passive,'RFBudget_RF.s2p');
fi2 = read(rfckt.passive,'RFBudget_IF.s2p');
```

Create RF and IF amplifiers by specifying an `rfdata.network` object as an input to the `rfckt.amplifier` object.

```
ai1 = rfckt.amplifier('NetworkData', ...
    rfdata.network('Type','S','Freq',2.1e9,'Data',[0,0;3.98,0]), ...
    'NoiseData',2,'NonlinearData',35);
ai2 = rfckt.amplifier('NetworkData', ...
    rfdata.network('Type','S','Freq',2.1e9,'Data',[0,0;31.66,0]), ...
    'NoiseData',8,'NonlinearData',37);
```

Create a demodulator and microstrip transmission line with the specified parameters using the `rfckt.mixer` and `rfckt.microstrip` objects, respectively. In this example, the `rfdata.network` object used in `rfckt.amplifier` and `rfckt.mixer` are used to set the gain of the amplifier and the mixer in linear scale.

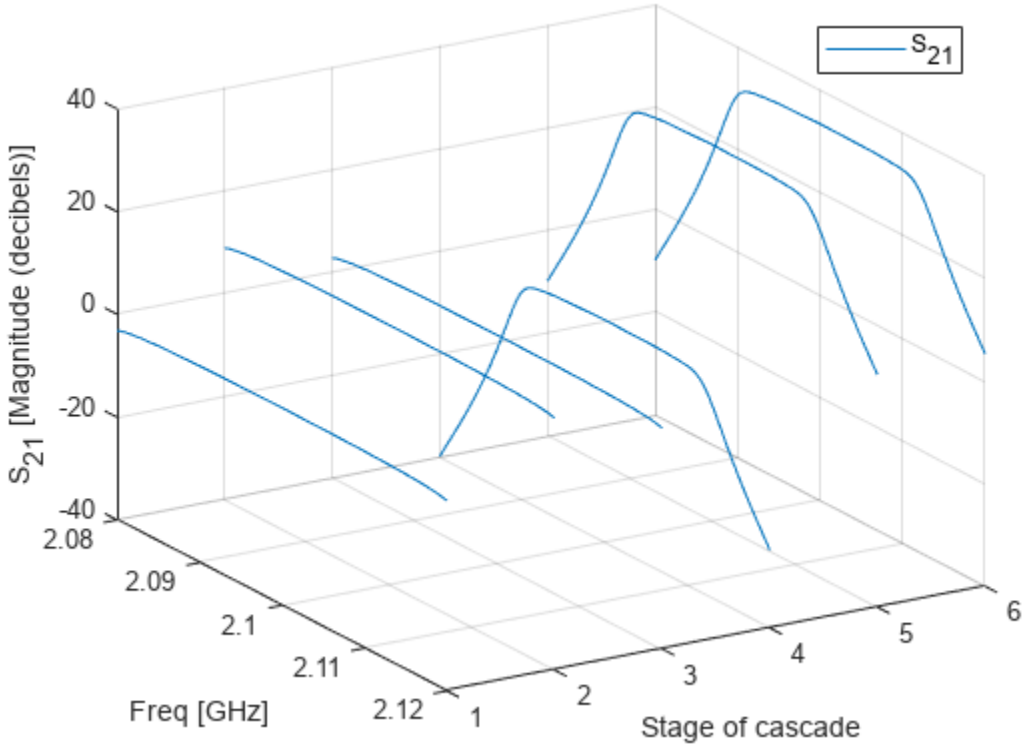
```
mi1 = rfckt.mixer('NetworkData', ...
    rfdata.network('Type','S','Freq',2.1e9,'Data',[0,0;0.501,0]),...
    "MixerType",'Downconverter','FLO',2.03e9,'NoiseData',4,'NonlinearData',50);
tx1 = rfckt.microstrip('Thickness',0.0075e-6);
```

Cascade the circuit using the `rfckt.cascade` object.

```
c = rfckt.cascade('Ckts',{fi1 ai1 mi1 fi2 ai2 tx1});
```

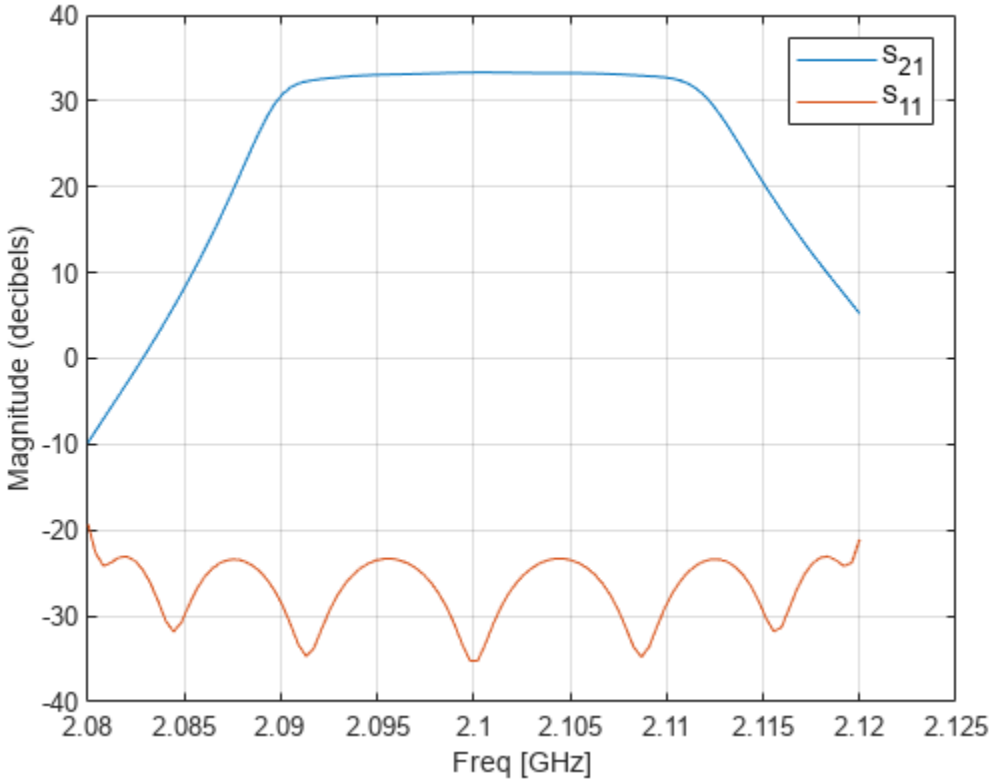
Analyze the cascaded circuit and plot the 3-D S21 plot.

```
analyze(c,linspace(2.08e9,2.12e9,100));
plot(c,'budget','s21')
```



Plot the magnitude of the S21 parameter for the cascade at stage 6.

```
plot(c,'s21','db')
hold on;
plot(c,'s11','db')
```



For more information on how to analyze and visualize RF components in the frequency-domain, see "Analyze and Plot RF Components".

See Also

"RF Filter Design" | "RF Network Construction" | "RF Budget Analysis"

More About

- "RF System Design Using circuit or rfbudget Objects" on page 1-21
- "RF circuit or rfbudget vs. rfckt Objects" on page 1-20
- "RF Analysis" on page 1-7