spfile

Release 0.1

Tuncay Erdöl

Jul 06, 2022

CONTENTS

CHAPTER

ONE

TOUCHSTONE MODULE

touchstone.**average_networks**(*networks*)

touchstone.**cascade_2ports**(*filenames*)

touchstone.**extract_gamma_ereff**(*filename_long_line*, *filename_short_line*, *dL*, *sm=1*)

Extraction of complex propagation constant (gamma) and complex effective permittivity from the S-Parameters of 2 uniform transmission lines with different lengths.

Parameters

- **filename_long_line** (*str*) – S-Parameter filename of longer line.
- **filename_short_line** (*str*) – S-Parameter filename of shorter line. If None is given, only long line is used for extraction.
- **dL** (*float*) – Difference of lengths of two lines (positive in meter). If filename_short_line is None, then this is the length of long line.
- **sm** (*int, optional*) – If this is larger than 1, this is used as number of points for smoothing. Defaults to 1.

Returns

tuple of two complex numpy arrays (gamma, er_eff).

Return type

tuple

touchstone.**extract_gamma_ereff_all**(*files*, *Ls*, *sm=1*)

Extraction of average complex propagation constant (gamma) and complex effective permittivity from the S-Parameters of multiple uniform transmission lines with different lengths.

Parameters

- **files** (*list*) – List of S-Parameter filenames of transmission lines.
- **Ls** (*list*) – List of lengths of transmission lines in the same order as *files* parameter.
- **sm** (*int, optional*) – If this is larger than 1, this is used as number of points for smoothing. Defaults to 1.

Returns

tuple of two complex numpy arrays (gamma, er_eff).

Return type

tuple

touchstone.**extract_rlgc**(*spr*, *length*)

Extraction of RLGC parameters from S-parameters of a uniform transmission line.

Parameters

- **spr** (*SPFILE*) – SPFILE object of two port network.
- **length** (*float*) – Length of transmission line.

Returns
tuple of two complex numpy arrays (Inductance per unit length, Characteristic impedance of the line).

Return type
tuple

touchstone.**generate_multiport_spfile**(*conf_filename*, *output_filename*)

Configuration file format: - comments start by "#" - every line's format is:

i,j ? filename ? is, js meaning: S(is,js) of touchstone file filename is S(i,j) of output_filename

Parameters

- **conf_filename** (*str*) – Name of the configuration filename.
- **output_filename** (*str*) – Name of the output filename.

touchstone.**parse_format**(*line*)

class touchstone.**spfile**(*filename=''*, *freqs=None*, *n_ports=1*, *skiplines=0*)

Bases: object

Class to process Touchstone files.

odo

TODO:

Extraction(*measspfile*)

Extract die S-Parameters using measurement data and simulated S-Parameters Port ordering in *measspfile* is assumed to be the same as this *spfile*. Remaining ports are ports of block to be extracted. See "Extracting multiport S-Parameters of chip" in technical document.

Parameters
measspfile (spfile) – *SPFILE* object of measured S-Parameters of first k ports

Returns
SPFILE object of die's S-Parameters

Return type
spfile

Ffunc(*imp*)

Calculates F-matrix in a, b definition of S-Parameters. For internal use of the library.

$$a = F(V + Z_r I)$$
$$b = F(V - Z_r^* I)$$

Parameters
imp (*numpy.ndarray*) – Zref, Reference impedance array for which includes the reference impedance for each port.

Returns
F-Matrix

Return type
numpy.matrix

ImpulseResponse(*i=2*, *j=1*, *dc_interp=1*, *dc_value=0.0*, *max_time_step=1.0*, *freq_res_coef=1.0*, *window_name='blackman'*)

Calculates impulse response of S_{ij}

Parameters

- **i** (*int, optional*) – Port-1. Defaults to 2.
- **j** (*int, optional*) – Port-2. Defaults to 1.
- **dc_interp** (*int, optional*) – If 1, add DC point to interpolation. Defaults to 1.
- **dc_value** (*float, optional*) – dc_value to be used at interpolation if *dc_interp=0*. Defaults to 0.0. This value is appended to S_{ij} and the rest is left to interpolation in *data_array* function.
- **max_time_step** (*float, optional*) – Not used for now. Defaults to 1.0.
- **freq_res_coef** (*float, optional*) – Coeeficient to increase the frequency resolution by interpolation. Defaults to 1.0 (no interpolation).
- **window** (*str, optional*) – Windows function to prevent ringing. Defaults to "blackman". Other windows will be added later.

Returns

The elements of the tuple are the following in order:

1. Raw frequency data used as input
2. Window array
3. Time array
4. Time-Domain Waveform of Impulse Response
5. Time-Domain Waveform of Impulse Input
6. Time step
7. Frequency step
8. Size of input array
9. Max Value of Impulse Input

Return type
9-tuple

ImpulseResponseBanded(*i=2*, *j=1*, *dc_interp=1*, *dc_value=0.0*, *max_time_step=1.0*, *freq_res_coef=1.0*, *Window='blackman'*)

Calculates impulse response of S_{ij}

Parameters

- **i** (*int, optional*) – Port-1. Defaults to 2.
- **j** (*int, optional*) – Port-2. Defaults to 1.
- **dc_interp** (*int, optional*) – If 1, add DC point to interpolation. Defaults to 1.

- **dc_value** (*float, optional*) – dc_value to be used at interpolation if *dc_interp=0*. Defaults to 0.0. This value is appended to S_{ij} and the rest is left to interpolation in *data_array* function.
- **max_time_step** (*float, optional*) – Not used for now. Defaults to 1.0.
- **freq_res_coef** (*float, optional*) – Coeeficient to increase the frequency resolution by interpolation. Defaults to 1.0 (no interpolation).
- **Window** (*str, optional*) – Windows function to prevent ringing. Defaults to "blackman". Other windows will be added later.

Returns

The elements of the tuple are the following in order:

1. Raw frequency data used as input
2. Window array
3. Time array
4. Time-Domain Waveform of Impulse Response
5. Time-Domain Waveform of Impulse Input
6. Time step
7. Frequency step
8. Size of input array
9. Max Value of Impulse Input

Return type
9-tuple

S(*i=1*, *j=1*, *data_format='COMPLEX'*, *freqs=None*)

Return S_{ij} in format *data_format* Uses *data_array* method internally. A convenience function for practical use.

Parameters

- **i** (*int, optional*) – Port-1. Defaults to 1.
- **j** (*int, optional*) – Port-2. Defaults to 1.
- **data_format** (*str, optional*) – See *data_format* parameter of *data_array* method. Defaults to "COMPLEX".

Returns
S_{ij} as *data_format*

Return type
numpy.array

T(*i=1*, *j=1*, *data_format='COMPLEX'*, *freqs=None*)

Return T_{ij} in format *data_format* Uses *data_array* method internally. A convenience function for practical use.

Parameters

- **i** (*int, optional*) – Port-1. Defaults to 1.
- **j** (*int, optional*) – Port-2. Defaults to 1.

- **data_format** (*str, optional*) – See *data_format* parameter of *data_array* method. Defaults to "COMPLEX".

Returns
T_{ij} as *data_format*

Return type
numpy.array

UniformDeembed(*quantity*, *ports='all'*, *kind='degrees'*, *inplace=- 1*)

This function deembeds some of the ports of S-Parameters. Deembedding quantity can be:

- Phase in degrees
- Phase in radians
- Length in meters
- Delay in seconds

A positive quantity means deembedding into the circuit. The Zc of de-embedding lines is the reference impedances of each port.

Parameters

- **quantity** (*float or list*) – Quantity to be deembedded. - If a number is given, it is used for all frequencies and ports - If a list is given, if its size is 1, its element is used for all ports. If its size is equal to number of ports, the list is used for all frequencies. Otherwise its size should be equal to the number of frequencies. If an element of list is number, it is used for all ports. If an element of the list is also a list, the elements size should be same as the number of ports.
- **ports** (*list, optional*) – List of port numbers to be deembedded. If not given all ports are deembedded.
- **kind** (*string, optional*) – One of the following values, "degrees", "radians", "length" and "delay". Defaults to "degrees".
- **inplace** (*int, optional*) – Object editing mode. Defaults to -1.

Returns
De-embedded spfile

Return type
spfile

Y(*i=1*, *j=1*, *data_format='COMPLEX'*, *freqs=None*)

Return Y_{ij} in format *data_format* Uses *data_array* method internally. A convenience function for practical use.

Parameters

- **i** (*int, optional*) – Port-1. Defaults to 1.
- **j** (*int, optional*) – Port-2. Defaults to 1.
- **data_format** (*str, optional*) – See *data_format* parameter of *data_array* method. Defaults to "COMPLEX".

Returns
Y_{ij} as *data_format*

Return type
numpy.array

Z(*i=1*, *j=1*, *data_format='COMPLEX'*, *freqs=None*)

Return Z_{ij} in format *data_format* Uses *data_array* method internally. A convenience function for practical use.

Parameters

- **i** (*int, optional*) – Port-1. Defaults to 1.
- **j** (*int, optional*) – Port-2. Defaults to 1.
- **data_format** (*str, optional*) – See *data_format* parameter of *data_array* method. Defaults to "COMPLEX".

Returns

Z_{ij} as *data_format*

Return type

numpy.array

Z_conjmatch(*port1=1*, *port2=2*)

Calculates source and load reflection coefficients for simultaneous conjugate match.

Parameters

- **port1** (*int, optional*) – [description]. Defaults to 1.
- **port2** (*int, optional*) – [description]. Defaults to 2.

Returns

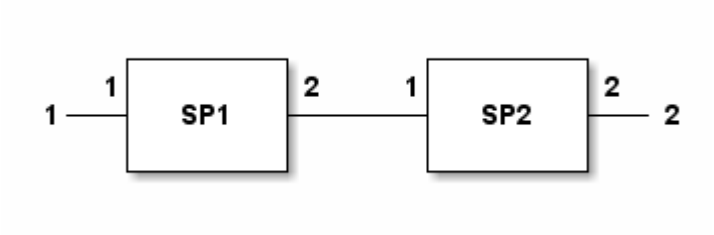
- GS: Reflection coefficient at Port-1
- GL: Reflection coefficient at Port-2

Return type

2-tuple of numpy.arrays (GS, GL)

__add__(*SP2*)

Implements SP1+SP2. Cascades port-1 of SP2 to port-2 of SP1. Port ordering is as follows:



SP1 is *self*.

Parameters

SP2 (spfile) – Appended spfile network

Returns

The result of cascade of 2 networks

Return type

spfile

__neg__()

Calculates an spfile object for two-port networks which is the inverse of this network. This is used to use + and - signs to cascade or deembed 2-port blocks.

Returns

1. *None* if number of ports is not 2.
2. *spfile* which is the inverse of the spfile object operated on.

Return type
spfile

`__sub__`(*SP2*)

Implements SP1-SP2. Deembeds SP2 from port-2 of SP1. Port ordering is as follows: (1)-SP1-(2)—(1)-SP2-(2) SP1 is *self*.

Parameters
SP2 (`spfile`) – Deembedded spfile network

Returns
The resulting of deembedding process

Return type
spfile

`add_abs_noise`(*dbnoise=0.1*, *phasenoise=0.1*, *inplace=- 1*)

This method adds random amplitude and phase noise to the s-parameter data. Mean value for both noises are 0.

Parameters

- **dbnoise** (*float, optional*) – Standard deviation of amplitude noise in dB. Defaults to 0.1.
- **phasenoise** (*float, optional*) – Standard deviation of phase noise in degrees. Defaults to 0.1.
- **inplace** (*int, optional*) – object editing mode. Defaults to -1.

Returns
object with noisy data

Return type
spfile

`calc_syz`(*input='S'*, *indices=None*)

This function calculates 2 of S, Y and Z parameters using the remaining parameter given. Y and Z-matrices calculated separately instead of calculating one and taking inverse. Because one of them may be undefined for some circuits.

Parameters

- **input** (*str, optional*) – Input parameter type (should be S, Y or Z). Defaults to "S".
- **indices** (*list, optional*) – If given, output matrices are calculated only at the indices given by this list. If it is None, then output matrices are calculated at all frequencies. Defaults to None.

`calc_t_eigs`(*port1=1*, *port2=2*)

Eigenfunctions and Eigenvector of T-Matrix is calculated. Only power-wave formulation is implemented

`change_formulation`(*formulation*)

`change_ref_impedance`(*Znewinput*, *inplace=- 1*)

Changes reference impedance and re-calculates S-Parameters.

Parameters
Znew (*float or list*) – New Reference Impedance. Its type can be: - float: In this case Znew value is used for all ports - list: In this case each element of this list is assgined to different ports in order as reference impedance. Length of *Znew* should be equal to number of ports. If an element of the list is None, then the reference impedance for corresponding port is not changed.

Returns
The spfile object with new reference impedance

Return type
spfile

check_passivity()

This method determines the frequencies and frequency indices at which the network is not passive. Reference: Fast Passivity Enforcement of S-Parameter Macromodels by Pole Perturbation.pdf For a better discussion: "S-Parameter Quality Metrics (Yuriy Shlepnev)"

Returns
For non-passive frequencies (indices, frequencies, eigenvalues)

Return type
3-tuple of lists

column_of_data(*i*, *j*)

Gets the indice of column at *sdata* matrix corresponding to S_{ij} For internal use of the library.

Parameters

- **i** (*int*) – First index
- **j** (*int*) – Second index

Returns
Index of column

Return type
int

conj_match_uncoupled(*ports=[]*, *inplace=- 1*, *noofiters=50*)

Sets the reference impedance for given ports as the complex conjugate of output impedance at each port. The ports are assumed to be uncoupled. Coupling is taken care of by doing the same operation multiple times.

Parameters

- **ports** (*list,optional*) – [description]. Defaults to all ports.
- **inplace** (*int, optional*) – Object editing mode. Defaults to -1.
- **noofiters** (*int, optional*) – Numberof iterations. Defaults to 50.

Returns
spfile object with new s-parameters

connect_2_ports(*k*, *m*, *inplace=- 1*)

Port-m is connected to port-k and both ports are removed. Reference: QUCS technical.pdf, S-parameters in CAE programs, p.29

Parameters

- **k** (*int*) – First port index to be connected.

- **m** (*int*) – Second port index to be connected.
- **inplace** (*int, optional*) – Object editing mode. Defaults to -1.

Returns
New spfile object

Return type
spfile

connect_2_ports_list(*conns*, *inplace=- 1*)

Short circuit ports together one-to-one. Short circuited ports are removed. Ports that will be connected are given as tuples in list *conns* i.e. conns=[(p1,p2),(p3,p4),..] The order of remaining ports is kept. Reference: QUCS technical.pdf, S-parameters in CAE programs, p.29

Parameters

- **conns** (*list of tuples*) – A list of 2-tuples of integers showing the ports connected
- **inplace** (*int, optional*) – Object editing mode. Defaults to -1.

Returns
New spfile object

Return type
spfile

connect_2_ports_retain(*k*, *m*, *inplace=- 1*)

Port-m is connected to port-k and both ports are removed. New port becomes the last port of the circuit. Reference: QUCS technical.pdf, S-parameters in CAE programs, p.29

Parameters

- **k** (*int*) – First port index to be connected.
- **m** (*int*) – Second port index to be connected.
- **inplace** (*int, optional*) – Object editing mode. Defaults to -1.

Returns
New *spfile* object

Return type
spfile

connect_network_1_conn(*EX*, *k*, *m*, *preserveportnumbers=False*, *inplace=- 1*)

Port-m of EX circuit is connected to port-k of this circuit. Both of these ports will be removed. Remaining ports of EX are added to the port list of this circuit in order. Reference: QUCS technical.pdf, S-parameters in CAE programs, p.29

Parameters

- **EX** (*spfile*) – External network to be connected to this.
- **k** (*int*) – Port number of self to be connected.
- **m** (*int*) – Port number of EX to be connected.
- **inplace** (*int, optional*) – Object editing mode. Defaults to -1.
- **preserveportnumbers** (*bool, optional*) – if True, the number of the first added port will be k. Defaults to False.

Returns
Connected network

Return type
spfile

connect_network_1_conn_retain(*EX*, *k*, *m*, *inplace=- 1*)

Port-m of EX circuit is connected to port-k of this circuit. This connection point will also be a port. Remaining ports of EX are added to the port list of this circuit in order. The port of connection point will be the last port of the final network. Reference: QUCS technical.pdf, S-parameters in CAE programs, p.29

Parameters

- **EX** (spfile) – External network to be connected to this.
- **k** (*int*) – Port number of self to be connected.
- **m** (*int*) – Port number of EX to be connected.
- **inplace** (*int, optional*) – Object editing mode. Defaults to -1.
- **preserveportnumbers1** (*bool, optional*) – if True, the number of the first added port will be k. Defaults to False.

Returns
Connected network

Return type
spfile

convert_s1p_to_s2p()

copy()

copy_data_from_spfile(*local_i*, *local_j*, *source_i*, *source_j*, *sourcespfile*)

This method copies S-Parameter data from another SPFILE object

classmethod cpwgline(*length*, *w*, *th*, *er*, *s*, *h*, *freqs=None*)

Create an spfile object corresponding to a cpwg transmission line.

Parameters

- **length** (*float*) – Length of cpwg line.
- **w** (*float*) – Width of cpwg line.
- **th** (*float*) – Thickness of metal.
- **er** (*float*) – Relative permittivity of substrate.
- **s** (*float*) – Gap of cpwg line.
- **h** (*float*) – Thickness of substrate.
- **freqs** (*float, optional*) – Frequency list of object. Defaults to None. If None, frequencies should be set later.

Returns
An spfile object.

Return type
spfile

crop_with_frequency(*fstart=None*, *fstop=None*, *inplace=- 1*)

Crop the points below fstart and above fstop. No recalculation or interpolation occurs.

Parameters

- **fstart** (*float, optional*) – Lower frequency for cropping. Default value is None which means no cropping will occur at lower frequency side.
- **fstop** (*float, optional*) – Higher frequency for cropping. Default value is None which means no cropping will occur at higher frequency side.
- **inplace** (*int, optional*) – Object editing mode. Defaults to -1.

Returns
spfile object with new frequency points.

Return type
spfile

data_array(*data_format='DB'*, *M='S'*, *i=1*, *j=1*, *frequencies=None*, *ref=None*, *DCInt=0*, *DCValue=0.0*, *smoothing=0*, *InterpolationConstant=0*)

Return a network parameter between ports *i* and *j* (M_{ij}) at specified frequencies in specified format.

Parameters

- **data_format** (*str, optional*) – Defaults to "DB". The format of the data returned. Possible values (case insensitive): - "K": Stability factor of 2-port - "MU1": Input stability factor of 2-port - "MU2": Output stability factor of 2-port - "VSWR": VSWR ar port i - "MAG": Magnitude of M_{ij} - "DB": Magnitude of M_{ij} in dB - "REAL": Real part of M_{ij} - "IMAG": Imaginary part of M_{ij} - "PHASE": Phase of M_{ij} in degrees between 0-360 - "UNWRAPPEDPHASE": Unwrapped Phase of M_{ij} in degrees - "GROUPDELAY": Group Delay of M_{ij} in degrees
- **M** (*str, optional*) – Defaults to "S". Possible values (case insensitive): - "S": Return S-parameter data - "Y": Return Y-parameter data - "Z": Return Z-parameter data - "ABCD": Return ABCD-parameter data
- **i** (*int, optional*) – First port number. Defaults to 1.
- **j** (*int, optional*) – Second port number. Defaults to 1. Ignored for *data_format*="VSWR"
- **frequencies** (*list, optional*) – Defaults to []. List of frequencies in Hz. If an empty list is given, networks whole frequency range is used.
- **ref** (*spfile, optional*) – Defaults to None. If given the data of this network is subtracted from the same data of *ref* object.
- **DCInt** (*int, optional*) – Defaults to 0. If 1, DC point given by *DCValue* is used at frequency interpolation if *frequencies* is not [].
- **DCValue** (*complex, optional*) – Defaults to 0.0. DCValue that can be used for interpolation over frequency.
- **smoothing** (*int, optional*) – Defaults to 0. if this is higher than 0, it is used as the number of points for smoothing.
- **InterpolationConstant** (*int, optional*) – Defaults to 0. If this is higher than 0, it is taken as the number of frequencies that will be added between 2 consecutive frequency points. By this way, number of frequencies is increased by interpolation.

Returns
Network data array

Return type
numpy.array

gav(*port1=1*, *port2=2*, *ZS=[]*, *dB=True*)

Available gain from port1 to port2. If dB=True, output is in dB, otherwise it is a power ratio.

$$G_{av} = \frac{P_{av,toLoad}}{P_{av,fromSource}}$$

Parameters

- **port1** (*int, optional*) – Index of input port. Defaults to 1.
- **port2** (*int, optional*) – Index of output port. Defaults to 2.
- **ZS** (*list or numpy.ndarray, optional*) – Impedance of input port. Defaults to current reference impedance.
- **dB** (*bool, optional*) – Enable dB output. Defaults to True.

Returns
Array of Gmax values for all frequencies

Return type
numpy.ndarray

get_formulation()

get_frequency_list()

Returns the frequency list of network

Returns
Frequency list of network

Return type
numpy.array

get_no_of_ports()

get_port_names()

Get list of port names.

get_port_number_from_name(*isim*)

Index of first port index with name *isim*

Parameters
isim (*bool*) – Name of the port

Returns
Port index if port is found, 0 otherwise

Return type
int

get_sym_parameters()

This function is used to get the values of symbolic variables of the network.

Returns
This is a dictionary containing the values of symbolic variables of the network

Return type
dict

get_sym_smatrix()

get_undefinedYindices()

get_undefinedZindices()

getdata_format()

getfilename()

gmax(*port1=1*, *port2=2*, *dB=True*)

Calculates Gmax from port1 to port2. Other ports are terminated with current reference impedances. If dB=True, output is in dB, otherwise it is a power ratio.

Parameters

- **port1** (*int, optional*) – Index of input port. Defaults to 1.
- **port2** (*int, optional*) – Index of output port. Defaults to 2.
- **dB** (*bool, optional*) – Enable dB output. Defaults to True.

Returns
Array of Gmax values for all frequencies

Return type
numpy.ndarray

gop(*port1=1*, *port2=2*, *ZL=None*, *dB=True*)

Operating power gain from port1 to port2 with load impedance of ZL. If dB=True, output is in dB, otherwise it is a power ratio.

$$G_{op} = \frac{P_{toLoad}}{P_{toNetwork}}$$

Parameters

- **port1** (*int, optional*) – Index of input port. Defaults to 1.
- **port2** (*int, optional*) – Index of output port. Defaults to 2.
- **ZL** (*ndarray or float, optional*) – Load impedance. Defaults to current port impedance at port2.
- **dB** (*bool, optional*) – Enable dB output. Defaults to True.

Returns
Array of Gop values for all frequencies

Return type
numpy.ndarray

gop2(*port1=1*, *port2=2*, *ZL=50.0*, *dB=True*)

Operating power gain from port1 to port2 with load impedance of ZL. If dB=True, output is in dB, otherwise it is a power ratio.

$$G_{op} = \frac{P_{toLoad}}{P_{toNetwork}}$$

Parameters

- **port1** (*int, optional*) – Index of input port. Defaults to 1.
- **port2** (*int, optional*) – Index of output port. Defaults to 2.
- **ZL** (*ndarray or float, optional*) – Load impedance. Defaults to current port impedance at port2.
- **dB** (*bool, optional*) – Enable dB output. Defaults to True.

Returns
Array of Gop values for all frequencies

Return type
numpy.ndarray

gt(*port1=1*, *port2=2*, *ZS=[]*, *ZL=[]*, *dB=True*)

This method calculates transducer gain (GT) from port1 to port2. Source and load impedances can be specified independently. If any one of them is not specified, current reference impedance is used for that port. Other ports are terminated by reference impedances. This calculation can also be done using impedance renormalization.

$$G_{av} = \frac{P_{load}}{P_{av,fromSource}}$$

Parameters

- **port1** (*int, optional*) – Index of source port. Defaults to 1.
- **port2** (*int, optional*) – Index of load port. Defaults to 2.
- **dB** (*bool, optional*) – Enable dB output. Defaults to True.
- **ZS** (*float, optional*) – Source impedance. Defaults to 50.0.
- **ZL** (*float, optional*) – Load impedance. Defaults to 50.0.

Returns
Array of GT values for all frequencies

Return type
numpy.ndarray

input_impedance(*k*)

Input impedance at port k. All ports are terminated with reference impedances.

Parameters
port (*int*) – Port number for input impedance.

Returns
Array of impedance values for all frequencies

Return type
numpy.ndarray

interpolate(*number_of_points=5*, *inplace=- 1*)

This method increases the number of frequencies through interpolation.

Parameters

- **number_of_points** (*int, optional*) – Number of points used for interpolation. Defaults to 5.

- **inplace** (*int, optional*) – object editing mode. Defaults to -1.

Returns
Network object with smooth data

Return type
spfile

interpolate_data(*datain*, *freqs*)

Calculate new data corresponding to new frequency points *freqs* by interpolation from original data corresponding to current frequency points of the network.

Parameters

- **data** (*numpy.ndarray or list*) – Original data specified at current frequency points of the network.
- **freqs** (*numpy.ndarray or list*) – New frequency list.

Returns
New data corresponding to *freqs*

Return type
numpy.ndarray

inverse_2port(*inplace=- 1*)

Take inverse of 2-port data for de-embedding purposes. The reference impedance of the network is not changed.

Parameters
inplace (*int, optional*) – Object editing mode. Defaults to -1.

Returns
Inverted 2-port spfile

Return type
spfile

load_impedance(*Gamma_in*, *port1=1*, *port2=2*)

Calculates termination impedance at port2 that gives Gamma_in reflection coefficient at port1.

Parameters

- **Gamma_in** (*float,ndarray*) – Required reflection coefficient.
- **port1** (*int*) – Source port.
- **port2** (*int*) – Load port.

Returns
Array of reflection coeeficient of termination at port2

Return type
numpy.ndarray

make_symmetric(*kind=1*, *inplace=- 1*)

Make SPFILE symmetric by taking the average of S11 and S22. S12=S21 assumed.

Parameters
inplace (*int, optional*) – Object editing mode. Defaults to -1.

Returns
Modified spfile object

Return type
spfile

`classmethod microstripline`(*length*, *w*, *h*, *t*, *er*, *freqs=None*)

Create an `spfile` object corresponding to a microstrip line.

Parameters

- **length** (*float*) – Length of microstrip line.
- **w** (*float*) – Width of microstrip line.
- **h** (*float*) – Thickness of substrate.
- **t** (*float*) – Thickness of metal.
- **er** (*float*) – Relative permittivity of microstrip substrate.
- **freqs** (*float, optional*) – Frequency list of object. Defaults to None. If None, frequencies should be set later.

Returns
An spfile object.

Return type
spfile

`classmethod microstripstep`(*w1*, *w2*, *eps_r*, *h*, *t*, *freqs=None*)

Create an `spfile` object corresponding to a microstrip step.

Parameters

- **w1** (*float*) – Width of microstrip line at port-1.
- **w2** (*float*) – Width of microstrip line at port-2.
- **t** (*float*) – Thickness of metal.
- **freqs** (*float, optional*) – Frequency list of object. Defaults to None. If None, frequencies should be set later.

Returns
An spfile object equivalent to microstrip step.

Return type
spfile

`prepare_ref_impedance_array`(*imparray=None*)

Turns reference impedance array which is composed of numbers,arrays, functions or 1-ports to numerical array which
is composed of numbers and arrays. It is made sure that :math:`Re(Z)

eq 0`. Mainly for internal use.

Args:
imparray (list): List of impedance array

Returns:
numpy.ndarray: Calculated impedance array

`read_file`(*file_name*, *skiplines=0*)

Network data is read from file. *filename* attribute of object is set with given argument.

Parameters

- **filename** (*str*) – Name of the file to be read. Its extension should be either "ts" of in the form of "sNp" or "sN".
- **skiplines** (*int, optional*) – This option is used if some beginning lines will be ignored. Default value is 0.

read_file_again()

Network data is read from the file named *filename*.

restore_passivity(*inplace=- 1*)

Make the network passive by minimum modification. Reference: Fast and Optimal Algorithms for Enforcing Reciprocity, Passivity and Causality in S-parameters.pdf

Parameters
inplace (*int, optional*) – Object editing mode. Defaults to -1.

Returns
Passive network object

Return type
spfile

restore_passivity2()

Obsolete Bu metod S-parametre datasinin pasif olmadigi frequenciesda S-parametre datasina mumkun olan en kucuk degisikligi yaparak S-parametre datasini pasif hale getirir. Referans: Restoration of Passivity In S-parameter Data of Microwave Measurements.pdf

return_s2p(*port1=1*, *port2=2*)

s2abcd(*port1=1*, *port2=2*)

S-Matrix to ABCD matrix conversion between 2 chosen ports. Other ports are terminated with reference impedances

Parameters

- **port1** (*int, optional*) – Index of Port-1. Defaults to 1.
- **port2** (*int, optional*) – Index of Port-2. Defaults to 2.

Returns
ABCD data. Numpy.matrix of size (ns,4) (ns: number of frequencies). Each row contains (A,B,C,D) numbers in order.

Return type
numpy.matrix

s2t()

Calculate transmission matrix for 2-port networks.

Returns
SPFILE object

Return type
spfile

scaledata(*scale=1.0*, *dataindices=None*)

classmethod series_impedance(*Z*, *freqs=None*)

Create an `spfile` object corresponding to a stripline step

Parameters

- **R** (*float*) – Shunt resistance.

- **freqs** (*float, optional*) – Frequency list of object. Defaults to None. If None, frequencies should be set later.

Returns
An spfile object.

Return type
spfile

set_formulation(*formulation*)

set_frequencies_wo_recalc(*freqs*)

Directly sets the frequencies of this network, but does not re-calculate s-parameters.

Parameters
freqs (*list or ndarray*) – New frequency values

set_frequency_limits(*flow*, *fhigh*, *inplace=- 1*)

Remove frequency points higher than *fhigh* and lower than *flow*.

Parameters

- **flow** (*float*) – Lowest Frequency (Hz)
- **fhigh** (*float*) – Highest Frequency (Hz)
- **inplace** (*int, optional*) – Object editing mode. Defaults to -1.

Returns
spfile object with new frequency points.

Return type
spfile

set_frequency_points(*frequencies*, *inplace=- 1*)

Set new frequency points. if S-Parameter data generator function is available, use that to calculate new s-parameter data. If not, use interpolation/extrapolation. For new frequency points, S-Parameters and reference impedances which are in the form of array are re-calculated.

Parameters

- **frequencies** (*list*) – New frequency array
- **inplace** (*int, optional*) – Object editing mode. Defaults to -1.

Returns
spfile object with new frequency points.

Return type
spfile

set_frequency_points_array(*fstart*, *fstop*, *NumberOfPoints*, *inplace=- 1*)

Set the frequencies of the object using start-end frequencies and number of points.

Parameters

- **fstart** (*[type]*) – Start frequency.
- **fstop** (*[type]*) – End frequency.
- **NumberOfPoints** (*int*) – Number of frequencies.
- **inplace** (*int, optional*) – Object editing mode. Defaults to -1.

Returns
spfile object with new frequency points.

Return type
spfile

`set_inplace`(*inplace*)

`set_port_name`(*name*, *i*)
Set name of a specific port.

Parameters

- **name** (*str*) – New name of the port
- **i** (*int*) – Port number

`set_port_names`(*names*)
Set port names with a list.

Parameters
names (*list*) – List of new names of the ports

`set_smatrix_at_frequency_point`(*indices*, *smatrix*)
Set S-Matrix at frequency indices

Parameters

- **indices** (*list*) – List of frequency indices
- **smatrix** (*numpy.matrix*) – New S-Matrix value which is to be set at all *indices*

`set_sparam_gen_func`(*func=None*)
This function is used to set the function that generates s-parameters from frequency.

Parameters
func (*function, optional*) – function to be set. Defaults to None.

`set_sparam_mod_func`(*func=None*)
This function is used to set the function that generates s-parameters from frequency.

Parameters
func (*function, optional*) – function to be set. Defaults to None.

`set_sym_parameters`(*paramdict*)
This function is used to set the values of symbolic variables of the network. This is used if the S-Matrix of the network is defined by an arithmetic expression containing symbolic variables. This property is used in conjunction with *sympy* library for symbolic manipulation. Arithmetic expression for S-Matrix is defined by `set_sym_smatrix` function.

Parameters
paramdict (*dict*) – This is a dictionary containing the values of symbolic variables of the network

`set_sym_smatrix`(*SM*)
This function is used to set arithmetic expression for S-Matrix, if S-Matrix is defined using symbolic variables.

Parameters
SM (*sympy.Matrix*) – Symbolic `sympy.Matrix` expression for S-Parameter matrix

`setdata_format`(*data_format*)

setdatapoint(*m*, *indices*, *x*)

Set the value for some part of S-Parameter data.

$$S_{ij}[m : m + len(x)] = x$$

Parameters

- **m** (*int*) – Starting frequency indice
- **indices** (*tuple of int*) – Parameters to be set (i,j)
- **x** (*number or list*) – New value. If this is a number, it is converted to a list.

classmethod shunt_impedance(*Z*, *freqs=None*)

Create an `spfile` object corresponding to a stripline step

Parameters

- **R** (*float*) – Shunt resistance.
- **freqs** (*float, optional*) – Frequency list of object. Defaults to None. If None, frequencies should be set later.

Returns
An spfile object.

Return type
spfile

smoothing(*smoothing_length=5*, *inplace=- 1*)

This method applies moving average smoothing to the s-parameter data

Parameters

- **smoothing_length** (*int, optional*) – Number of points used for smoothing. Defaults to 5.
- **inplace** (*int, optional*) – object editing mode. Defaults to -1.

Returns
Network object with smooth data

Return type
spfile

snp2smp(*ports*, *inplace=- 1*)

This method changes the port numbering of the network port j of new network corresponds to ports[j] in old network.

if the length of "ports" argument is lower than number of ports, remaining ports are terminated with current reference impedances and number of ports are reduced.

Parameters

- **ports** (*list*) – New port order
- **inplace** (*int, optional*) – Object editing mode. Defaults to -1.

Returns
Modified spfile object

Return type
spfile

`sqrt_network`()

Calculate the spfile, when two of which are cascaded, this spfile is obtained.

Returns
SPFILE object

Return type
spfile

`stability_factor_k`(*port1=1*, *port2=2*)

Calculates k stability factor, from port1 to port2. Other ports are terminated with reference impedances.

Parameters

- **port1** (*int, optional*) – Index of source port. Defaults to 1.
- **port2** (*int, optional*) – Index of load port. Defaults to 2.

Returns
Array of stability factor for all frequencies

Return type
numpy.ndarray

`stability_factor_mu1`(*port1=1*, *port2=2*)

Calculates μ_1 stability factor, from port1 to port2. Other ports are terminated with reference impedances.

Parameters

- **port1** (*int, optional*) – Index of source port. Defaults to 1.
- **port2** (*int, optional*) – Index of load port. Defaults to 2.

Returns
Array of stability factor for all frequencies

Return type
numpy.ndarray

`stability_factor_mu2`(*port1=1*, *port2=2*)

Calculates μ_2 stability factor, from port1 to port2. Other ports are terminated with reference impedances.

Parameters

- **port1** (*int, optional*) – Index of source port. Defaults to 1.
- **port2** (*int, optional*) – Index of load port. Defaults to 2.

Returns
Array of stability factor for all frequencies

Return type
numpy.ndarray

`classmethod stripline`(*length*, *w*, *er*, *h1*, *h2*, *t*, *freqs=None*)

Create an `spfile` object corresponding to a stripline transmission line.

Parameters

- **length** (*float*) – Length of cpwg line.
- **w** (*float*) – Width of stripline.

- **er** (*float*) – Relative permittivity of substrate.
- **h1** (*float*) – Thickness of substrate from bottom ground to bottom of line.
- **h2** (*float*) – Thickness of substrate from top line to top ground.
- **t** (*float*) – Thickness of metal.
- **freqs** (*float, optional*) – Frequency list of object. Defaults to None. If None, frequencies should be set later.

Returns
An spfile object.

Return type
spfile

classmethod striplinestep(*w1*, *w2*, *eps_r*, *h1*, *h2*, *t*, *freqs=None*)

Create an spfile object corresponding to a stripline step

Parameters

- **w1** (*float*) – Width of stripline line at port-1.
- **w2** (*float*) – Width of stripline line at port-2.
- **eps_r** (*float*) – Relative permittivity of stripline substrate.
- **h** (*float*) – Thickness of stripline substrate.
- **t** (*float*) – Thickness of metal.
- **freqs** (*float, optional*) – Frequency list of object. Defaults to None. If None, frequencies should be set later.

Returns
An spfile object.

Return type
spfile

write2file(*filename=''*, *parameter='S'*, *freq_unit=''*, *data_format=''*, *normalized=True*)

This function writes a parameter (S, Y or Z) file. If the filename given does not have the proper filename extension, it is corrected.

Parameters

- **filename** (*str, optional*) – Filename to be written. Defaults to "".
- **parameter** (*str, optional*) – Parameter to be written (S, Y or Z). Defaults to "S".
- **freq_unit** (*str, optional*) – Frequency unit (GHz, MHz, kHz or Hz). Defaults to "Hz".
- **data_format** (*str, optional*) – Format of file DB, RI or MA. Defaults to "".

touchstone.**thru_line_deembedding**(*thru_filename*, *line_filename*, *make_symmetric=True*)

Extraction of transition s-parameters from THRU and LINE measurements. Transitions on both sides are assumed to be identical. For output *spfile* objects, port-1 is launcher side and port-2 is transmission line side. The length difference between LINE and THRU should be ideally $\lambda/4$. The reference impedance for the 2. port of the transition should be the same as the characteristic impedance of the interconnecting line. So the reference impedances of the output *spfile* should be adjusted (without renormalizing s-parameters) after calling this function. The minimum frequency in the S-Parameter files should be such that the phase difference between the measurements should be smaller than 2:math:*pi*.

Parameters

- **thru_filename** (*str*) – 2-Port S-Parameter filename of THRU measurement
- **line_filename** (*str*) – 2-Port S-Parameter filename of LINE measurement

Returns
2-Element tuple of (transition spfile, complex phase vector ($-\gamma l$) of connecting line of LINE standard (in radian))

Return type
tuple(*spfile*, numpy.ndarray)

touchstone.**trl_launcher_extraction**(*thru_file*, *line_file*, *reflect_file*, *refstd=False*)

Extraction of launcher s-parameters by THRU, LINE, REFLECT calibration. For both output *spfile* objects, port-1 is launcher side and port-2 is transmission line side. Reference: TRL algorithm to de-embed a RF test fixture.pdf (Note that the T-Matrix definiton in the reference document is different than this library.)

Parameters

- **thru_file** (*str*) – 2-Port S-Parameter filename or *spfile* of THRU measurement
- **line_file** (*str*) – 2-Port S-Parameter filename or *spfile* of LINE measurement
- **reflect_file** (*str*) – 2-Port S-Parameter filename or *spfile* of REFLECT measurement
- **refstd** (*boolean*) – True if OPEN is used as REFLECT standard and False (default) if SHORT is used

Returns
3-Element tuple of (left side launcher spfile, right side launcher spfile, positive phase vector of connecting line of LINE standard (in radian))

Return type
tuple(*spfile*, *spfile*, numpy.ndarray)

touchstone.**untermination_method**(*g1*, *g2*, *g3*, *gL1*, *gL2*, *gL3*, *returnS2P=False*, *freqs=None*)

Determination of S_{11}, S_{22} and $S_{21} = S_{12}$ for a 2-port network network using 3 reflection coefficient values at port-1 for 3 terminations at port-2. S_{21} can only be calculated with a sign ambiguity because it exists only as square in the formulae.

Port-1: Input port. Port-2: Output port where load impedances are switched.

Parameters

- **g1** (*float, complex or ndarray*) – Reflection coefficient at port-1 when port-2 is terminated by a load with reflection coefficient gL1
- **g2** (*float, complex or ndarray*) – Reflection coefficient at port-1 when port-2 is terminated by a load with reflection coefficient gL2
- **g3** (*float, complex or ndarray*) – Reflection coefficient at port-1 when port-2 is terminated by a load with reflection coefficient gL3
- **gL1** (*float, complex or ndarray*) – Reflection coefficient of load at port-2 that gives g1 reflection coefficient at port-1
- **gL2** (*float, complex or ndarray*) – Reflection coefficient of load at port-2 that gives g2 reflection coefficient at port-1
- **gL3** (*float, complex or ndarray*) – Reflection coefficient of load at port-2 that gives g3 reflection coefficient at port-1

- **returnS2P** (*boolean*) – If True, function returns an *spfile* object of the 2-port network, if False, it returns 3-tuple of S-Parameter arrays. Default is False.
- **freqs** (*numpy.ndarray, list*) – If returnS2P is True, this input is used as the frequency points of the returned *spfile* object. Default is None.

Returns
Either 3-Element tuple of (S11, S22, S21) or *spfile* object, depending on returnS2P input.

Return type
tuple

touchstone.**untermination_method_old**(*g1*, *g2*, *g3*, *gL1*, *gL2*, *gL3*, *returnS2P=False*, *freqs=None*)

Determination of S_{11}, S_{22} and $S_{21} = S_{12}$ for a 2-port network network using 3 reflection coefficient values at port-1 for 3 terminations at port-2. S_{21} can only be calculated with a sign ambiguity because it exists only as square in the formulae.

Port-1: Input port. Port-2: Output port where load impedances are switched.

Parameters

- **g1** (*float, complex or ndarray*) – Reflection coefficient at port-1 when port-2 is terminated by a load with reflection coefficient gL1
- **g2** (*float, complex or ndarray*) – Reflection coefficient at port-1 when port-2 is terminated by a load with reflection coefficient gL2
- **g3** (*float, complex or ndarray*) – Reflection coefficient at port-1 when port-2 is terminated by a load with reflection coefficient gL3
- **gL1** (*float, complex or ndarray*) – Reflection coefficient of load at port-2 that gives g1 reflection coefficient at port-1
- **gL2** (*float, complex or ndarray*) – Reflection coefficient of load at port-2 that gives g2 reflection coefficient at port-1
- **gL3** (*float, complex or ndarray*) – Reflection coefficient of load at port-2 that gives g3 reflection coefficient at port-1
- **returnS2P** (*boolean*) – If True, function returns an *spfile* object of the 2-port network, if False, it returns 3-tuple of S-Parameter arrays. Default is False.
- **freqs** (*numpy.ndarray, list*) – If returnS2P is True, this input is used as the frequency points of the returned *spfile* object. Default is None.

Returns
Either 3-Element tuple of (S11, S22, S21) or *spfile* object, depending on returnS2P input.

Return type
tuple

touchstone.**write_impedance_as_s1p**(*filename*, *frequencies*, *Z*)

touchstone.**write_impedance_as_table**(*filename*, *frequencies*, *Z*)

CHAPTER

TWO

COMPONENTS MODULE

Created on Tue Nov 17 11:52:33 2009

@author: Tuncay

components.**AWG2Dia**(*arg*, *defaultunits=[]*)

Convert AWG to Diameter. Reference: Wikipedia, Current rating is calculated through curve fit from online data.

Parameters

- **arg** (*list*) – First 1 arguments are inputs.
  1. AWG ;
  2. Diameter ;length
  3. Current rating in still air ; current
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Absorptive_Filter_Equalizer**(*arg*, *defaultunits=[]*)

Equalizer using an absorptive filter composed of two coupled lines.

Parameters

- **arg** (*list*) – First 4 arguments are inputs.
  1. Reference Impedance ; impedance
  2. Coupling (dB) ;
  3. Center Frequency ; frequency
  4. Test Frequency ; frequency
  5. S21 (dB) ;
  6. Zeven ; impedance
  7. Zodd ; impedance

  Reference:
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Binomial_QWave_Impedance_Transformer**(*arg*, *defaultunits=[]*)

Binomial Quarter Wave Impedance Transformer.

Parameters

- **arg** (*list*) – First 5 arguments are inputs.
  1. Source Impedance;impedance
  2. Load Impedance;impedance
  3. Number Of Matching Sections;
  4. Max(dB(S₁₁)) In Frequency Band ;
  5. Center Frequency ; frequency
  6. Impedances ; impedance
  7. Bandwidth ; frequency Reference: Impedance Matching and Transformation.pdf
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Bridged_Tee_Attenuator_Analysis**(*arg*, *defaultunits=[]*)

Bridged Tee Attenuator Analysis.

Parameters

- **arg** (*list*) – First 3 arguments are inputs.
  1. Reference Impedance (Zo); impedance
  2. Series Impedance (Rs); impedance
  3. Parallel Impedance (Rp); impedance
  4. S(1,1) ;
  5. S(2,1) ; Reference:
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Bridged_Tee_Attenuator_Synthesis**(*arg*, *defaultunits=[]*)

Bridged Tee Attenuator Synthesis.

Parameters

- **arg** (*list*) – First 3 arguments are inputs.
  1. Reference Impedance (Zo); impedance
  2. Series Impedance (Rs); impedance
  3. Parallel Impedance (Rp); impedance
  4. S(1,1) ;
  5. S(2,1) ; Reference:
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Chebyshev_QWave_Impedance_Transformer**(*arg*, *defaultunits=[]*)

Chebyshev Quarter Wave Impedance Transformer.

Parameters

- **arg** (*list*) – First 6 arguments are inputs.
  1. Source Impedance ; impedance
  2. Load Impedance ; impedance
  3. Number Of Matching Sections ;
  4. Minimum Frequency ; frequency
  5. Maximum Frequency ; frequency
  6. Test Frequency ; frequency
  7. Impedances ; impedance
  8. Return Loss at Test Frequency ; Reference: Impedance Matching and Transformation.pdf + eski kod
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Chebyshev_Taper_Impedance_Transformer**(*arg*, *defaultunits=[]*)

Calculates performance and impedance values for an N-section Chebyshev Impedance Taper. Reference: Foundations for Microwave Engineering, Collin

Parameters

- **arg** (*list*) – First 5 arguments are inputs.
  1. Source Impedance ; impedance
  2. Load Impedance ; impedance
  3. Number Of Sections (Even) ;

4. Fractional Bandwidth (F2/F1) ;
5. Length (normalized to Lambda at fcenter) ;
6. Impedances ; impedance
7. Return Loss ;

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**CircularPlateCap**(*arg*, *defaultunits=[]*)

Circular Plate Capacitance.

Parameters

- **arg** (*list*) – First 3 arguments are inputs.
  1. Radius;length
  2. Height;length
  3. Dielectric Permittivity;
  4. Frequency; frequency
  5. Capacitance; capacitance
  6. Impedance; impedance
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Dia2AWG**(*arg*, *defaultunits=[]*)

Convert Diameter to AWG. Reference: Wikipedia

Parameters

- **arg** (*list*) – First 1 arguments are inputs.
  1. AWG ;
  2. Diameter ;length
  3. Current rating in still air ; current
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**DualFrequencyTransformer**(*arg*, *defaultunits=[]*)

Dual Frequency Transformer.

Parameters

- **arg** (*list*) – First 4 arguments are inputs.

  1. Source Impedance; impedance
  2. Load Impedance; impedance
  3. f1 Lower Frequency; frequency
  4. f2 Higher Frequency; frequency
  5. Z1; impedance
  6. Z2; impedance
  7. Electrical Length ; angle Reference: A Small Dual Frequency Transformer in Two Sections

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**DualTransformation1**(*arg*, *defaultunits=[]*)

Dual Transformation 1.

Parameters

- **arg** (*list*) – First 4 arguments are inputs.

  1. L1 ; inductance
  2. C1 ; capacitance
  3. L2 ; inductance
  4. C2 ; capacitance
  5. L1' ; inductance
  6. C1' ; capacitance
  7. L2' ; inductance
  8. C2' ; capacitance Reference: Microstrip Filters for RF-Microwave Applications, s.25, Figure 2.6a

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**DualTransformation2**(*arg*, *defaultunits=[]*)

Dual Transformation 1. Reference: Microstrip Filters for RF-Microwave Applications, s.25, Figure 2.6b

Parameters

- **arg** (*list*) – First 4 arguments are inputs.
  1. L1 ; inductance
  2. C1 ; capacitance
  3. L2 ; inductance
  4. C2 ; capacitance
  5. L1' ; inductance
  6. C1' ; capacitance
  7. L2' ; inductance
  8. C2' ; capacitance
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**EWG_ABCD**(*a*, *b*, *er*, *length*, *frek*)

components.**EWG_inv**(*a*, *b*, *er*, *length*, *frek*)

components.**EvanescentWGEquivalent**(*arg*, *defaultunits=[]*)

Waveguide Width Step from Rectangular Waveguide to Evanescent Mode Rectangular Waveguide. Reference: The Design of Evanescent Mode Waveguide Bandpass Filters for a Prescribed Insertion Loss Characteristic.pdf

Model= Xp1,Xs1,Xp1 ya da Xs2,Xp2,Xs2 (p: shunt, s: series) Zo=jXo

Parameters

- **arg** (*list*) – First 5 arguments are inputs.
  1. Waveguide Width;length
  2. Waveguide Height;length
  3. Dielectric Permittivity;
  4. Waveguide Length;length
  5. Frequency; frequency
  6. Series Inductance For Shunt-Series-Shunt Model; inductance
  7. Shunt Inductance For Shunt-Series-Shunt Model; inductance
  8. Series Inductance For Series-Shunt-Series Model; inductance
  9. Shunt Inductance For Series-Shunt-Series Model; inductance
  10. Characteristic Impedance; impedance
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Exponential_Taper_Impedance_Transformer**(*arg*, *defaultunits=[]*)

Exponential Impedance Taper. Reference: Foundations for Microwave Engineering, Collin

Parameters

- **arg** (*list*) – First 5 arguments are inputs.
  1. Source Impedance ; impedance
  2. Load Impedance ; impedance
  3. Number Of Sections ;
  4. Fractional Bandwidth (F2/F1) ;
  5. Length (normalized to Lambda at fcenter) ;
  6. Impedances ; impedance
  7. Return Loss ;
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**GyselPowerDivider**(*arg*, *defaultunits=[]*)

Triangle network to Star network transformation. Reference:

Zo1: 1. port impedance Zo2: 2. port impedance Zo3: 3. port impedance R1: first isolation resistor (2.porta yakin) R2: second isolation resistor (3.porta yakin) ratio: P2/P3 power ratio Z1: impedance of transmission line between 1.port and 2.port Z2: impedance of transmission line between 1.port and 3.port Z3: impedance of transmission line between 2.port and isolation resistor Z4: impedance of transmission line between 3.port and isolation resistor

Parameters

- **arg** (*list*) – First 6 arguments are inputs.
  1. Zo1; impedance
  2. Zo2; impedance
  3. Zo3; impedance
  4. R1; impedance
  5. R2; impedance
  6. P2/P3 ratio;
  7. Z1; impedance
  8. Z2; impedance
  9. Z3; impedance

10. Z4; impedance

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**HomogeneousRectWaveguideParameters_TE**(*arg*, *defaultunits=[]*)

Homogeneous Rectangular Waveguide Parameters. Reference: Marcuvitz Waveguide Handbook s.253

Parameters

- **arg** (*list*) – First 10 arguments are inputs.

1. Dielectric Permittivity in Waveguide;
2. Waveguide Width;length
3. Waveguide Height;length
4. Mode (0: Te, 1: Tm);
5. M;
6. N;
7. Tand Of Dielectric;
8. Electrical Conductivity Of Walls; electrical conductivity
9. Frequency; frequency
10. Physical Length;length
11. Cond Loss; loss per length
12. Diel Loss; loss per length
13. Cutoff Freq; frequency
14. Lambda_Guided;length
15. Impedance; impedance
16. Electrical Length; angle
17. Group Velocity; 17. Group Delay; time

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**InductivePostInWaveguide**(*arg*, *defaultunits=[]*)

Inductive Post In Waveguide.

Parameters

- **arg** (*list*) – First 6 arguments are inputs.

1. Dielectric Permittivity in Waveguide ;
2. Waveguide Width (a);length
3. Waveguide Height (b);length
4. Post Diameter (d);length
5. Waveguide Sidewall To Post Center (s);length
6. Frequency; frequency
7. Inductance;inductance
8. Capacitance; capacitance
9. Impedance; impedance Reference: Marcuvitz Waveguide Handbook s.257

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**InductiveWindowInWaveguide**(*arg*, *defaultunits=[]*)

Waveguide Width Step from Rectangular Waveguide to Evanescent Mode Rectangular Waveguide. Reference: Marcuvitz Waveguide Handbook s.253

Parameters

- **arg** (*list*) – First 6 arguments are inputs.

1. Dielectric Permittivity in Waveguide ;
2. Waveguide Width (a);length
3. Waveguide Height (b);length
4. Difference Of Waveguide Width To Window Width;length
5. Window Thickness;length
6. Frequency; frequency
7. Inductance;inductance
8. Capacitance; capacitance
9. Impedance; impedance

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Interference_Phase_Amp_Error**(*arg*, *defaultunits=[]*)

Maximum phase and amplitude variation of a signal in presence of an interfering signal.

Parameters

- **arg** (*list*) – First 1 arguments are inputs.

1. Difference in dB ;
2. Amplitude Error;
3. Phase Error; angle

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Klopfenstein_Taper_Impedance_Transformer**(*arg*, *defaultunits=[]*)

Calculates performance and impedance values for an N-section Klopfenstein Impedance Taper.

Parameters

- **arg** (*list*) – First 6 arguments are inputs.

1. Source Impedance ; impedance
2. Load Impedance ; impedance
3. Maximum Reflection Coefficient (dB) ;
4. Number Of Sections ;
5. Minimum Frequency ; frequency
6. Test Frequency ; frequency
7. Minimum Total Phase at Minimum Frequency ; angle ;
8. Impedances ; impedance
9. MAG(Reflection Coefficient) ; Reference: Microwave Engineering, Pozar

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**LC_Balun**(*arg*, *defaultunits=[]*)

Calculate LC Balun.

Parameters

- **arg** (*list*) – First 4 arguments are inputs.

1. Source Impedance (Rin) ; impedance
2. Load Impedances (RL) ; impedance
3. Frequency; frequency
4. Test Frequency ; frequency
5. Inductance ; inductance
6. Capacitance ; capacitance

7. S11 (dB) ;
8. S21 (dB) ;
9. S31 (dB) ; Reference:

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**L_BondWire**(*arg*, *defaultunits=[]*)

Inductance of a bond wire.

Parameters

- **arg** (*list*) – First 4 arguments are inputs.
  1. Bondwire Radius ;length
  2. Substrate Thickness ;length
  3. Distance Between End Points ;length
  4. Angle At End Points In Degrees ; angle
  5. Inductance ;inductance Reference: Transmission Line Design Handbook, Wadell, s.153
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**L_StraightFlatWire**(*arg*, *defaultunits=[]*)

Inductance of a flat wire.

Parameters

- **arg** (*list*) – First 6 arguments are inputs.
  1. Wire Width ;length
  2. Wire Thickness ;length
  3. Wire Length ;length
  4. Frequency ; frequency
  5. Relative Permeability ;
  6. Conductivity ; electrical conductivity
  7. Inductance ;inductance
  8. Impedance ;impedance Reference: Transmission Line Design Handbook, Wadell, s.382
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**L_StraightRoundWire**(*arg*, *defaultunits=[]*)

Inductance of a straight round wire.

Parameters

- **arg** (*list*) – First 5 arguments are inputs.
  1. Wire Diameter ;length
  2. Wire Length ;length
  3. Frequency ; frequency
  4. Dielectric Permeability ;
  5. Conductivity ; electrical conductivity
  6. Inductance ;inductance
  7. Impedance ; impedance Reference: Transmission Line Design Handbook, Wadell, s.380
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**L_air_core_coil**(*arg*, *defaultunits=[]*)

Inductance of a via hole in microstrip.

Parameters

- **arg** (*list*) – First 4 arguments are inputs.
  1. Wire Diameter (d) ;length
  2. Coil Inner Diameter (d_in) ;length
  3. Spacing Between Turns (s) ; length
  4. Number Of Turns ;
  5. Inductance ; inductance
  6. Resonance Frequency ; frequency Reference: www.microwavecoil.com , Microwave Components Inc.
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**L_microstrip_via_hole**(*arg*, *defaultunits=[]*)

Inductance of a via hole in microstrip.

Parameters

- **arg** (*list*) – First 2 arguments are inputs.
  1. Via Radius ;length
  2. Substrate Thickness ;length
  3. Inductance ; inductance Reference: Microstrip Via Hole Grounds in Microstrip.pdf
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**OptimumMitered90DegMicrostripBend**(*arg*, *defaultunits=[]*)

Optimum Mitered Microstrip Bend Parameters. Reference: Tranmission line design handbook, p.290

Parameters

- **arg** (*list*) – First 2 arguments are inputs.
  1. Microstrip Width;length
  2. Substrate Height;length
  3. Miter Length; length
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**OptimumMiteredArbitraryAngleMicrostripBend**(*arg*, *defaultunits=[]*)

Optimum Mitered Microstrip Bend Parameters. Reference: MWOHELP, MBENDA model

Parameters
arg (*list*) – First 2 arguments are inputs.

1. Microstrip Width;length;
2. Substrate Height;length;
3. Angle (0-180 degrees); angle ;
4. Miter Length; length ;

Burada scipy.interpolate.griddata kullanildi ve maalesef extrapolation yapmiyor. Sinir disi degerlerde dogrudan en yakin deger kullanildi.

defaultunits(list, optional): Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**PCBTrackCurrentCapacity**(*arg*, *defaultunits=[]*)

PCB Track Current Capacity.

Parameters

- **arg** (*list*) – First 7 arguments are inputs.
  1. Metal Width; length
  2. PCB Height; length
  3. Metal Thickness; length
  4. Allowable Temperature Rise; temperature
  5. Thermal Conductivity; thermal conductivity
  6. Electrical Conductivity; electrical conductivity
  7. External if 1, Internal if 0;
  8. Current ; current
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**PCBTrackCurrentCapacityIPC**(*arg*, *defaultunits=[]*)

PCB Track Current Capacity, IPC. Reference: IPC2221A

Parameters

- **arg** (*list*) – First 4 arguments are inputs.
  1. Metal Width;length
  2. Metal Thickness;length
  3. Allowable Temperature Rise; temperature
  4. External if 1, Internal if 0;
  5. Current ; current
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**ParallelPlateCap**(*arg*, *defaultunits=[]*)

Parallel Plate Capacitance.

Parameters

- **arg** (*list*) – First 4 arguments are inputs.

1. Width;length
2. Length;length
3. Height;length
4. Dielectric Permittivity;
5. Frequency; frequency
6. Capacitance; capacitance
7. Impedance; impedance

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Patch_Antenna_Analysis**(*arg*, *defaultunits=[]*)

Calculates performance and impedance values for an N-section Chebyshev Impedance Taper. Ref: Overview of Microstrip Antennas (Jackson) (Presentation) Reference: Foundations for Microwave Engineering, Collin

Parameters

- **arg** (*list*) – First 6 arguments are inputs.

1. Width (W) ; length
2. Length (L) ; length
3. Substrate Thickness (h);length
4. Dielectric Permittivity ;
5. Dielectric Loss Tangent ;
6. Metal Conductivity ; electrical conductivity
7. Resonance Frequency (f) ; frequency
8. Bandwidth ; frequency

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Pi_Attenuator_Analysis**(*arg*, *defaultunits=[]*)

Pi Attenuator Analysis.

Parameters

- **arg** (*list*) – First 3 arguments are inputs.

1. Reference Impedance (Zo); impedance
2. Series Impedance (Rs); impedance
3. Parallel Impedance (Rp); impedance

4. S(1,1) ;
5. S(2,1) ;
6. P1 ;
7. P2 ;
8. P3 ; Reference:

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Pi_Attenuator_Synthesis**(*arg*, *defaultunits=[]*)

Pi Attenuator Analysis.

Parameters

- **arg** (*list*) – First 3 arguments are inputs.

1. Reference Impedance (Zo); impedance
2. Series Impedance (Rs); impedance
3. Parallel Impedance (Rp); impedance
4. S(1,1) ;
5. S(2,1) ;
6. P1 ;
7. P2 ;
8. P3 ; Reference:

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**RectWG2EvanescentRectWGStep**(*a1*, *a2*)

Waveguide Width Step from Rectangular Waveguide to Evanescent Mode Rectangular Waveguide. Reference: The Design of Evanescent Mode Waveguide Bandpass Filters for a Prescribed Insertion Loss Characteristic.pdf

Parameters

- **arg** (*list*) – First 2 arguments are inputs.

1. Width of Rectangular Waveguide;length;
2. Width of Evanescent Mode Rectangular Waveguide;length;
3. Inductance; inductance
4. Turns Ratio;

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**SIW_EquivalentWidth**(*w*, *d*, *s*)

Equivalent width of substrate integrated waveguide.

Parameters

- **w** (*float*) – Distance between the centers of two via arrays.
- **d** (*float*) – Diameter of vias.
- **s** (*float*) – Distance between the centers of consecutive vias of via arrays.

Returns
Equivalent width of waveguide.

Return type
float

components.**Shorten90DegreeLine**(*arg*, *defaultunits=[]*)

Shortening 90 Degree Line with a capacitive load.

Parameters

- **arg** (*list*) – First 3 arguments are inputs.
  1. Impedance (Zo); impedance
  2. Center Frequency ; frequency
  3. Electrical Length (theta) ; angle
  4. Impedance (Z); impedance
  5. Capacitance ; capacitance
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Star2TriangleTransformation**(*arg*, *defaultunits=[]*)

Star network to Triangle network transformation.

Parameters

- **arg** (*list*) – First 3 arguments are inputs.
  1. Z1; impedance
  2. Z2; impedance
  3. Z3; impedance
  4. Z1'; impedance

5. Z2'; impedance

6. Z3'; impedance Reference: At star, z1 is connected to A-node, z2 is connected to B-node, z3 is connected to C-node At triangle, z1 is between A-B, z2 is between A-C, z3 is between B-C

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**SymmetricLangeCoupler**(*arg*, *defaultunits=[]*)

Symmetric Lange Coupler.

Parameters

- **arg** (*list*) – First 3 arguments are inputs.

1. C: Voltage coupling coefficient in dB (positive);

2. n: Number of fingers (should be even);

3. Reference Impedance;impedance

4. Zoo;impedance

5. Zoe;impedance Reference: Microwave Circuits, Analysis and Computer-Aided Design, Fusco

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Tee_Attenuator_Analysis**(*arg*, *defaultunits=[]*)

Tee Attenuator Analysis.

Parameters

- **arg** (*list*) – First 3 arguments are inputs.

1. Reference Impedance (Zo); impedance

2. Series Impedance (Rs); impedance

3. Parallel Impedance (Rp); impedance

4. S(1,1) ;

5. S(2,1) ;

6. P1 ;

7. P2 ;

8. P3 ; Reference:

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Tee_Attenuator_Synthesis**(*arg*, *defaultunits=[]*)

Tee Attenuator Synthesis.

Parameters

- **arg** (*list*) – First 5 arguments are inputs.
  1. Reference Impedance (Zo); impedance
  2. Series Impedance (Rs); impedance
  3. Parallel Impedance (Rp); impedance
  4. S(1,1) ;
  5. S(2,1) ;
  6. P1 ;
  7. P2 ;
  8. P3 ; Reference:
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Triangle2StarTransformation**(*arg*, *defaultunits=[]*)

Triangle network to Star network transformation. At star, z1 is connected to A-node, z2 is connected to B-node, z3 is connected to C-node At triangle, z1' is between A-B, z2' is between A-C, z3' is between B-C

Parameters

- **arg** (*list*) – Last 3 arguments are inputs.
  1. Z1; impedance
  2. Z2; impedance
  3. Z3; impedance
  4. Z1'; impedance
  5. Z2'; impedance
  6. Z3'; impedance
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Triangular_Taper_Impedance_Transformer**(*arg*, *defaultunits=[]*)

Triangular Impedance Taper. Reference: Foundations for Microwave Engineering, Collin

Parameters

- **arg** (*list*) – First 5 arguments are inputs.
  1. Source Impedance ; impedance
  2. Load Impedance ; impedance
  3. Number Of Sections (Even) ;
  4. Fractional Bandwidth (F2/F1) ;
  5. Length (normalized to Lambda at fcenter) ;
  6. Impedances ; impedance
  7. Return Loss ;
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Z_CWG**(*rad*, *freq*, *eps_r=1*, *v=0*, *n=1*, *mode='TE'*)

Computes the wave impedance of circular waveguide.

Parameters

- **v** (*int*) – Mode number of ϕ.
- **n** (*int*) – Radial mode number.
- **eps_r** (*float*) – Permittivity of filling material.
- **freq** (*float*) – Frequency (Hz).
- **mode** (*str*) – “TE” or “TM”.
- **rad** (*float*) – Radius.

Returns
Impedance.

Return type
Z (float)

components.**Z_WG_TE10**(*er*, *a*, *b*, *freq*, *formulation=1*)

components.**Zo_eeff_StraightWireOverSubstrate**(*arg*, *defaultunits=[]*)

Impedance and Effective Permittivity of Straight Wire Over Substrate.

Parameters

- **arg** (*list*) – First 4 arguments are inputs.
  1. Wire Diameter (d);length
  2. Height Of Wire Center Above Ground (h);length
  3. Dielectric Thickness (t);length

4. Dielectric Permittivity ;

5. Impedance ; impedance

6. Effective Diel. Permittivity ; Reference: Transmission Line Design Handbook, Wadell, s.151

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**Zo_eeff_WireOnGroundedSubstrate**(*arg*, *defaultunits=[]*)

Impedance and Effective Permittivity of Straight Wire Over Substrate.

Parameters

- **arg** (*list*) – First 4 arguments are inputs.

1. Wire Diameter (d);length

2. Dielectric Thickness (t);length

3. Dielectric Permittivity ;

4. Impedance ; impedance

5. Effective Diel. Permittivity ; Reference: Transmission Line Design Handbook, Wadell, s.151 Note: eeff is the same as eeff of microstrip with w=2*d, t=0

- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns
arg

Return type
list

components.**fcutoff_CWG**(*rad*, *eps_r=1*, *v=0*, *n=1*, *mode='TE'*)

Computes the cutoff frequency of circular waveguide.

Parameters

- **v** (*int*) – Mode number of ϕ.
- **n** (*int*) – Radial mode number.
- **eps_r** (*float*) – Permittivity of filling material.
- **mode** (*str*) – "TE" or "TM".
- **rad** (*float*) – Radius.

Returns
Cutoff frequency (Hz).

Return type
fc (float)

components.**thermal_conductance_of_via_farm**(*arg*, *defaultunits*)

Thermal conductance of an array of vias in PCB.

Parameters

- **arg** (*list*) – First 7 arguments are inputs.
  1. Plated Via Diameter (d);length
  2. Plating Thickness (t);length
  3. Area Width (w);length
  4. Area Height (l);length
  5. Dielectric Height (h);length
  6. Number Of Vias (n);
  7. Dielectric Thermal Conductivity ; thermal conductivity
  8. Metal Thermal Conductivity ; thermal conductivity
  9. Thermal Conductance (W/K) ;
  10. Thermal Resistance (K/W) ;
- **defaultunits** (*list, optional*) – Default units for quantities in *arg* list. Default is [] which means SI units will be used if no unit is given in *arg*.

Returns

arg

Return type

list

components.**thermal_conductance_of_via_farm_view**(*arg*, *defaultunits*)

CHAPTER

THREE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX

Symbols

A

B

C

D

E

F

G

T

U

W

Y

Z