
ARES

Release 1.0

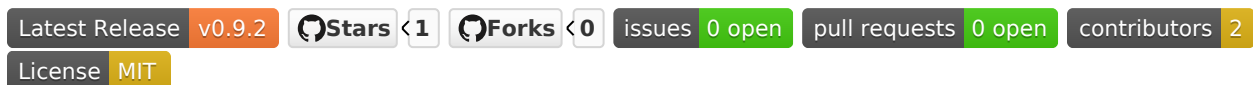
Alex D. Santiago-Vargas, José Abraham Bolaños Vargas

May 13, 2025

CONTENTS

1	Automated Radio Evaluation Suite	1
1.1	Tutorials	1
1.2	Key Features	1
1.3	Contributors	2
1.4	Acknowledgments	3
2	Getting Started	5
2.1	Requirements	5
2.2	Download and Install	5
2.3	Network Configuration	5
2.4	Compatibility	6
3	Tutorials	7
3.1	App Introduction	7
3.2	Instrument Database Tutorial	10
3.3	Antenna Tutorial	12
3.4	PA Tutorial	21
4	Code Reference	31
4.1	Source Code Overview	31
4.2	App Overview	33
4.3	Instrument Interfacing	34
4.4	Antenna Functions	39
4.5	Power Amplifier Functions	45
4.6	Supporting Functions	53
5	Example Gallery	59
5.1	Antenna Example Data	59
5.2	PA Example Data	61
6	FAQ	63
6.1	Do I need to install anything else to use ARES?	63
6.2	How do I find my instrument's VISA address?	63
6.3	My instrument is not connecting. What should I check?	64
6.4	Can I edit the instrument list manually?	64
6.5	Where are my saved test results and logs stored?	65
6.6	How do I show hidden tabs in the app?	66
6.7	Why can't I export a plot as PDF or TikZ?	66
6.8	VNA isn't returning expected values, what should I do?	66

AUTOMATED RADIO EVALUATION SUITE



ARES is an open-source MATLAB-based platform for performing automated RF measurements on power amplifiers and antennas. It interfaces seamlessly with existing laboratory equipment via VISA (LAN, GPIB, USB), supporting streamlined data collection, visualization, and export, all within a graphical user interface.

ARES is free, fully customizable, and actively maintained. It provides a cost-effective alternative to commercial measurement software and allows full transparency and control over your test flow. [Download the latest release.](#) and [get started with the setup guide.](#)

1.1 Tutorials

Explore the functionality of ARES through guided documentation:

- [Instrument Database Tutorial](#)
- [Antenna Measurement Tutorial](#)
- [PA Measurement Tutorial](#)

Full documentation is hosted on [Read the Docs.](#)

1.2 Key Features

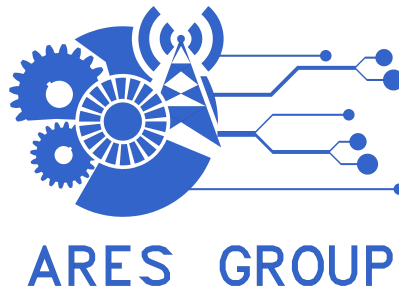
- VISA-based instrument control via **GPIB**, **LAN**, and **USB**
- Support for multiple ETS-Lindgren, Keysight, and Rohde & Schwarz instruments
- Built in deembedding support and DC power supply control
- PA FoM measurements:
 - Gain
 - Output Power
 - Drain Efficiency

- Power Added Efficiency
- Antenna gain measurements:
 - Gain Comparison Method (i.e., Two-Antenna Method)
 - Gain Transfer Method (i.e., Comparison Antenna Method) using a reference measurement
 - Antenna return loss (magnitude & phase).
 - 2D/3D radiation pattern visualization
- Save/load test measurements in standardized file formats for data analysis.
- Plot test measurements within the app for quick visualization.
- Export plots to **PDF**, **PNG**, **JPEG**, and **TikZ** for publication

i Export Tip

TikZ export is supported for Cartesian 2D plots. Polar and 3D plots support image formats only.

1.3 Contributors



- Author: José Abraham Bolaños Vargas (@bolanosv)
- Mentor: Alex David Santiago Vargas (@AlexDCode, [Google Scholar](#))
- PI: Dimitrios Peroulis ([Google Scholar](#))
- Adaptive Radio Electronics and Sensors Group
- Purdue University

1.4 Acknowledgments

This project makes use of several open-source tools. The authors acknowledge the following:

- **matlab2tikz** for enabling high-quality \LaTeX -compatible plot exports. Copyright (c) 2008–2016 Nico Schlömer. All rights reserved.

E. Geerardyn, N. Schlömer, et. al. (2025). “matlab2tikz: Version 1.1.0”. Zenodo, Oct. 20, 2016. doi: 10.5281/zenodo.162246.

GETTING STARTED

2.1 Requirements

To run the Automated Radio Evaluation Suite (ARES), ensure the following software is installed:

- [MATLAB R2023b](#) or above.
- [MATLAB Instrument Control Toolbox](#) (for VISA communication)
- [MATLAB RF Toolbox](#) (for S-parameter analysis)
- [MATLAB Antenna Toolbox](#) (for 3D radiation pattern visualization)
- [Keysight Connection Expert](#) (for VISA drivers; Install the pre-requisite first, then the main installer)

2.2 Download and Install

1. Download the latest release of the Automated Radio Evaluation Suite from [releases](#).
2. **If using the MATLAB App installer:** Follow the instructions in the [Packaging and Installing MATLAB Apps Guide](#).

2.3 Network Configuration

Skip this section if you only intend to use the app for plotting previously collected data.

Skip this section if you're using USB or GPIB. No additional configuration is needed when using USB or GPIB connections.

For LAN-based communication, ensure the PC is on the same network as your instruments and apply the following manual network settings:

1. **Connect via Wi-Fi or Ethernet** to the same local network as your lab instruments.
2. **Set the following IPv4 configuration:** Ensure your device's IP address is set up with the following network settings to connect to the instruments within the ARES Lab:
 - **IP Address:** 192.168.0.XXX (where XXX is between 2 and 255).

- **Default Gateway:** 192.168.1.1
- **Subnet Mask:** 255.255.0.0 (This enables accessing instruments with IP addresses in 192.168.XXX.XXX)
- **DNS Servers:**
 - Primary: 128.210.11.5
 - Alternate: 128.210.11.57

To verify connectivity, open a terminal or command prompt and **ping** the IP address of the intended instrument in the command window. If this is successful for the desired instrument, the network settings are appropriate.

2.4 Compatibility

ARES has been tested and is compatible with the following instruments:

- **Keysight PNA-L N5235B** — 4-Port Network Analyzer
- **Keysight E36233A/E36234A** — Dual Output DC Power Supplies
- **Keysight CXA** — Signal Analyzer
- **Keysight PXA** — Signal Analyzer
- **Rohde & Schwarz SMW200A** — Signal Generator
- **Hewlett-Packard E4433B** — Signal Generator
- **ETS Lindgren EMCenter** — Position Controller
- **ETS Lindgren Linear Slider** — Slider Controller

Other SCPI-compatible instruments may also work, provided they support the same command set as those listed.

3.1 App Introduction

3.1.1 Instrument Connections

ARES connects to lab equipment using **VISA resource addresses**, which are stored in the instrument database. When the user presses *Connect*, ARES reads these addresses and establishes communication using MATLAB's `visadev` interface.

How It Works:

- Each dropdown in the *Instruments* tab corresponds to an entry in the instrument database.
- When a valid resource string is selected, ARES:
 - Attempts to connect using `visadev`.
 - Sends an `*IDN?` query to confirm the instrument identity.
 - Updates the GUI with the model number or connection status.
 - Handles connection failures gracefully using the error logging system.

Supported Address Types

ARES supports VISA-compatible resource strings:

Type	Example Format
GPIB	GPIB0::19::INSTR
LAN	TCPIP0::192.168.0.101::inst0::INSTR
SOCKET	TCPIP0::192.168.2.16::5025::SOCKET
USB	USB0::0x0957::0x2807::MY62003488::0::INSTR

Plot Exporting

ARES includes a built-in **right-click export system** for all major plots across the application, making it easy to save results directly from the GUI.

How It Works:

- Right-click on any plot.
- A context menu with export options appears next to the plot.
- Choose a format:
 - **PNG, JPG** – High resolution images
 - **PDF** – Vector graphics for papers and presentations
 - **TikZ** – For LaTeX users

Note

- **TikZ export** requires the `matlab2tikz` package and is only supported for **Cartesian 2D plots**.
- **PDF export** is **not supported** for the 3D Radiation Plot due to MATLAB's `exportgraphics` limitations on 3D content.

Measurement Time Logging

ARES automatically logs performance metrics for every measurement session to help users monitor and optimize testing efficiency.

Each log entry includes:

- **Measurement Type:** PA or Antenna
- **Start Time** and **End Time**
- **Total Duration** of the measurement
- **Number of Measured** points or positions
- **Average Time** per point/position

This log is stored in:

```
<userpath>/ARES/ARES_Measurement_Log.txt
```

This log enables users to:

- Benchmark performance across different configurations
- Optimize sweep parameters and delays
- Maintain a record of test durations for reproducibility or documentation

Measurement Log Example:

```
[27-Apr-2025 15:57:38]
Antenna Measurement Summary:
Start Time: 27-Apr-2025 15:56:11
End Time: 27-Apr-2025 15:57:38
```

(continues on next page)

(continued from previous page)

```

Duration: 00:01:26
Total Positions Measured: 19
Average Time Per Position: 00:00:04
-----

```

Error Logging

ARES includes a built-in error handling system to capture unexpected issues during execution. When an error occurs:

- 1) A **pop-up alert** with the error message is shown to the user.
- 2) A detailed error report is automatically logged to:

```
<userpath>/ARES/ARES_Error_Log.txt
```

Each error entry includes:

- **Timestamp**
- **Error Message**
- **Error Identifier**
- **Stack Trace** (file, function name, and line number)

Note

- If the log file or folder doesn't exist, ARES will create it automatically.

This feature is particularly useful when debugging or reporting issues. The user can easily share the log file with our team to diagnose problems quickly.

Error Log Example:

```

[08-May-2025 19:57:30]
Error: Unrecognized function or variable 'units'.
Identifier: MATLAB:UndefinedFunction

Stack Trace:
In file: C:\Users\USERNAME\Documents\GitHub\AutomatedRadioEvaluationSuite\src\support\
↳ AntennaFunctions\RADIATIONPATTERN3D.m
Function: RADIATIONPATTERN3D
Line: 70

In file: C:\Users\USERNAME\Documents\GitHub\AutomatedRadioEvaluationSuite\src\support\
↳ AntennaFunctions\plotAntenna3DRadiationPattern.m
Function: plotAntenna3DRadiationPattern
Line: 83

In file: C:\Users\USERNAME\Documents\GitHub\AutomatedRadioEvaluationSuite\src\ARES.mlapp
Function: ARES.AntennaSelectedFrequencyValue
Line: 1793

In file: C:\Program Files\MATLAB\R2025a\toolbox\matlab\appdesigner\appdesigner\runtime\

```

(continues on next page)

(continued from previous page)

```

→ +matlab\+apps\AppBase.m
Function: @(source,event)executeCallback(ams,app,callback,requiresEventData,event)
Line: 54
-----

```

3.2 Instrument Database Tutorial

ARES maintains a user-defined instrument database stored as a .csv file within the ARES directory in your MATLAB user path. This database enables persistent tracking of instrument configurations across app sessions.

If the database does not exist, ARES automatically creates one by copying a default template from the source directory.

3.2.1 Database Format

The instrument database is a simple .csv file with the following structure:

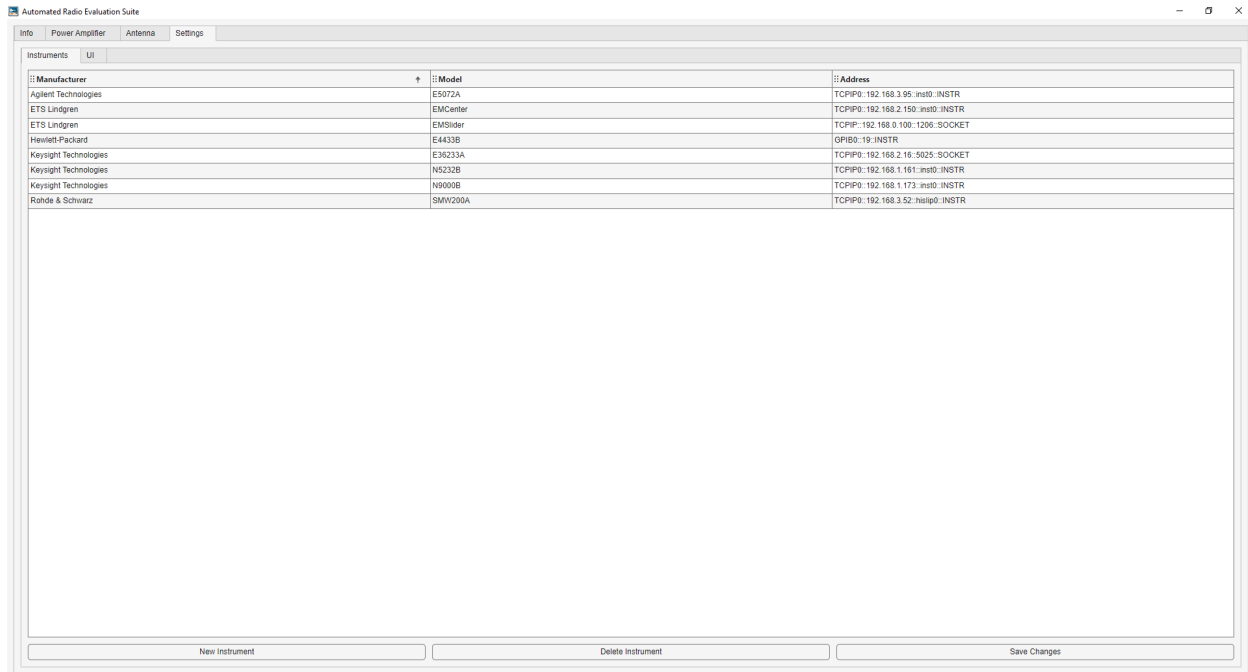
Manufacturer	Model	Address
Agilent Technologies	E5072A	TCPIP0::192.168.3.95::inst0::INSTR
ETS Lindgren	EMCenter	TCPIP0::192.168.2.150::inst0::INSTR
Hewlett-Packard	E4433B	GPIB0::19::INSTR
Keysight Technologies	E36233A	TCPIP0::192.168.2.16::5025::SOCKET

- **Manufacturer:** The name of the equipment vendor.
- **Model:** The specific model number.
- **Address:** The VISA resource string used to connect (supports LAN, GPIB, or socket-based communication).

3.2.2 Managing Instruments

To manage instruments in the app:

1. Click **New Instrument** to add a new entry.
2. Fill in the required fields: **Manufacturer**, **Model**, and **Address**.
3. Added instruments will appear in each measurement module's **Instruments** tab.
4. To remove an instrument, select its row and click **Delete Instrument**.
5. After making changes, click **Save Changes** to update your local database.



For more technical details, explore the [Instrument Interfacing Section](#), which explains how ARES uses VISA protocols to communicate with connected instruments.

3.2.3 Manual Editing (Optional)

Although the app provides a user-friendly interface for managing instruments, users can edit the instrument database file directly using a text editor or spreadsheet software (e.g., Excel, VS Code, etc.).

File Location

The database file is located in your **MATLAB user path**, under the ARES directory:

```
<userpath>/ARES/instrument_database.csv
```

To check your user path, run the following command in the MATLAB command window:

```
userpath
```

Editing Guidelines

- Open the .csv file with a CSV-compatible editor to avoid corrupting the format.
- Do not modify the header row (Manufacturer, Model, Address).
- Each row must contain exactly three fields: Manufacturer, Model, and a valid VISA Address.

3.3 Antenna Tutorial

The Antenna Module enables parametric measurements by varying the position of the Device Under Test (DUT) and acquiring its frequency response. The application captures the data, calculates antenna gain, saves the results, and visualizes the plots. Measurements can be reloaded from saved data, so tests do not need to be repeated unnecessarily.

Sample datasets are available in the [data/Antenna](#) folder.

Average Measurement Time

- Scan θ (1° step) with fixed ϕ : ~16 minutes
- Scan θ (3° step) with fixed ϕ : ~8 minutes
- Scan θ (5° step) with fixed ϕ : ~6 minutes
- 3D scan (3° step for both θ and ϕ): ~20 hours, (~380 MB for 201 frequency points)
- 3D scan (5° step for both θ and ϕ): ~8 hours
 - Note only half θ scan may be needed for full 3D pattern, cutting the total time in half.

3.3.1 Theory

Foundational Equations

Friis Transmission Equation:

$$\frac{P_r}{P_t} = G_t G_r \left(\frac{\lambda}{4\pi d} \right)^2$$

Where:

- P_r : Received power
- P_t : Transmitted power
- G_t : Transmitter Antenna Gain
- G_r : Receiver Antenna Gain
- λ Wavelength
- d : Distance between transmitter and receiver

Which can be expressed in dB scale as:

$$P_r^{[dB]} = P_t^{[dB]} + G_t^{[dBi]} + G_r^{[dBi]} + 20 \log \left(\frac{\lambda}{4\pi d} \right)$$

The Free Space Path Loss (FSPL) factor is given by:

$$FSPL = 20 \log \left(\frac{\lambda}{4\pi d} \right)$$

The ratio of received to transmitted power will be the measured magnitude:

$$S_{21} = \frac{P_r}{P_t}$$

Gain Comparison Method

In the gain comparison method (i.e., two antenna method), the reference antenna gain is known. Hence, we can solve the Friis transmission equation with this assumption and express it in terms of the measured S-parameters $S_{21}^{[dB]}$, calculated $FSPL$, and reference antenna gain $G_{REF}^{[dBi]}$.

$$G_{DUT}^{[dBi]} = S_{21}^{[dB]} - FSPL - G_{REF}^{[dBi]}$$

Gain Transfer Method

In the gain transfer method (i.e., one antenna method), the DUT and reference antenna are identical ($G_t^{[dBi]} = G_r^{[dBi]}$). Hence, we can solve the Friis transmission equation with this assumption and express it in terms of the measured S-parameters $S_{21}^{[dB]}$ and calculated $FSPL$.

$$G_{DUT}^{[dBi]} = \frac{S_{21}^{[dB]} - FSPL}{2}$$

3.3.2 Performing the Measurement

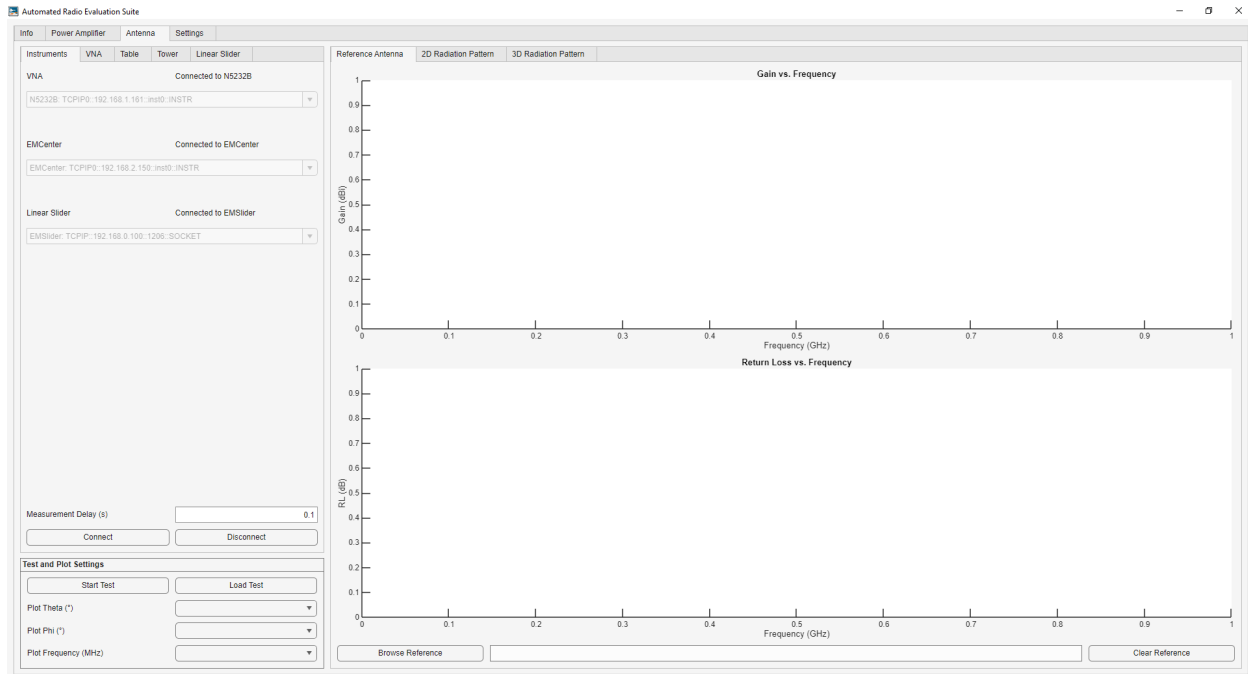
Calibration

- To get started, calibrate the Vector Network Analyzer (VNA) at the measurement plane, where the reference antenna and DUT will be connected. Set your frequency range and number of points (or step size) as desired **before calibration**.
- Using an eCal is highly recommended, as shown in the following [demonstration](#).

Connect to the Instruments

- Select the relevant instrument VISA addresses in each dropdown of the *Instruments* tab.
- Select *None: NA* for the instruments that will not be used.
- Follow the [Instrument Database Tutorial](#) for detailed information on how to edit the user-defined instrument database.
- Once all the addresses have been populated, click on *Connect* at the bottom to establish a connection to each instrument and *Disconnect* to clear all connections.

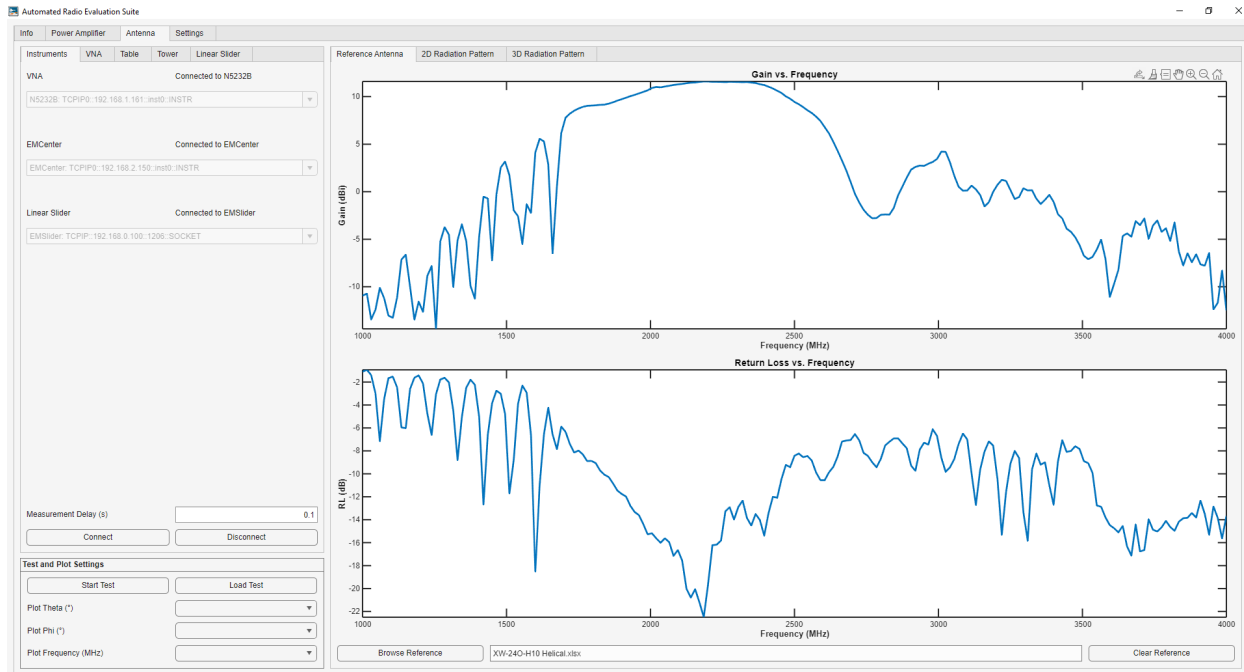
The *Measurement Delay* (s) can be modified at any time before the measurement starts. This value is the time in seconds to wait between setting all the instruments and capturing the data.



Load Reference Antenna Data (for Gain Comparison Method)

For the *Gain Comparison Method*, the reference antenna gain needs to be loaded using the same data format as measured antennas.

- In the *Reference Antenna* window, click *Browse Reference* and select the file containing your reference antenna data.
- Only the boresight gain ($\theta = 0, \phi = 0$) is required from the reference antenna.
- After loading, the **boresight gain** and **return loss magnitude** over frequency will be plotted in the results view.
- The filename of the loaded reference data will be shown below the plots.
- To remove the file, click *Clear Reference*.

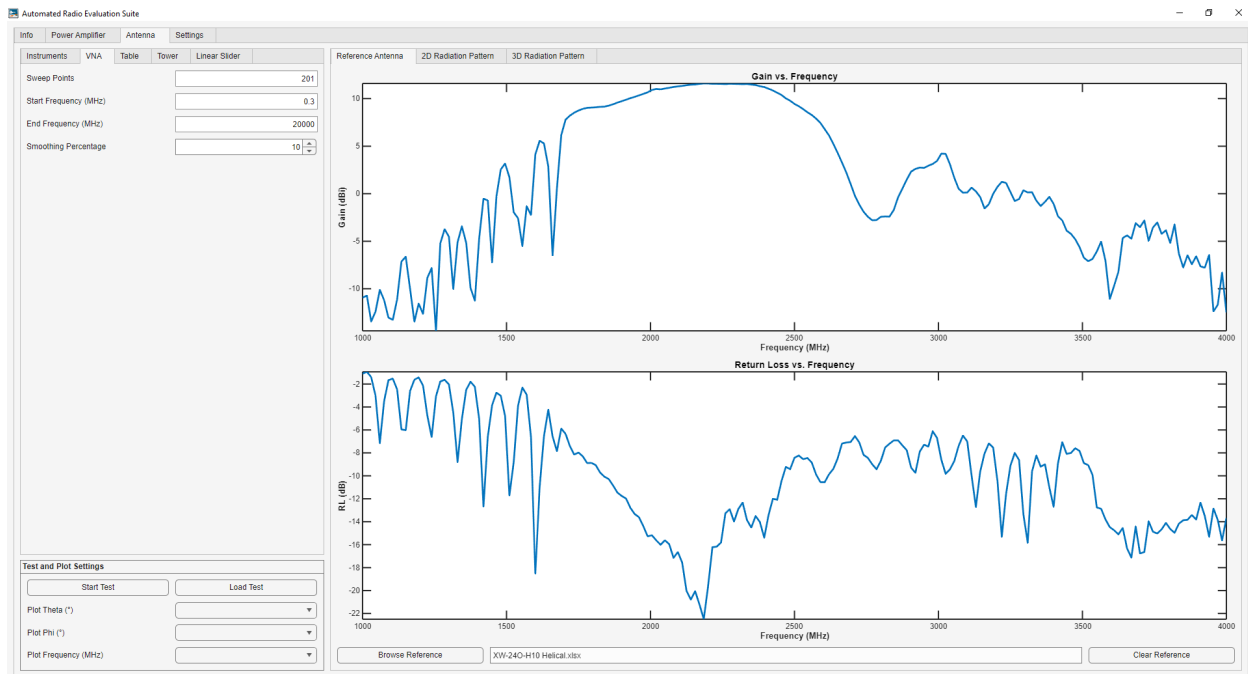


Configure the VNA

Use the *VNA* tab to review and adjust Vector Network Analyzer (VNA) settings. When the VNA is connected, its current configuration is automatically loaded into the app. If you modify key parameters such as *Sweep Points*, *Start Frequency*, or *Stop Frequency*, the existing calibration may no longer be valid.

To avoid invalidating the calibration, it is recommended to perform the VNA calibration before connecting to the VNA and then leave the loaded values unchanged. Keep in mind that the frequency and power ranges are constrained by the capabilities of the connected instrument.

Optionally, the app can apply smoothing to the VNA data using a moving average filter with the given number of samples. This is controlled via the *Smoothing Percentage* setting. If set to zero, smoothing is disabled.



Configure the Turntable (θ axis)

Use the *Table* tab to set up θ -axis rotation for antenna measurements.

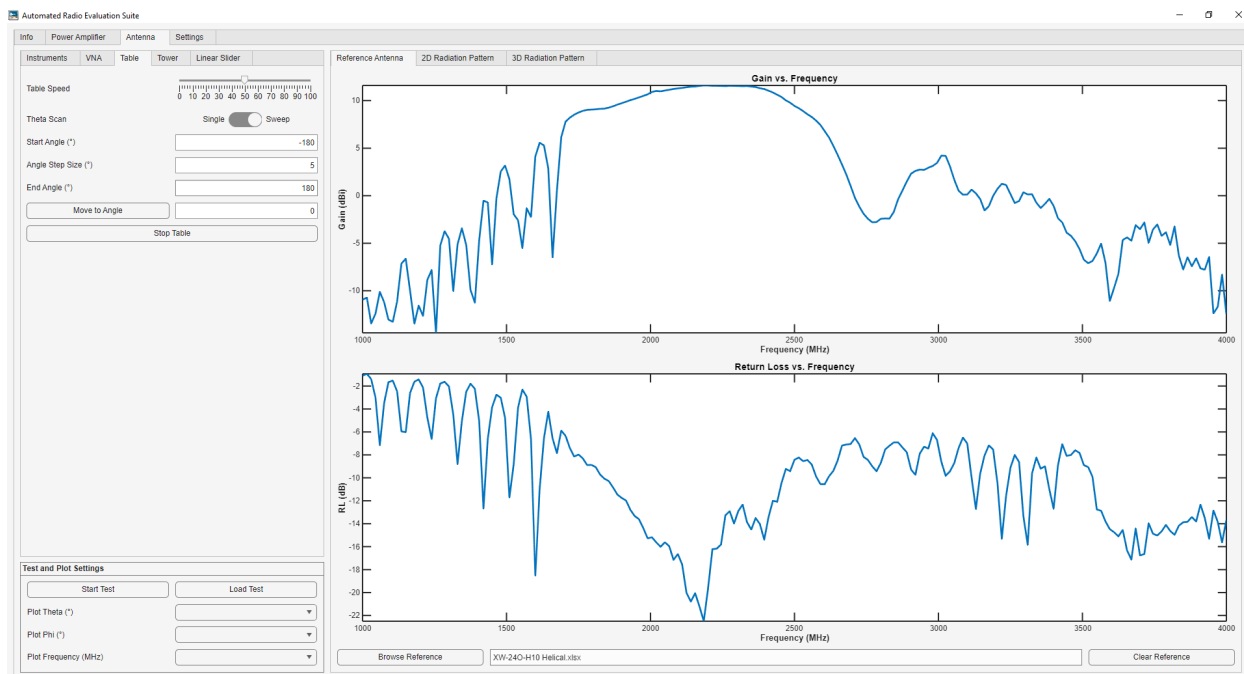
To configure the turntable:

- Select either a static or parametric sweep.
- Adjust the *Table Speed* using the slider.
- Enter appropriate *Start Angle*, *Angle Step Size*, and *Stop Angle* values.

Note: Software inputs range from -180° to 180° , but the table is configured to operate from 0° to 360° . The software automatically translates angles accordingly.

To control the turntable manually:

- Enter a desired target position and click on *Move to Angle*.
- To stop the movement at any time, click on *Stop Table*.



Configure the Tower (ϕ axis)

Use the *Tower* tab to set up ϕ -axis rotation for antenna measurements.

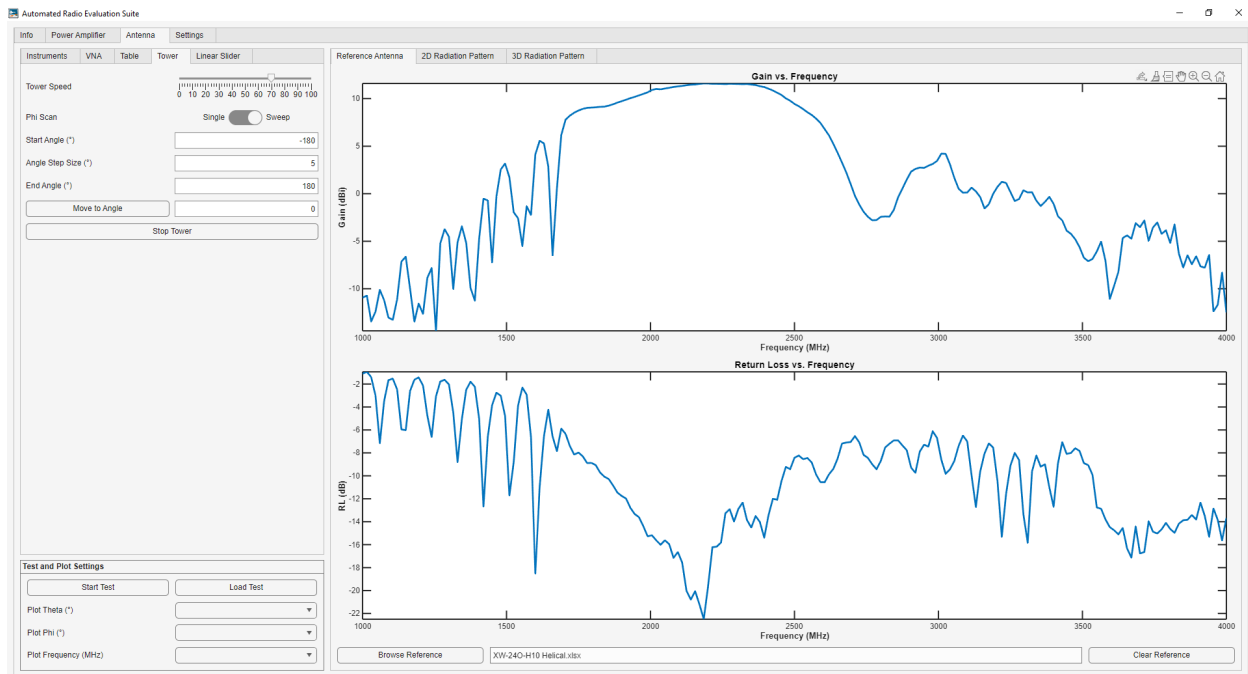
To configure the tower:

- Select either a static or parametric sweep.
- Adjust the *Tower Speed* using the slider.
- Enter appropriate *Start Angle*, *Angle Step Size*, and *Stop Angle* values.

Note: Software accepts inputs between -180° and 180° .

To control the tower manually:

- Enter a desired target position and click on *Move to Angle*.
- To stop the movement at any time, click on *Stop Tower*.



Configure the Linear Slider

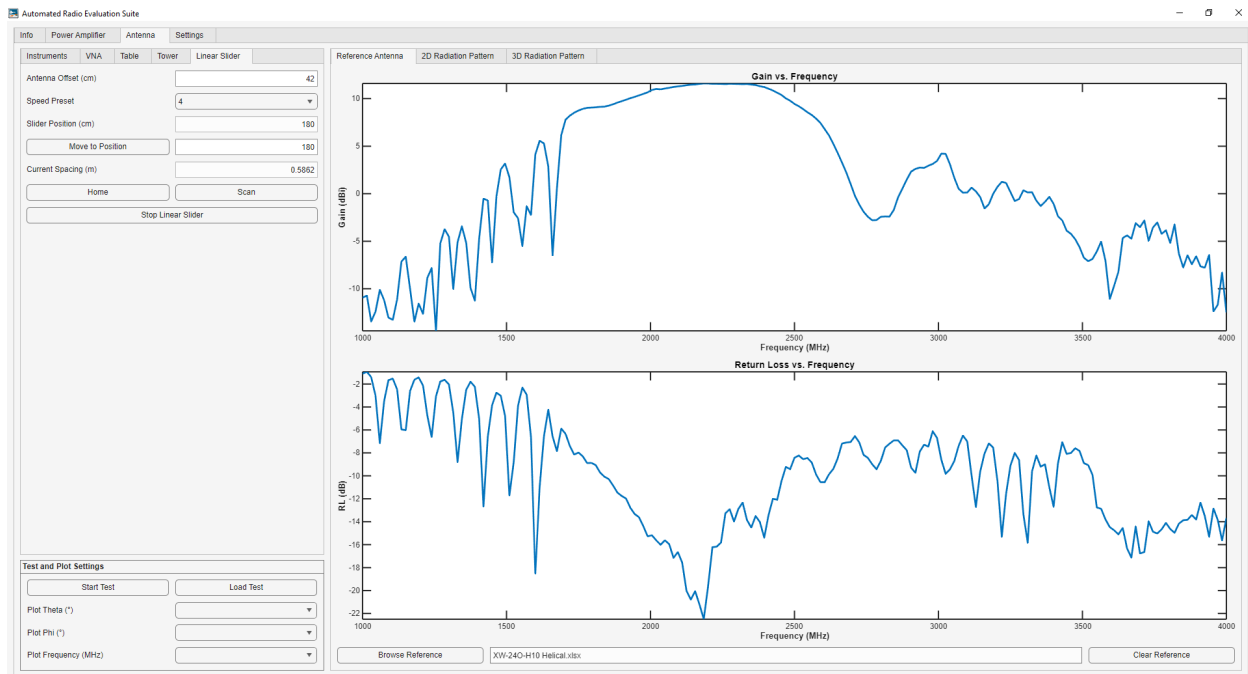
Use the *Linear Slider* tab to control antenna spacing and movement along the rail.

To configure and operate the slider:

- Set the *Antenna Offset* in centimeters, which accounts for the physical length of both the DUT and reference antennas relative to the mounting fixtures.
- Choose a *Speed Preset* for how fast the slider moves.
- Enter a target *Slider Position* and click *Move to Position* to move the antenna.
- Use the *Home*, *Scan*, or *Stop* buttons to control motion directly.

Live displays include:

- *Slider Position* – the real-time position of the slider, given in centimeters.
- *Current Spacing* – the calculated antenna separation, factoring in the slider position and antenna offset, given in meters.

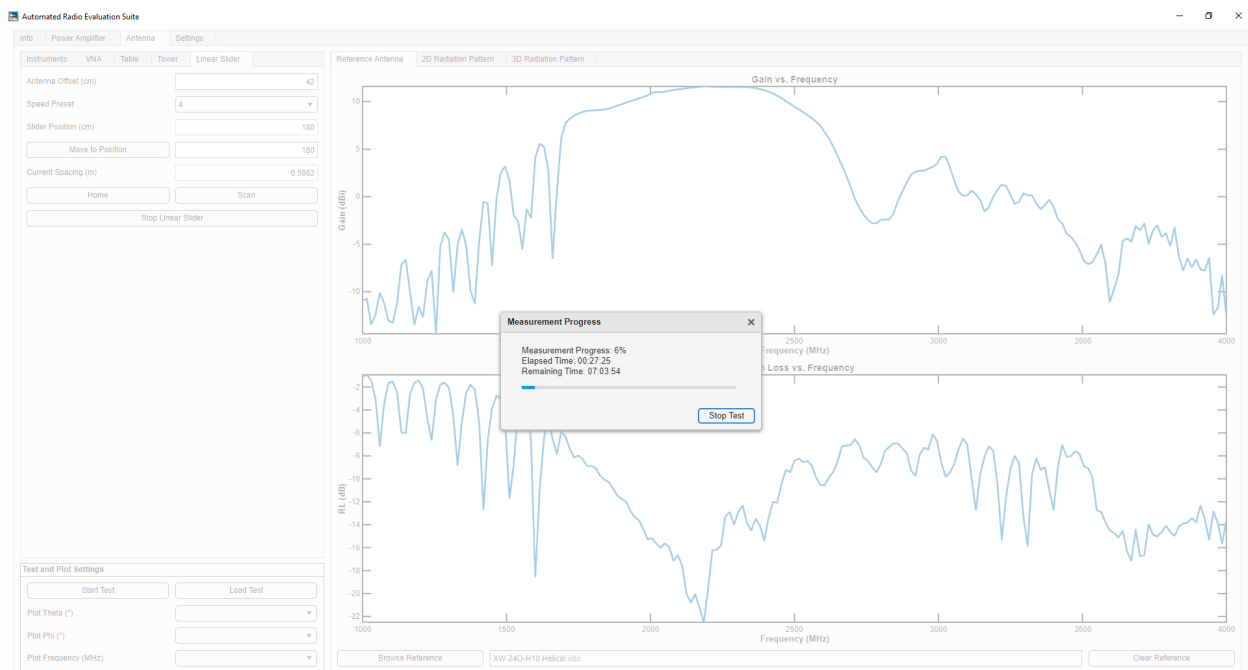


Run the Test and Plot the Results

After verifying all configuration settings, click *Start Test* to begin the measurement.

During the test:

- A progress window will show elapsed time and estimated completion time.
- To terminate the test, press the *Stop Test* button on the progress window.
- Once the test is finished, a prompt will appear to save the results.



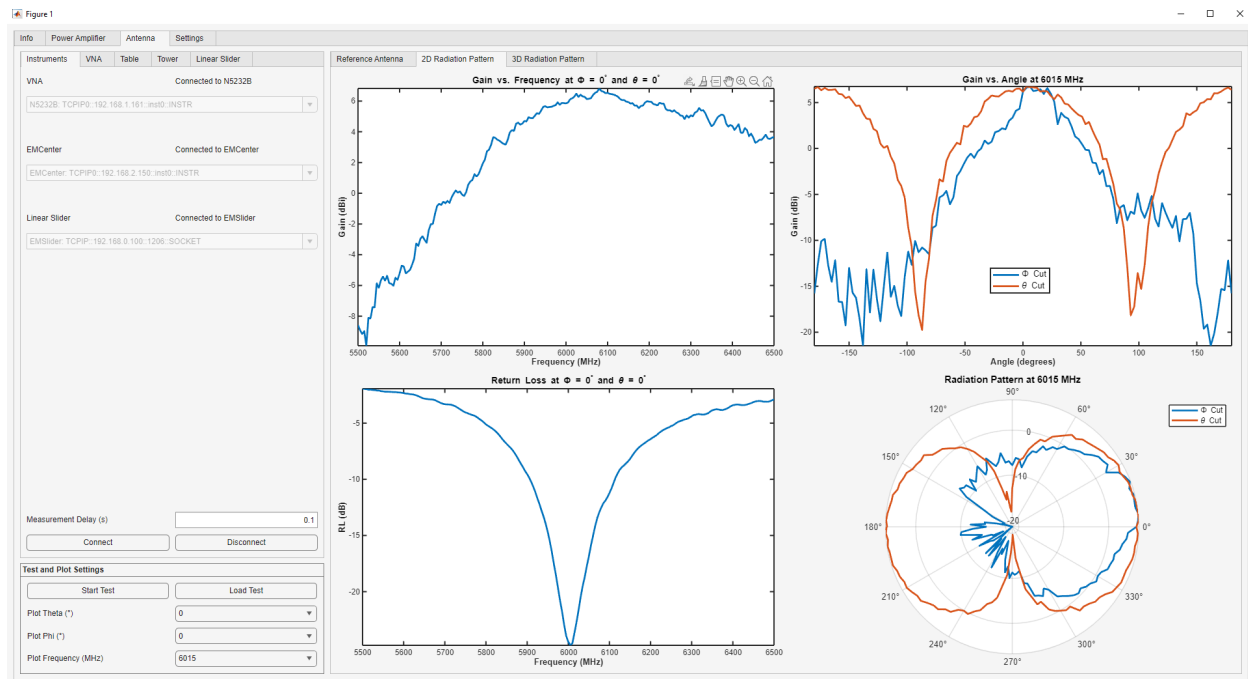
After saving your results:

- The measurement data is automatically loaded and visualized by ARES.
- To view previous results, use the *Load Test* button.

2D Radiation Pattern Results

The *2D Radiation Pattern* window displays:

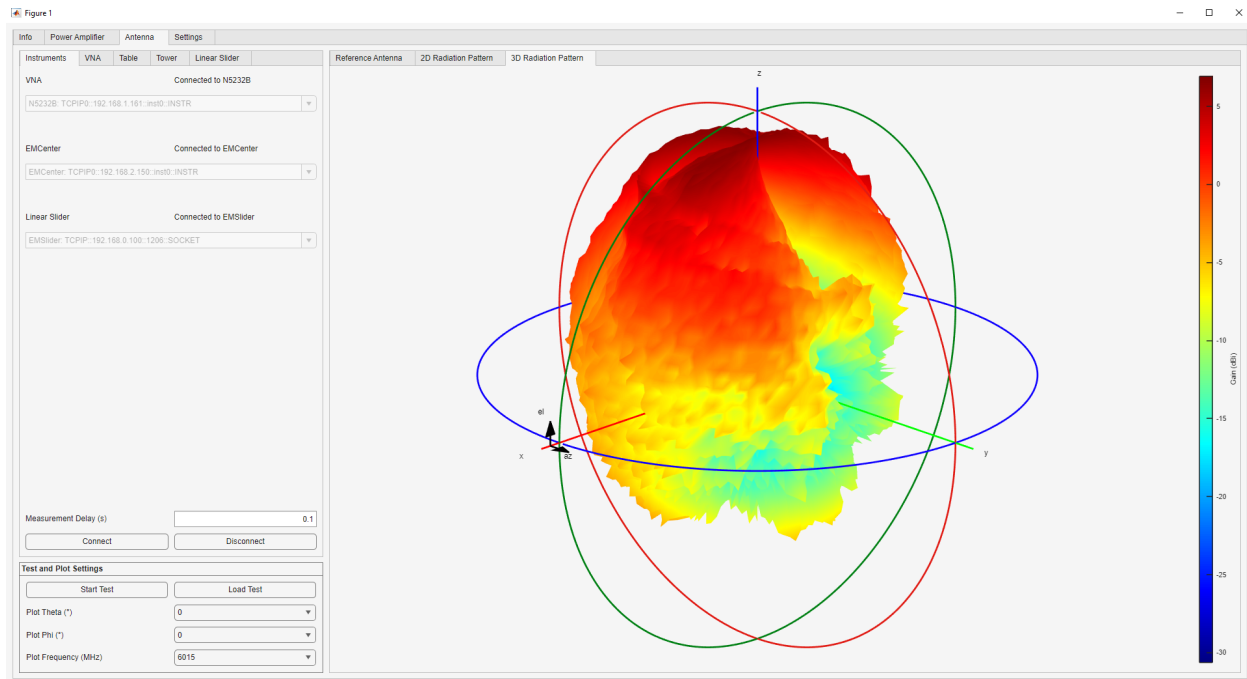
- Realized gain vs. frequency
- Return loss vs. frequency
- 2D cartesian and polar plots based on the selected value from the *Frequency*, θ , and ϕ dropdowns.



3D Radiation Pattern Results

The *3D Radiation Pattern* window displays:

- Full 3D gain pattern for the selected frequency.



3.4 PA Tutorial

The Power Amplifier (PA) module performs parametric measurements by sweeping frequency, input RF power, and controlling up to four power supplies. The software captures measurements, calculates figures of merit (FoM), saves the data, and plots the results. You do not need to rerun the test every time, as previously saved data can be reloaded into the app.

Sample datasets are available in the [data/PA](#) folder.

i Average Measurement Time

- Less than 1 second per data point with the default measurement delay.
- A data point corresponds to one combination of the parametric sweep, represented as a single row in the parameters and results table.

3.4.1 Theory

Gain and Efficiency

Drain Efficiency (DE):

$$DE = \frac{P_{out}}{P_{DC}}$$

Power Added Efficiency (PAE):

$$PAE = \frac{P_{out} - P_{in}}{P_{DC}}$$

Gain:

$$G = \frac{P_{out}}{P_{in}} = P_{out}^{[dB]} - P_{in}^{[dB]}$$

Where,

- P_{out} : RF output power
- P_{in} : RF input power
- P_{DC} : DC supply (drain) power

Gain Compression Points

The gain compression points characterize the nonlinearity of a power amplifier (PA). These include:

- **1 dB Compression Point** (P_{-1dB}): Output power where gain drops by 1 dB from peak value.
- **3 dB Compression Point** (P_{-3dB}): Output power where gain drops by 3 dB from peak value.
- **Saturation Power** (P_{sat}): Absolute maximum output power.

These refer to output-referred compression points, which are standard for PAs. In contrast, input-referred compression points may be used for linear or low-noise amplifiers. These figures of merit indicate the linearity of the device under test (DUT).

Calibration

Direct Measurement (Supported):

- Fixed-level measurements.
- Small-signal (passive) S-parameters.
- Large-signal driver measurements.

Standard instrument connections support these calibration types directly through the ARES interface.

Indirect Measurement (Not Yet Supported):

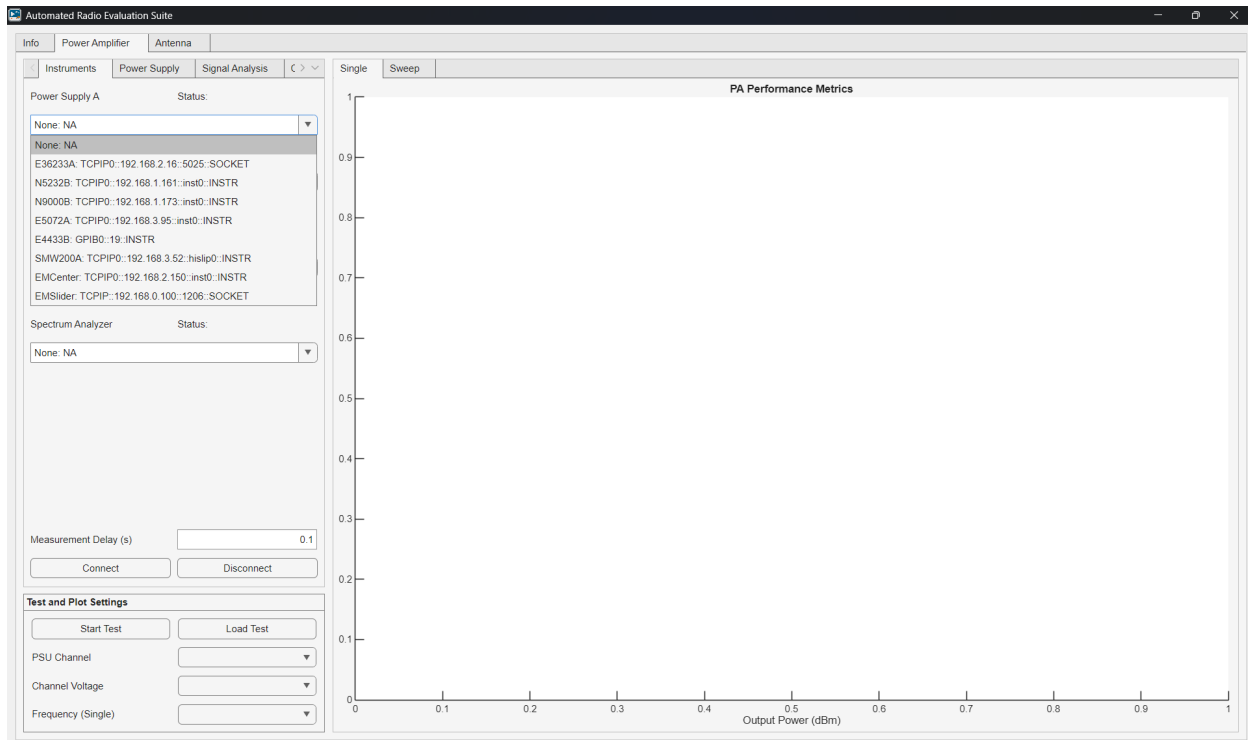
- **Planned Feature:** Support for coupled measurements using two signal analyzers—one measuring the input and the other measuring the output via directional couplers (e.g., -X dB).
- This would enable real-time tracking of gain and distortion with a more accurate measurement path.

3.4.2 Performing the Measurement

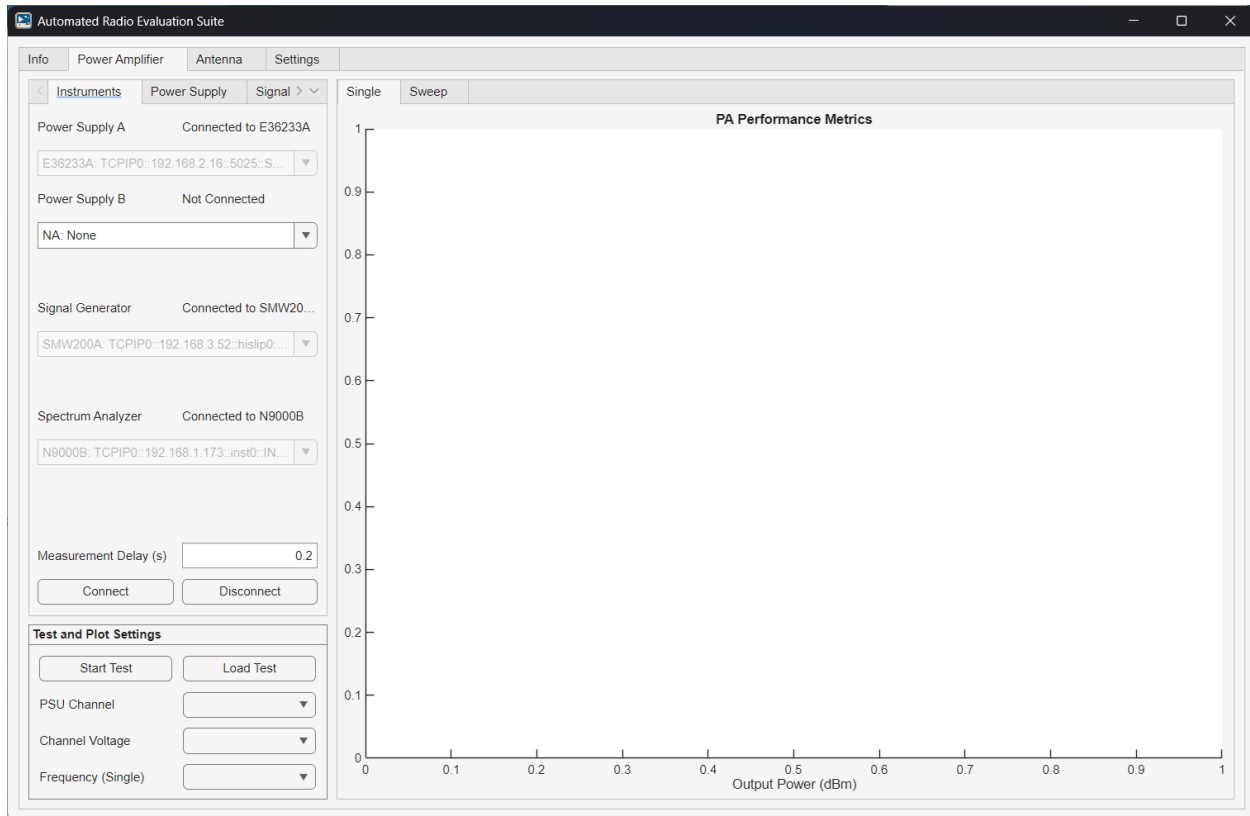
Connect to the Instruments

Power supply selection is optional; however, omitting it will limit which FoMs can be calculated.

- Select the relevant instrument VISA addresses in each dropdown of the *Instruments* tab.
- Select *None: NA* for the instruments that will not be used.
- Follow the [Instrument Database Tutorial](#) for detailed information on how to edit the user-defined instrument database.
- Once all relevant instrument addresses have been populated, click on *Connect* at the bottom to establish a connection to each instrument and *Disconnect* to clear all connections.



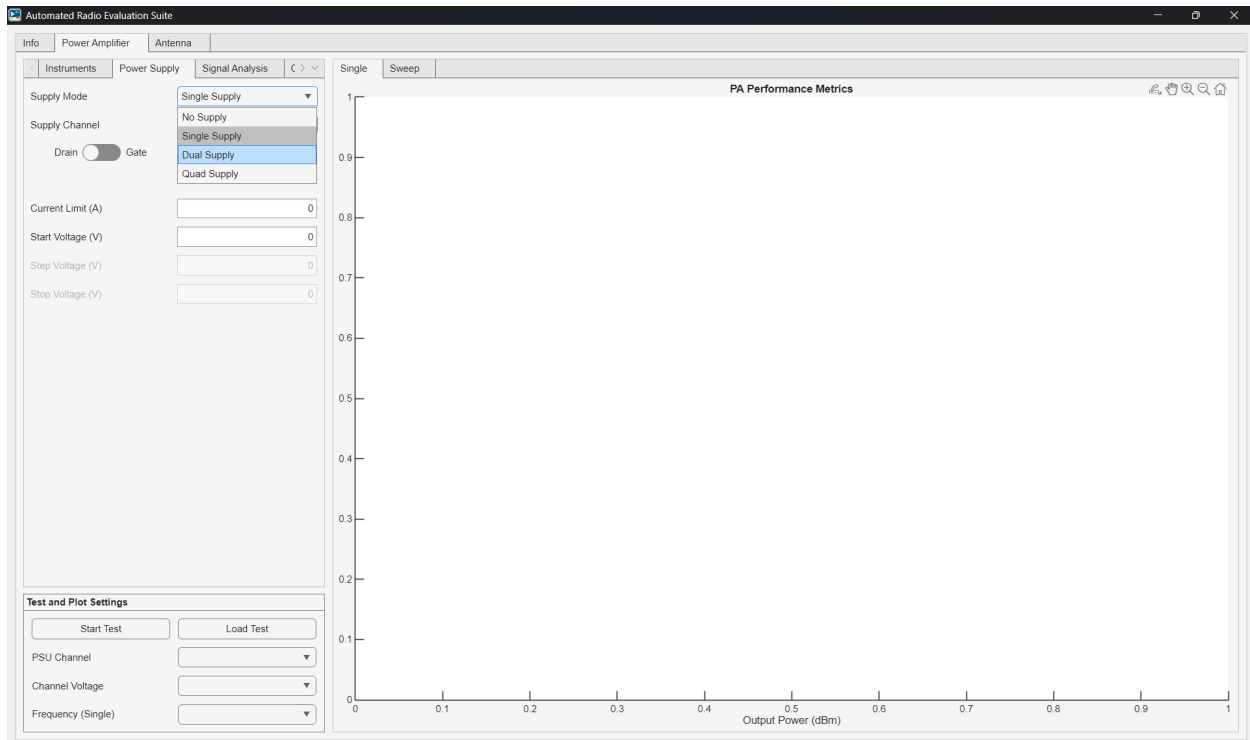
The *Measurement Delay* (s) can be modified at any time before the measurement starts. This value is the time in seconds to wait between setting all the instruments and capturing the data.



Configure the DC Power Supplies

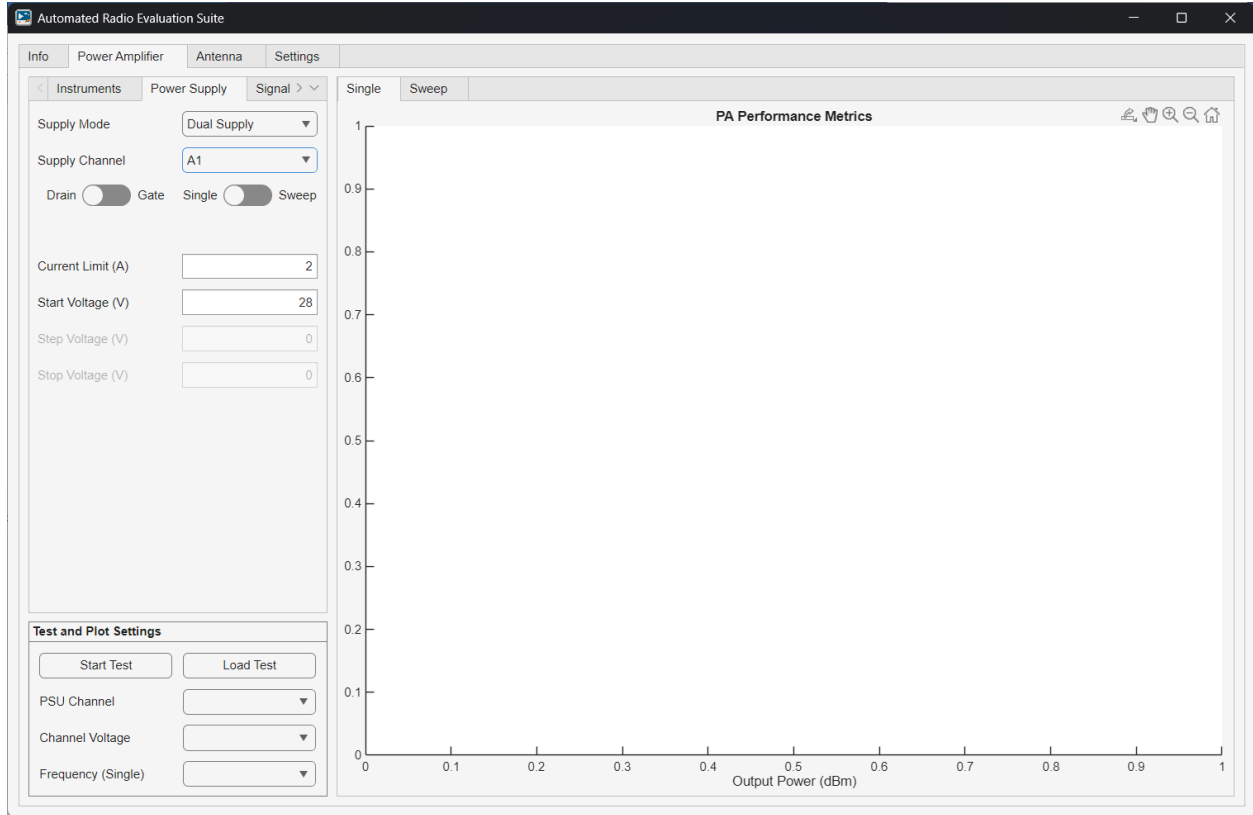
Set up the DC Power Supplies (PSUs) in the *Power Supply* tab. First, choose the appropriate *Supply Mode* based on the number of PSUs connected via the *Instruments* tab. ARES supports up to two dual-output PSUs. The available supply modes are:

- **No Supply:** No control or measurement of any power supply, even if connected.
- **Single Supply:** Uses a single output of one PSU.
- **Dual Supply:** Uses two outputs from one or two PSUs.
- **Quad Supply:** Uses all four outputs from two dual-output PSUs.



Each supply output is labeled with a supply name (A or B) and an output number (1 or 2). Use the *Supply Channel* dropdown to configure each output:

- Assign whether the output is for the drain or gate (for efficiency calculations).
- Choose between a static or swept voltage setting.
- For static outputs, set the *Current Limit* and the *Start Voltage*.
- For swept outputs, also enter the *Step Voltage* and *Stop Voltage*.



Configure the Signal Generator and Analyzer

Set up the RF input signal in the *Signal Analysis* tab.

- Select the *Measurement Type* — choose between *Single* or *Swept Frequencies* in the dropdown.

In the *Signal Generator Settings* Section:

- Define the frequency sweep (if applicable) by filling out the *Start Frequency*, *Step Frequency*, and *Stop Frequency* according to the measurement type.
- Define the RF input power sweep by filling out the *Start Power*, *Step Power*, and *Stop Power*.

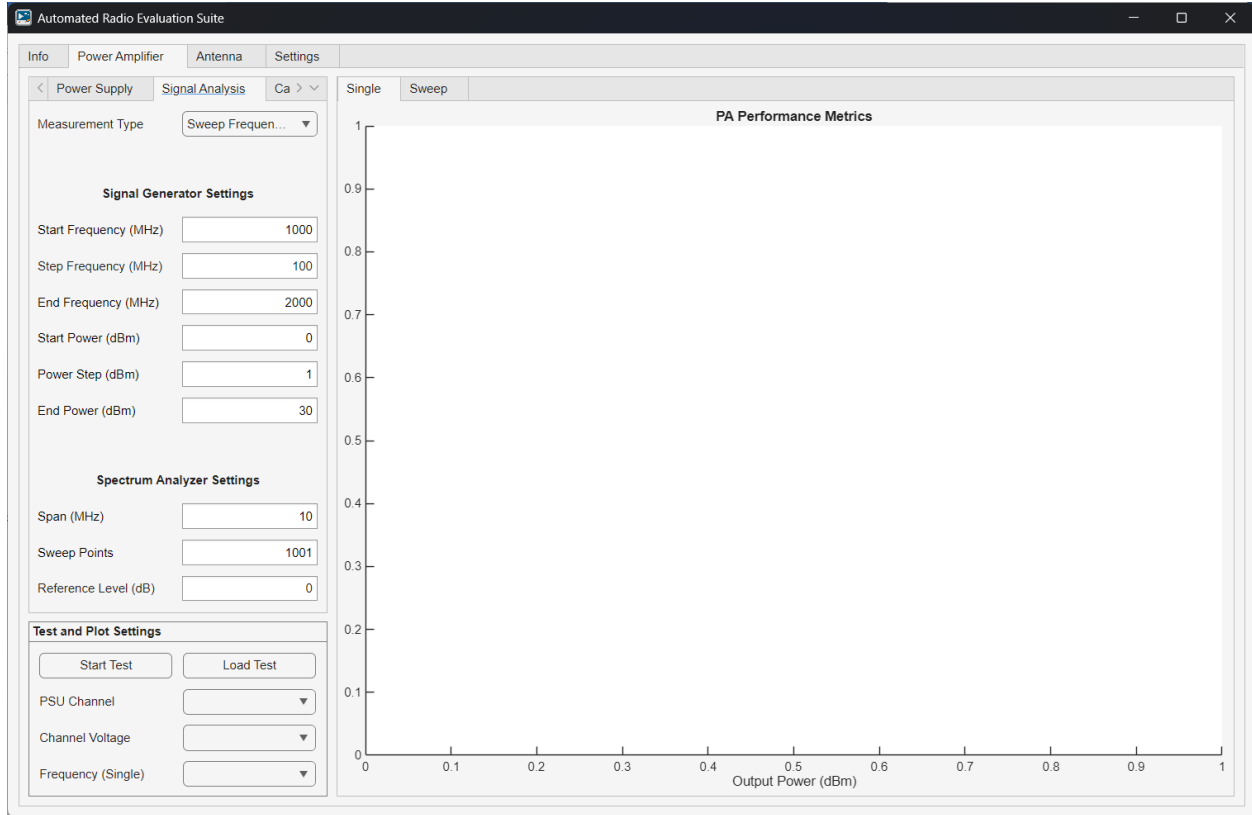
Caution

These settings are limited to the capabilities of the instruments (i.e., frequency and power range).

In the *Signal Analyzer Settings* Section:

- Define an appropriate *Span*.
- Define the number of *Sweep Points*.
- Define the *Reference Level* as needed.

Default values should be feasible for most constant-wave (CW) measurements. The app will automatically change the center frequency of the signal analyzer to the same frequency as the signal generator.

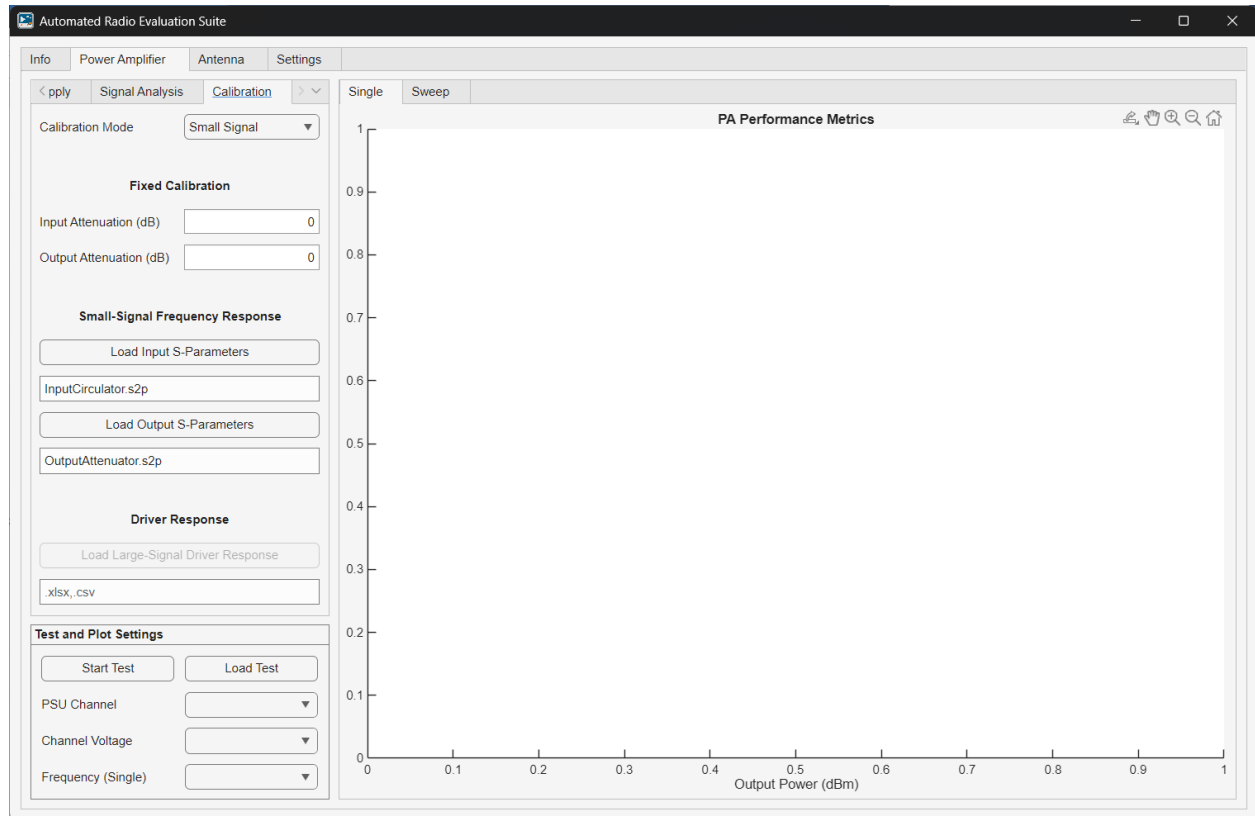


Configure the Measurement Calibration

The *Calibration* tab allows you to optionally account for attenuation and external gain stages.

Use the *Calibration Mode* dropdown to choose one of the following options:

- **None:** No calibration will be applied.
- **Fixed:** Specify static input and output attenuation in decibels, which will be applied equally to all frequencies. Used for passives (i.e., cables, attenuators, etc.).
- **Small Signal:** Load an s-parameter measurement (*.s2p file) for the input and output to specify a frequency-dependent attenuation. This is used for passives (i.e., cables, attenuators, etc.). The fixed attenuation options are still available, and they will be added to the s-parameter attenuations.
- **Large Signal:** In addition to the small-signal calibration options, load a driver amplifier measurement following the same data format in the app. The DUT input power will be calculated by adding the input attenuation and grabbing the output power of the driver at this input power and frequency. Intermediary values are linearly interpolated.

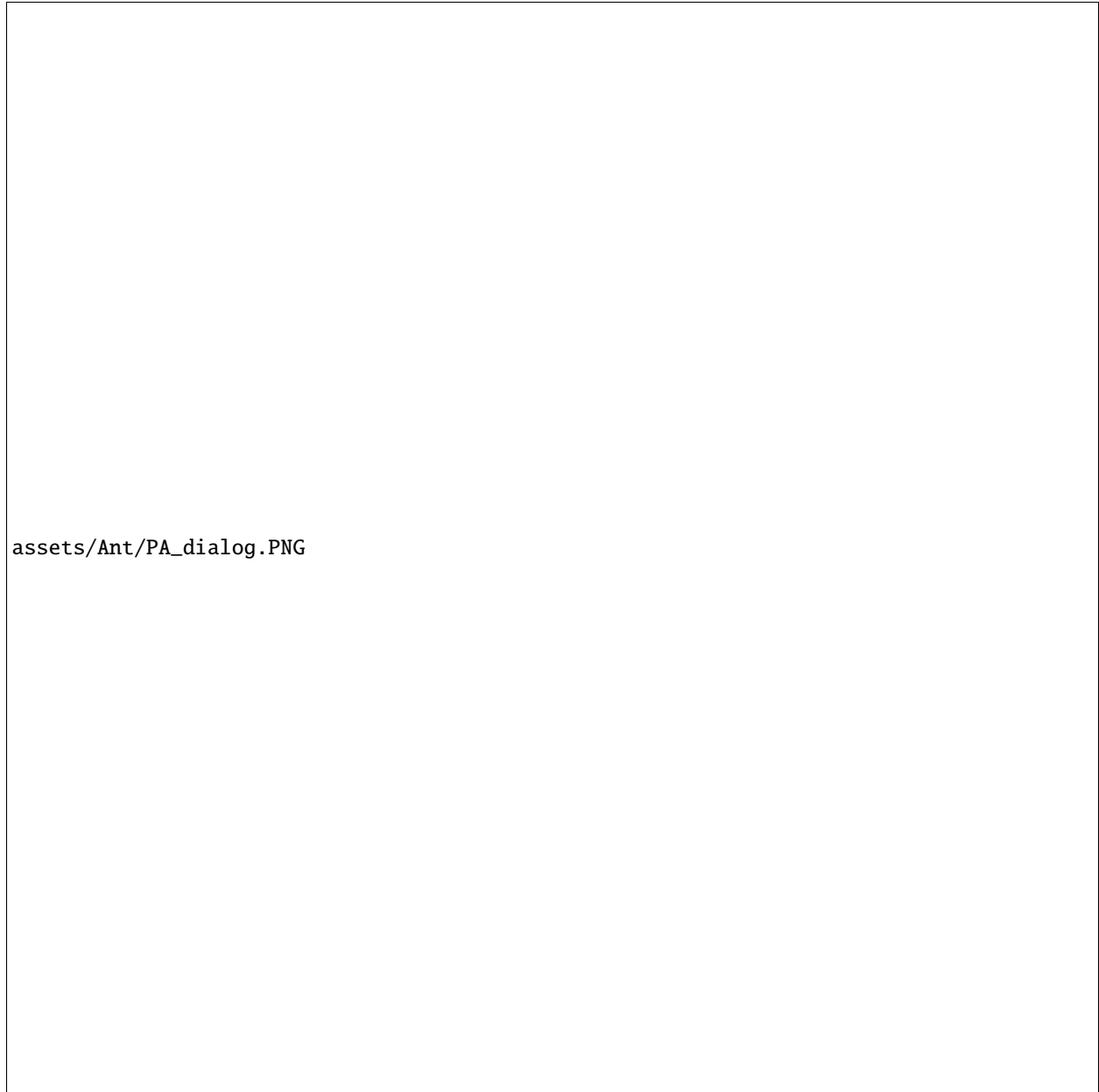


Run the Test and Plot the Results

After verifying all configuration settings, click *Start Test* to begin the measurement.

During the test:

- A progress window will show elapsed time and estimated completion time.
- To terminate the test, press the *Stop Test* button on the progress window.
- Once the test is finished, a prompt will appear to save the results.



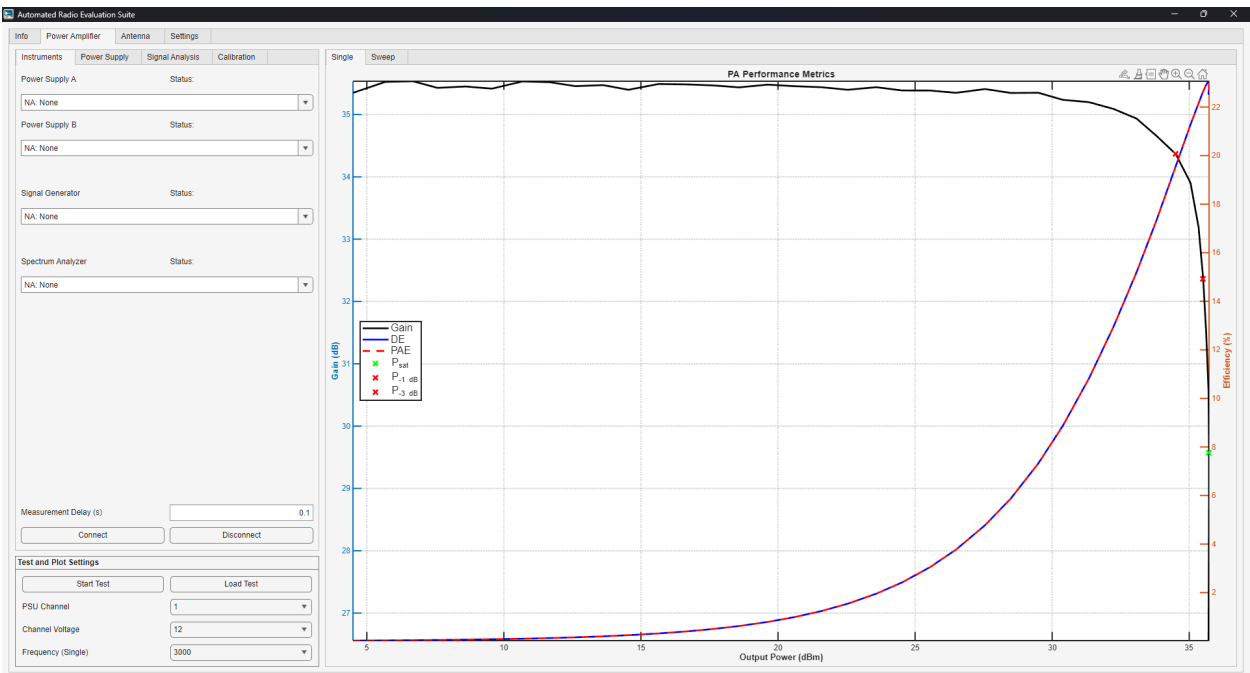
assets/Ant/PA_dialog.PNG

After saving your results:

- The measurement data is automatically loaded and visualized by ARES.
- To view previous results, use the *Load Test* button.

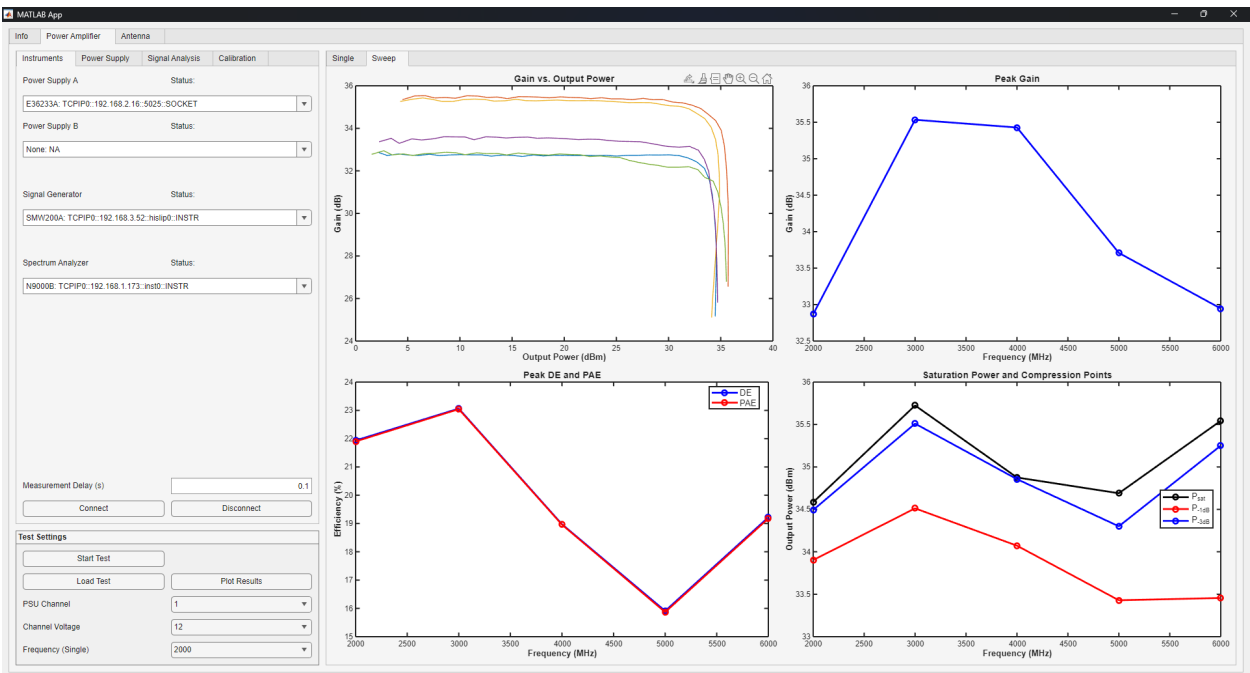
Single Frequency Results

In the *Single* results view window, use the *Frequency* and *PSU Channel* dropdowns to select which voltage configuration and channel to plot. This view shows gain, efficiency, and compression points for the selected parameters.



Sweep Frequencies Results

The *Sweep* results view window displays all gain curves, peak gain, efficiency, and compression points across all measured frequencies. Right-click on any plot to export it in the preferred format.



CODE REFERENCE

4.1 Source Code Overview

The full source code for ARES is organized within the `src` directory. This section provides a high-level overview of the directory structure and key components.

4.1.1 Main Application

`ARES.mlapp`

The main application file contains the graphical user interface (GUI) and core app logic. This main file defines:

- A tab-based interface layout.
- Laboratory instrument connection workflows.
- Measurement test execution for both antenna and PA modules
- Plotting, exporting, and data handling behavior.

`instrument_address_history.txt`

A simple text file that logs laboratory instrument addresses (VISA resource strings).

- ARES reads this file upon startup to populate instrument dropdowns with addresses.
- You can edit or manage this list via the in-app Instrument Database interface.

4.1.2 Support Modules

Located in the `src/support` directory, the following submodules contain reusable measurement and processing logic:

Antenna Functions

Functions focused on antenna measurements, including:

- Gain calculations (Comparison & Transfer methods)
- S-parameter acquisition and interpretation
- Radiation pattern plotting (2D & 3D)
- Linear slider and rotator control

- Enabling and validating the antenna test setup
- Main measurement function for parametric sweeps, data collection, and result plotting

PA Functions

Functions focused on power amplifier (PA) characterization:

- Measuring DC power, RF power, gain, and efficiency
- DC power supply control
- Enabling and validating power supply unit channels
- Peak gain, peak efficiency, compression points, and saturation analysis
- PA deembedding
- Main measurement function for parametric sweeps, data collection, and result plotting

Support Functions

Shared utility functions used by both measurement modules:

- Unit conversions (e.g., dBm Watts, linear dB)
- Measurement data saving/loading
- UI and axes formatting
- Instrument wait timing

4.1.3 Sample Data

Located in the **data** folder, this directory provides example measurement results you can load directly into ARES for visualization or testing.

Antenna Data:

- Contains sample data specific to antenna measurements. Includes example gain, return loss, and radiation pattern datasets.

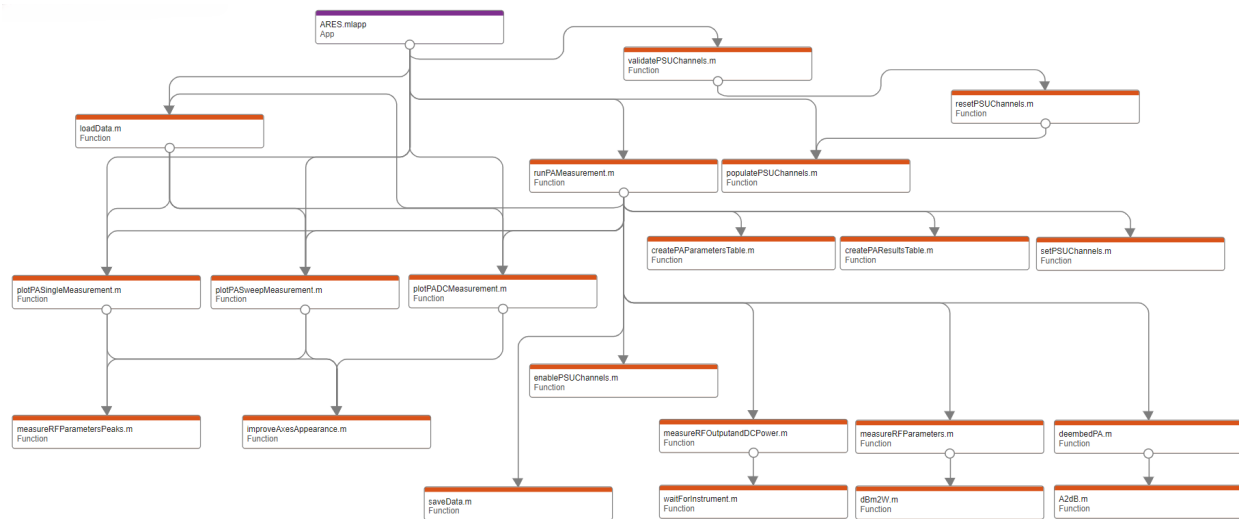
PA Data:

- Contains sample data related to power amplifier measurements. Includes example gain, efficiency, compression analysis, DC current, and DC power analysis.

You can reference these files and directories to build custom scripts or adapt the existing functionality to suit your specific needs.

4.2 App Overview

4.2.1 Internal Architecture – PA Measurement Flow



This diagram was generated using MATLAB's Dependency Analyzer and shows the internal function call structure for the Power Amplifier (PA) module in ARES.

At the top level, `ARES.mlapp` connects to the main controller functions, like:

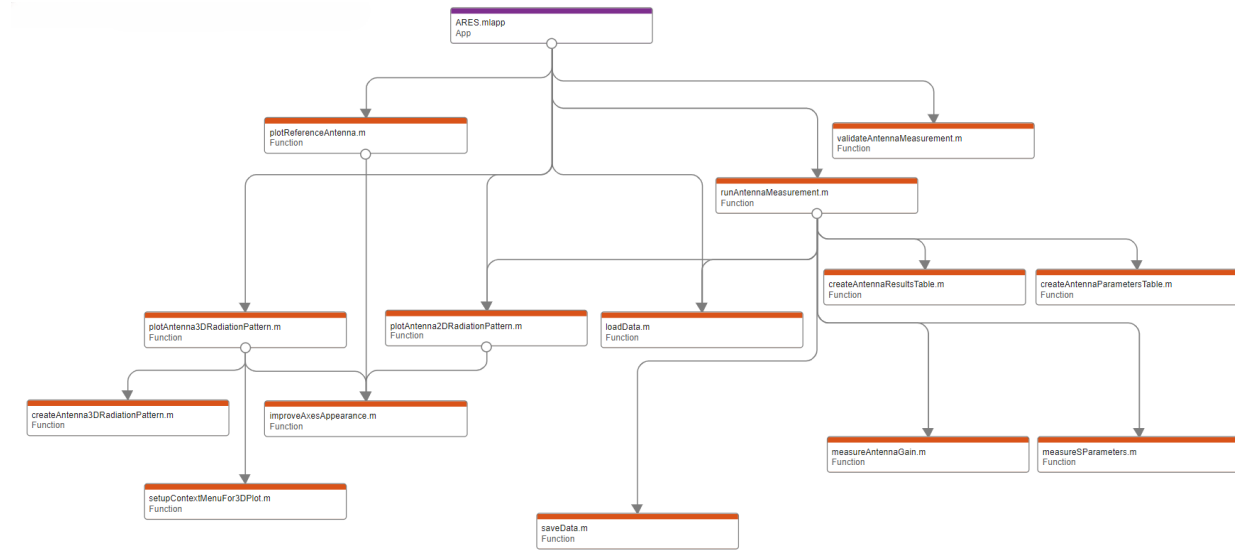
- `runPAMeasurement.m`: The central function for a PA test.
- `validatePSUChannels.m` and `populatePSUChannels.m`: Used to configure the power supplies.
- `loadData.m`: Handles importing previously saved measurements.

From there, the process branches into:

- **DC supply handling**: `enablePSUChannels.m`, `setPSUChannels.m`
- **Data preparation**: `createPAParametersTable.m`, `createPAResultsTable.m`
- **Data capture and analysis**: `measureRFParameters.m`, `measureRFParametersPeaks`, `measureRFOutputandDCPower.m`, and `deembedPA.m`
- **Plotting and visualization**: `plotPASingleMeasurement.m`, `plotPASweepMeasurement.m`, and `plotPADCMMeasurement.m`
- **Plot enhancements**: `improveAxesAppearance.m`

All results are ultimately passed to `saveData.m`.

4.2.2 Internal Architecture – Antenna Measurement Flow



This diagram was generated using MATLAB's Dependency Analyzer and shows the internal function call structure for the Antenna module in ARES.

At the top level, `ARES.mlapp` connects to the main controller functions, like:

- `runAntennaMeasurement.m`: The central function for an antenna test.
- `validateAntennaMeasurement`: Used to validate the measurement parameters before running the test.
- `loadData.m`: Handles importing previously saved measurements.

From there, the process branches into:

- **Data preparation**: `createAntennaParametersTable.m`, `createAntennaResultsTable.m`
- **Data capture and analysis**: `measureAntennaGain.m`, `measureSParameters`
- **Plotting and visualization**: `plotReferenceAntenna.m`, `plotAntenna2DRadiationPattern.m`, `createAntenna3DRadiationPattern`, and `plotAntenna3DRadiationPattern.m`
- **Plot enhancements**: `improveAxesAppearance.m`, `setupContextMenuFor3DPlot.m`

All results are ultimately passed to `saveData.m`.

4.3 Instrument Interfacing

This guide explains how the Automated Radio Evaluation Suite (ARES) communicates with lab instruments using standard APIs and SCPI commands. Whether you're measuring power amplifier (PA) performance or characterizing antennas, this section helps you interface with your instruments through MATLAB using VISA libraries.

4.3.1 VISA Library Overview

Virtual Instrument Standard Architecture (VISA) is a test and measurement industry-standard communication API (Application Programming Interface) for use with test and measurement devices regardless of their communication protocol (e.g., LAN, USB, GPIB, etc.). The VISA libraries are the communication drivers that allow ARES to control the instruments using Standard Commands for Programmable Instruments (SCPI).

ARES uses the VISAdev interface, introduced in MATLAB R2021a. VISAdev replaces the older VISA interface and integrates better with modern instrument drivers. Read more about the necessary software in [MATLAB: Get Started with VISA](#).

4.3.2 SCPI Command Basics

Standard Commands for Programmable Instruments (SCPI) are text-based commands used to configure and query instruments. These commands are device-independent and can be used over any supported communication interface.

[Keysight Command Expert](#) allows searching the necessary SCPI strings interactively and provides seamless integrations with different VISA libraries.

There are two types of SCPI strings:

- **Command String:** :KEY <parameter>
- **Query String:** :KEY?

Command strings are reserved for writing to the instrument, while query strings are reserved for reading back from the instrument. The KEY represents the name of the instrument feature to adjust, while ‘ ’ is the value to be sent to the instrument. Complex SCPI strings can be made by combining several KEYs and parameters into one string. You can chain multiple levels of control using colons, i.e., :KEY1:KEY2:KEY3 <parameter 1> <parameter 2>.

Transitioning from VISA to VISAdev

ARES uses the visadev interface instead of the older visa interface. Here’s a summary of equivalent MATLAB commands:

VISA Interface	VISAdev Interface	Purpose
visa()	visadev()	Connect to an instrument.
fprintf()	writeline()	Send a command.
fscanf()	readline()	Read a response.
query()	writeread()	Send a query and read the response.
binblockread()	readbinblock()	Read binary data blocks.
clrdevice()	flush()	Flush I/O data buffers.
fclose()	clear()	Disconnect the instrument object.

Common SCPI Strings Across Instruments

Many instruments share a basic set of SCPI commands, regardless of vendor or type. These are especially helpful for setup, synchronization, and debugging.

Type	SCPI Command/Query	Purpose	When to Use
Identification Instrument	*IDN?	Returns instrument model and info.	Verify the instrument after connecting.
Reset to Default State	*RST	Resets instrument to factory defaults.	Use at the start of the session unless your instrument uses a preconfigured environment.
Clear Status Register	*CLS	Clears error and communication flags.	Prevent corrupted data in sweeps.
Operation Complete	*OPC?	Queries if previous commands finished	For command debugging.
Wait for Completion	*WAI	Waits until prior operations are complete.	Improve speed/accuracy.

SCPI Data Format Commands

The following commands control how numerical data is formatted and transmitted between the instrument and controller, impacting compatibility, speed, and precision during data transfer.

Type	SCPI Command/Query	Purpose	When to Use
Set Byte Order	:FORM:BORD: SWAP	Configures Little-endian binary format.	Required by MATLAB for binary reads.
Set Byte Order	:FORM:BORD: NORM	Configures Big-endian binary format.	Used in other environments (non-MATLAB).
Query Byte Order	:FORM:BORD?	Queries current byte order.	Verifying before reading binary data.
Set Data Format	:FORM:DATA ASCII,0	Sets ASCII data format.	Useful for debugging, not efficient.
Set Data Format	:FORM:DATA REAL, 32	Sets 32-bit real binary format.	Faster transfer, sufficient for most tasks.
Set Data Format	:FORM:DATA REAL, 64	Sets 64-bit real binary format.	Highest precision, use for double-precision.
Query Data Format	:FORM:DATA?	Queries current data format.	Verifying before reading trace/sample data.

Sample SCPI Commands for PA Measurement Instruments

The following table lists frequently used SCPI commands for configuring and controlling instruments in the power amplifier (PA) measurement setup. These commands help automate voltage/current sourcing, RF signal generation, and signal analysis tasks.

Instrument	Command / Query	SCPI String	Input Parameter
Power Supply	Set voltage and current	:APPLY <channel>, <voltage>, <current>	Voltage and current within instrument limits.
Power Supply	Measure DC voltage	:MEAS:SCAL:VOLT:DC? <channel>	Use @1, @2, or @1, 2 to specify channel(s).
Power Supply	Measure DC current	:MEAS:SCAL:CURR:DC? <channel>	Use @1, @2, or @1, 2 to specify channel(s).
Power Supply	Enable or disable output	:OUTP:STAT <state>, (<channel>)	ON/OFF or 1/0. Channel in parentheses: (@1), (@2), or (@1, 2).
Signal Generator	Set RF power	:SOUR1:POW:LEV:IMM:AMPL <power>	Power in dBm, within allowed range.
Signal Generator	Set frequency	:SOUR1:FREQ:CW <frequency>	Frequency in Hz, within allowed range.
Signal Generator	Enable or disable output	:OUTP1:STAT <state>	ON/OFF or 1/0.
Signal Analyzer	Set number of sweep points	:SENS:SWE:POIN <points>	Integer between 1 and the maximum supported value..
Signal Analyzer	Set frequency span	:SENS:FREQ:SPAN 	Span in Hz, within valid range.
Signal Analyzer	Set center frequency	:SENS:FREQ:CENT <frequency>	Center frequency in Hz, within valid range.
Signal Analyzer	Enable continuous measurement	:INIT:CONT <state>	ON/OFF or 1/0.
Signal Analyzer	Restart current sweep	:INIT:IMM	No input. Immediately starts or restarts a sweep.
Signal Analyzer	Query sweep data	:TRAC:DATA? <trace>	Trace name, e.g., TRAC1, TRAC2, ..., TRAC6.

For the Power Supply Unit (PSU):

- **channel**: Use @1 for channel 1, @2 for channel 2, and @1, 2 to address both simultaneously.

Sample SCPI Commands for Antenna Measurement Instruments

The following table lists frequently used SCPI commands for configuring and controlling instruments in the antenna measurement setup. These commands allow automation of RF sweeps, mechanical motion, and data acquisition during the antenna measurements.

Instrument	Command Query	SCPI String	Input Parameter
VNA	Set start frequency	:SENS<cnum>:FREQ:START<num>	Frequency in Hz; within VNA's allowed range.
VNA	Set stop frequency	:SENS<cnum>:FREQ:STOP<num>	Frequency in Hz; minimum is often 70 Hz, depending on model.
VNA	Set sweep points	:SENS<cnum>:SWE:POIN<num>	Integer between 1 and max supported points.
VNA	Set sweep mode	:SENS<cnum>:SWE:MODE<mode>	Mode can be HOLD, CONT (continuous), or SING (single).
VNA	Enable smoothing	:CALC<cnum>:SMO:STAT<state>	ON/OFF or 1/0.
VNA	Set smoothing aperture	:CALC<cnum>:SMO:APER<num>	Percentage between 1 and 25.
EM Center	Set rotation speed	1<slot-letter>:SPEED<speed>	Speed between 1 and 100.
EM Center	Seek to angle	1<slot-letter>:SK<angle>	Angle in degrees.
EM Center	Stop movement	1<slot-letter>:ST	No parameters. Immediately halts motion.
Linear Slider	Set speed	AXIS1:S <speed>	Speed preset from 1 (slowest) to 8 (fastest).
Linear Slider	Seek to position	AXIS1:SK <position>	Position in cm (within travel limits).
Linear Slider	Query current position	AXIS1:CP?	Returns current position in cm.
Linear Slider	Enable scan mode	AXIS1:SCAN	No parameters. Starts programmed scan routine.
Linear Slider	Enable homing mode	AXIS1:HOME	No parameters. Returns the axis to mechanical zero.

For the Vector Network Analyzer (VNA):

- **cnum** is the channel number, which defaults to one if not specified.
- **mnum** is the measurement number, and **mname** is the name of the measurement.

For the EM Center, which controls the anechoic chamber:

- Slot 1 contains both tower and table actuators.
- Use A for the table and B for the tower as slot-letter designations (e.g., 1A, 1B).

4.4 Antenna Functions

4.4.1 createAntenna3DRadiationPattern.m

Path: src\support\AntennaFunctions\createAntenna3DRadiationPattern.m

Description:

This function generates a 3D radiation pattern plot for antennas in App Designer environments. It creates a visually informative 3D representation of radiation patterns without dependencies on internal MATLAB functions. The function:

- Renders a 3D surface representation of radiation pattern data.
- Draws reference coordinate system with x, y, z axes and plane indicators.
- Provides visual indicators for azimuth and elevation angles.
- Supports proper scaling and normalization of magnitude data.

Input Parameters

- axes - Target UIAxes object where the pattern will be rendered.
- Magnitude - Matrix of magnitude values (dBi) corresponding to the pattern.
- Theta - Vector of θ angles in degrees (elevation angle).
- Phi - Vector of ϕ angles in degrees (azimuth angle).

Output Parameters

- None. The function modifies the provided axes object directly.

4.4.2 createAntennaParametersTable.m

Path: src\support\AntennaFunctions\createAntennaParametersTable.m

Description:

This function generates a table of all possible combinations of θ and ϕ angles for antenna testing. The function ensures that the input angles are properly processed and sorted for efficient use in antenna measurements.

Input Parameters

- Theta - A vector of theta angles (in degrees). The angles can range from -180 to 180.
- Phi - A vector of phi angles (in degrees). The angles can range from -180 to 180.

i Output Parameters

- ParametersTable - A table containing all combinations of Theta in (degrees) and Phi in (degrees).

4.4.3 createAntennaResultsTable.m

Path: `src\support\AntennaFunctions\createAntennaResultsTable.m`

Description:

This function initializes and preallocates a results table for storing antenna test measurements. The results table is designed to hold various antenna parameters for a given number of measurements. It contains the following columns:

- Theta (deg)
- Phi (deg)
- Frequency (MHz)
- Gain (dBi)
- Return Loss (dB)
- Return Loss (deg)
- Return Loss Reference (dB)
- Return Loss Reference (deg)
- Path Loss (dB)
- Path Loss (deg)

i Input Parameters

- totalMeasurements - The total number of measurements (rows) to allocate in the results table.

i Output Parameters

- ResultsTable - The preallocated table.

4.4.4 measureAntennaGain.m

Path: `src\support\AntennaFunctions\measureAntennaGain.m`

Description:

This function calculates the gain of a test antenna in decibels relative to an isotropic radiator (dBi). The gain is computed based on the input test frequency, S-parameters, and the spacing between the antennas. The function calculates the antenna gain using one of two methods:

- **Comparison Antenna Method:** If reference gain and frequency values are provided, the antenna gain is calculated by interpolating the reference gain at the test frequencies.
- **Two Antenna Method:** If no reference data is provided, the function assumes the test antennas are identical.

i Input Parameters

- **TestFrequency** - A scalar or vector of frequency values in Hz at which the antenna gain is measured.
- **sParameter_dB** - A scalar or vector of S21 values (in dB), representing the magnitude of power transfer between two antennas.
- **Spacing** - A scalar value representing the distance in meters between the two antennas being tested.
- **ReferenceGain** - (Optional) A vector of reference antenna gain values (in dBi). Used for the Comparison Antenna Method.
- **ReferenceFrequency** - (Optional) A vector of frequencies (in Hz) corresponding to the reference antenna gain values.

i Output Parameters

- **antennaGain** - A vector containing the calculated antenna gain in dBi for the test antenna, at the specified test frequencies.

4.4.5 measureSParameters.m

Path: src\support\AntennaFunctions\measureSParameters.m

Description:

This function measures 2-port S-parameters (S11, S21, S22) with magnitude in dB and phase in degrees using a Vector Network Analyzer (VNA). Depending on the **smoothingPercentage** input, the function reads either smoothed or raw measurement data.

- **Smoothed Data:** If smoothing is enabled (**smoothingPercentage** > 0), the function retrieves the smoothed magnitude and phase data.
- **Raw Data:** If smoothing is disabled (**smoothingPercentage** = 0), the function retrieves raw data in the form of complex S Parameters and calculates the magnitude and phase from the complex data.

i Input Parameters

- **VNA** - The instrument object for the VNA, used for communication and measurement control.
- **smoothingPercentage** - The percentage of smoothing applied to the S-parameters data.

i Output Parameters

- **sParamdB** - A cell array containing the magnitude data (in dB) for each S-parameter.
- **sParamPhase** - A cell array containing the phase data (in degrees) for each S-parameter.

- freqValues - A vector containing the frequency sweep values (in Hz) corresponding to the S-parameters.
-

4.4.6 plotAntenna2DRadiationPattern.m

Path: src\support\AntennaFunctions\plotAntenna2DRadiationPattern.m

Description:

This function generates and displays several 2D plots related to antenna measurements. It extracts the relevant antenna data based on user-selected θ , ϕ , and frequency values. It updates four axes in the application UI to display the following plots:

- Gain vs. Frequency at a fixed θ/ϕ angle.
- Gain vs. Angle (θ and ϕ cuts) at a fixed frequency.
- Return Loss vs. Frequency at a fixed θ/ϕ angle.
- Polar Radiation Pattern (θ and ϕ cuts).

Input Parameters

- app - Application object containing the antenna measurement data and plot handles.

Output Parameters

- None
-

4.4.7 plotAntenna3DRadiationPattern.m

Path: src\support\AntennaFunctions\plotAntenna3DRadiationPattern.m

Description:

This function generates a 3D radiation pattern plot for the antenna based on the specified frequency. It creates a 3D visualization of the antenna's radiation characteristics using theta, phi, and gain values from the application data. The function:

- Extracts the antenna gain data for the selected frequency
- Processes angle data for consistent representation
- Creates a properly formatted gain matrix
- Renders the 3D radiation pattern with appropriate visual elements
- Displays appropriate error or warning messages if data is inconsistent or invalid.

Input Parameters

- app - Application object containing the antenna measurement data and UI components.

i Output Parameters

- None - The function updates the application's UI components directly.

4.4.8 plotReferenceAntenna.m

Path: `src\support\AntennaFunctions\plotReferenceAntenna.m`

Description:

Plots the reference antenna's gain and return loss versus frequency, serving as a baseline for comparison with DUT (Device Under Test) measurements. The function:

- Clears existing plots for gain and return loss.
- Plots gain (dBi) vs frequency (MHz).
- Plots return loss (dB) vs frequency (MHz).
- Enhances visual appearance of plots using standardized formatting.

i Input Parameters

- app - Application object containing the reference antenna measurement data and associated plot handles.

i Output Parameters

- None

4.4.9 runAntennaMeasurement.m

Path: `src\support\AntennaFunctions\runAntennaMeasurement.m`

Description:

This function executes a 2D antenna gain measurement sweep using a dual-axis positioner (Theta and Phi) and a VNA. The function automates rotation, measurement, data logging, and visualization of antenna radiation patterns. The sweep is performed across a user-defined frequency range and angular grid, capturing S-parameters and calculating gain, return loss, and path loss. Data is saved to the disk and loaded into the app for plotting. Additionally the user is able to manually stop the test run from within the application. If an error occurs:

- The errors are caught, reported via the app interface, and saved to the error log.
- Positioners are safely stopped in case of interruption or failure.

i Input Parameters

- app - Application object that holds hardware interfaces, user settings, UI elements, and measurement parameters.

i Output Parameters

- None (Results are saved to the user's machine and updated in the application UI).

4.4.10 setLinearSlider.m

Path: `src\support\AntennaFunctions\setLinearSlider.m`

Description:

This function controls the movement of the EMCenter linear slider by setting its speed preset and moving it to a user-specified target position. Once the speed is set, the slider will move smoothly to the specified target position, ensuring precise control over the motion.

i Input Parameters

- `speedPreset` - An integer between 1 (slowest) and 8 (fastest) that sets the speed of the linear slider.
- `targetPosition` - Target position in cm (0 - 200 cm) for the slider to move to.

i Output Parameters

- None

4.4.11 validateAntennaMeasurement.m

Path: `src\support\AntennaFunctions\validateAntennaMeasurement.m`

Description:

This function performs the following validation checks on the antenna test setup. If any condition is not met, the function displays an appropriate message and prompts the user to correct the configuration.

- Whether the VNA, EMCenter, and EMSlider are connected.
- Whether the start and end frequencies are specified and not equal.
- Whether the number of sweep points is defined.
- Whether the antenna's physical size is specified.
- Whether the turntable scan settings (mode, start angle, step angle, end angle) are configured.
- Whether the tower scan settings (mode, start angle, step angle, end angle) are configured.

i Input Parameters

- `app` - Application object containing the antenna setup configuration.

i Output Parameters

- isValid - Logical value. true if the configuration is valid; otherwise, false.

4.5 Power Amplifier Functions

4.5.1 createPAParametersTable.m

Path: src\support\PAFunctions\createPAParametersTable.m

Description:

This function generates a complete parameter sweep table for Power Amplifier (PA) testing. It creates all possible combinations of frequency, RF input power, voltage, and current based on the sweep settings defined in the application for each active PSU channel. The resulting table is used to drive automated PA characterization across various operating conditions, it contains the following columns:

- Frequency (Hz)
- RF Input Power (dBm)
- Voltage (V) and Current (A) per active PSU channel

i Input Parameters

- app - App object containing configuration parameters for frequency, RF input power, and power supply settings.

i Output Parameters

- ParametersTable - The resulting parameters table containing all combinations of PA test settings.

4.5.2 createPAResultsTable.m

Path: src\support\PAFunctions\createPAResultsTable.m

Description:

This function initializes an empty results table for Power Amplifier (PA) measurements, with columns dynamically generated based on the number of active PSU channels configured in the application. The table is pre-allocated to the specified number of measurements and channels. For n channels:

- Frequency (MHz)
- Channel 1 Voltage (V) (if n = 1)
- ...
- Channel n Voltages (V)

- RF Input Power (dBm)
- RF Output Power (dBm)
- Gain
- Channel 1 DC Current (A) (if n = 1)
- ...
- Channel n DC Current (A)
- Channel 1 DC Power (W) (if n = 1)
- ...
- Channel n DC Power (W)
- Total DC Drain Current (A)
- Total DC Gain Current (A)
- Total DC Drain Power (W)
- Total DC Gate Power (W)
- DE (%)
- PAE (%)

Input Parameters

- app - The app object containing configuration details, including active PSU channels.
- totalMeasurements - Total number of rows to preallocate in the results table.

Output Parameters

- ResultsTable - An empty table with predefined variable names and types for storing PA test results.

4.5.3 deembedPA.m

Path: src\support\PAFunctions\deembedPA.m

Description:

This function de-embeds Power Amplifier (PA) measurements by removing the effects of passive and active devices. It generates calibration factors for both the input and output of the PA, which are applied to the measured RF power values in order to obtain the corrected PA input and output RF power. The calibration factors are computed based on the selected calibration mode and available data on the app. The function supports the following calibration modes:

- **None:** No calibration is applied, and both input and output calibration factors are set to 0.
- **Fixed Attenuation:** The function directly applies the attenuation values set in the application for both input and output.
- **Small Signal:** Uses fixed attenuation values combined with interpolated Sparameters from the provided Sparameter file for both input and output.

- **Small + Large Signal:** Same behavior as **Small Signal** but also integrates driver gain data. The driver gain is interpolated based on both the test frequency and RF input power, which is then subtracted from the input calibration factor.

i Input Parameters

- app - The application object containing configuration settings and attenuation values for the PA setup.
- testFrequency - The measurement frequency (Hz) at which the calibration and de-embedding should be performed.
- RFInputPower - The measured RF input power (dBm) at the specified frequency.

i Output Parameters

- inCal - The input attenuation calibration factor (dB). Subtracts this from the input RF power to obtain the corrected PA input power.
- outCal - The output attenuation calibration factor (dB). Adds this to the measured output power to get the corrected PA output power.

4.5.4 enablePSUChannels.m

Path: src\support\PAFunctions\enablePSUChannels.m

Description:

This function enables or disables the specified channels on two power supply units (PSU A and PSU B) based on the provided state (1 for enabling, 0 for disabling). Channels are grouped by PSU (PSU A or PSU B) and then enabled or disabled accordingly. The order in which the channels are processed depends on the state: gate biases are controlled before drain supplies when enabling, and drain supplies are turned off before gate biases when disabling.

i Input Parameters

- app - The application object containing the power supplies and the channel-to-device mapping.
- channels - A cell array of channel names (e.g., {'CH1', 'CH2'}).
- state - Channel state (1 for enable, 0 for disable).

i Output Parameters

- None

4.5.5 measureRFOutputandDCPower.m

Path: src\support\PAFunctions\measureRFOutputandDCPower.m

Description:

This function measures the output RF power, DC drain power, and DC gate power based on the specified input RF power and test frequency.

i Input Parameters

- app - The application object containing instrument configurations.
- frequency - The test frequency for measurement.

i Output Parameters

- DCDrainPower - The DC power delivered to the drain (W).
 - DCGatePower - The DC power delivered to the gate (W).
-

4.5.6 measureRFParameters.m

Path: src\support\PAFunctions\measureRFParameters.m

Description:

This function calculates the RF Gain, Drain Efficiency (DE), and Power Added Efficiency (PAE) based on the specified input and output RF power and the DC power supplied to the drain.

i Output Parameters

- DCDrainPower - DC power supplied to the drain (in W).
 - Gain - RF Gain (dB).
 - DE - Drain Efficiency (%).
 - PAE - Power Added Efficiency (%).
-

4.5.7 measureRFParametersPeaks.m

Path: src\support\PAFunctions\measureRFParametersPeaks.m

Description:

This function calculates peak RF performance metrics from power amplifier (PA) measurement data, including saturation power, peak gain, peak drain efficiency (DE), peak power-added efficiency (PAE), and -1 dB and -3 dB compression points.

i Input Parameters

- app - The application object containing the PA measurement data table.
- idx - Logical or numeric index used to filter the rows of the PA data table for analysis.

i Output Parameters

- Psat - Table containing the maximum RF output power (Psat) per frequency and corresponding gain.
- peakGain - Table of peak small-signal gain values per frequency.
- peakDE - Table of maximum drain efficiency per frequency.
- peakPAE - Table of maximum power-added efficiency per frequency.
- compression1dB - Table containing the -1 dB gain compression points per frequency.
- compression3dB - Table containing the -3 dB gain compression points per frequency.

4.5.8 plotPADCMeasurement.m

Path: src\support\PAFunctions\plotPADCMeasurement.m

Description:

This function plots DC performance metrics from a frequency sweep Power Amplifier (PA) measurement, including drain currents (IDD) and drain DC power. It filters the PA dataset using user-selected supply voltages and generates the plots with styled axes and markers for clarity:

- Supply Current vs. Output Power for each frequency
- DC Supply Power vs. Output Power for each frequency
- Peak Drain Current vs. Frequency
- Peak DC Supply Power vs. Frequency

i Input Parameters

- app - Application object containing PA measurement data and plotting components.

i Output Parameters

- None

4.5.9 plotPASingleMeasurement.m

Path: `src\support\PAFunctions\plotPASingleMeasurement.m`

Description:

This function plots gain, drain efficiency (DE), and power-added efficiency (PAE) versus RF output power for a single frequency measurement. Also overlays peak values such as Psat, -1 dB and -3 dB compression points. This function generates a dual y-axis plot:

- Left Y axis: Gain (dB)
 - Green X: Psat (saturation output power)
 - Red X: -1 dB gain compression point
 - Blue X: -3 dB gain compression point
- Right Y axis: DE and PAE (%)

i Input Parameters

- app - Application object containing PA measurement data, user-selected frequency, supply voltages, and plotting handles.

i Output Parameters

- None
-

4.5.10 plotPASweepMeasurement.m

Path: `src\support\PAFunctions\plotPASweepMeasurement.m`

Description:

This function plots RF performance metrics from a frequency sweep Power Amplifier (PA) measurement, including gain, saturation power (Psat), efficiency (DE and PAE), and gain compression points (-1 dB, -3 dB). It filters the PA dataset using user-selected supply voltages and generates four annotated plots with styled axes and markers for clarity:

- Peak Gain vs. Frequency
- Peak Drain Efficiency (DE) and PowerAdded Efficiency (PAE) vs. Frequency
- Psat, 1 dB, and 3 dB compression points vs. Frequency

i Input Parameters

- app - Application object containing PA measurement data and plotting components.

i Output Parameters

- None

4.5.11 populatePSUChannels.m

Path: `src\support\PAFunctions\populatePSUChannels.m`

Description:

This function evaluates the configuration of power supply channels and identifies which channels are “filled,” meaning they have complete user-defined parameters (voltage, current, etc.) and are ready for use. These filled channels are stored in the app object for future processing or interaction with the PSU units.

i Input Parameters

- app - The application object containing the channel configurations and channel-to-device mapping.

i Output Parameters

- None
-

4.5.12 resetPSUChannels.m

Path: `src\support\PAFunctions\resetPSUChannels.m`

Description:

This function resets all power supply unit (PSU) channels to their default settings, including setting voltage and current values to 0 and configuring each channel to ‘Single’ mode. It also refreshes the GUI to reflect these changes and ensures the PA side of the application returns to its initial state.

i Input Parameters

- app - The application object containing the channel configurations.

i Output Parameters

- None
-

4.5.13 runPAMeasurement.m

Path: `src\support\PAFunctions\runPAMeasurement.m`

Description:

This function performs a full RF Power Amplifier (PA) measurement sweep. On error, the instruments are safely turned off, and the error message is displayed in the app and logged to the user path. The function process includes:

- Generating test parameter combinations and initializing the output results table.
- Configuring the signal analyzer and initializing the measurement loop.
- For each test point:
 - Sets frequency and signal levels
 - Configures PSU voltages and currents
 - Measures RF output power and DC power
 - Applies calibration factors (de-embedding)
 - Calculates Gain, DE (Drain Efficiency), and PAE (Power Added Efficiency)
 - Stores results in a structured table
- Providing a progress UI dialog with estimated time updates.
- Saving the results and loading them back into the application.

i Input Parameters

- app - Application object containing hardware interfaces, user settings, and UI components.

i Output Parameters

- None (Results are saved to the user's machine and updated in the application UI).

4.5.14 setPSUChannels.m

Path: `src\support\PAFunctions\setPSUChannels.m`

Description:

This function sets the voltage and current for a specific power supply channel based on the application's channel-to-device mapping. It determines the correct physical channel and PSU (A or B), and applies the specified settings via SCPI commands. The function handles numeric or string inputs for voltage and current values, ensuring compatibility with SCPI command formatting.

i Input Parameters

- app - App object containing power supply configurations and the channel-to-device mapping.
- deviceChannel - Logical channel name (e.g., 'CH1') as defined in the mapping structure.

- voltage - Desired voltage for the channel (numeric or string).
- current - Desired current limit for the channel (numeric or string).

i Output Parameters

- None

4.5.15 validatePSUChannels.m

Path: `src\support\PAFunctions\validatePSUChannels.m`

Description:

This function validates PSU channel configuration based on the selected mode. It checks:

- That devices are connected.
- That the mode matches available devices.
- That enough channels are configured.
- That channel count does not exceed mode limits.

i Input Parameters

- app - App object with mode, device, and channel info

i Output Parameters

- isValid - True if configuration is valid; otherwise false

4.6 Supporting Functions

4.6.1 A2dB.m

Path: `src\support\SupportFunctions\A2dB.m`

Description:

The function A2dB converts magnitudes (voltage or current) to dB.

i Input Parameters

- A - Magnitude (voltage or current) in linear scale

i Output Parameters

- AdB - Magnitude in decibels (dB)

4.6.2 dBm2W.m

Path: `src\support\SupportFunctions\dBm2W.m`

Description:

The function dBm2mag converts dBm to Watts (W).

i Input Parameters

- PdBm - Power in (dBm)

i Output Parameters

- P - Power in (W)

4.6.3 extractDocs.m

Path: `src\support\SupportFunctions\extractDocs.m`

Description:

Extracts documentation from .m files within a given folder and writes it to a Markdown file. Designed to support ReadTheDocs/Sphinx workflows. Example usage:

- `extractDocs('./src/support/AntennaFunctions/', './docs/readthedocs/source/code_antenna.md', 'Antenna Functions')`
- `extractDocs('./src/support/PAFunctions/', './docs/readthedocs/source/code_amp.md', 'Power Amplifier Functions')`
- `extractDocs('./src/support/SupportFunctions/', './docs/readthedocs/source/code_support.md', 'Supporting Functions', {'matlab2tikz'})`

i Input Parameters

- folderPath - Path to folder containing .m files (recursively searched)
- outFilename - Path to output .md file
- headerStr - (Optional) Header/title for the generated Markdown file
- excludedFolders - (Optional) Cell array of subfolders to exclude (by name)

i Output Parameters

- None

4.6.4 improveAxesAppearance.m

Path: src\support\SupportFunctions\improveAxesAppearance.m

Description:

This function improves the appearance of UIAxes in MATLAB App Designer. It supports enhancing single or dual Y-axis (yyaxis) plots.

i Input Parameters

- axesObj - Handle to the UIAxes object.
- 'YYAxis' - Logical (true/false), if the plot uses yyaxis.
- 'LineThickness' - Scalar > 0, sets line thickness for plotted lines.

i Output Parameters

- None

4.6.5 loadData.m

Path: src\support\SupportFunctions\loadData.m

Description:

This function loads data from a CSV or Excel file containing a single or sweep PA test measurement, or an Antenna test measurement.

i Input Parameters

- RFcomponent - Either 'PA', 'Antenna', or 'AntennaReference' depending on which type of measurement is being loaded.
- FileName - The name of the file that will be loaded into the application.

i Output Parameters

- combinedData - A struct containing all the data from each column of the loaded file.

4.6.6 saveData.m

Path: `src\support\SupportFunctions\saveData.m`

Description:

This function saves test data to either a CSV or Excel (.xlsx) file, depending on size and user selection. If the data exceeds Excel's row/column limits, only the CSV option is offered.

i Input Parameters

- `combinedData` - Either a table or a cell array of measurement vectors (e.g., {frequency, gain, ...}).
- `combinedNames` - Cell array of variable names corresponding to the data (e.g., {'Frequency (Hz)', 'Gain (dB)', ...}).

i Output Parameters

- `fullFilename` - Full path to the saved file, or an empty string if the user cancels the save dialog.
-

4.6.7 setupContextMenuFor3DPlot.m

Path: `src\support\SupportFunctions\setupContextMenuFor3DPlot.m`

Description:

This function sets up a right-click context menu for the 3D radiation pattern plot, allowing the user to export the visualization to image formats (PNG and JPG). This is particularly useful because plots rendered using the Antenna Toolbox may have limited export options or reduced visual quality by default.

i Input Parameters

- `app` - Application object containing the 3D plot handle and export logic.

i Output Parameters

- None
-

4.6.8 waitForInstrument.m

Path: src\support\SupportFunctions\waitForInstrument.m

Description:

The function waits for a connected instrument to complete its current operation by polling its status using the *OPC? SCPI query. This ensures that subsequent operations only proceed once the instrument is ready. The function enforces a timeout (default 15 seconds). The function:

- Starts a timer.
- Continuously queries instrument status via '*OPC?'.
- Waits between queries using appdefined delay.
- Exits if instrument reports ready (status == 1) or timeout is exceeded.

Input Parameters

- app - Application object that provides the delay setting between polls.
- Instrument - VISA-compatible instrument object supporting '*OPC?' queries.

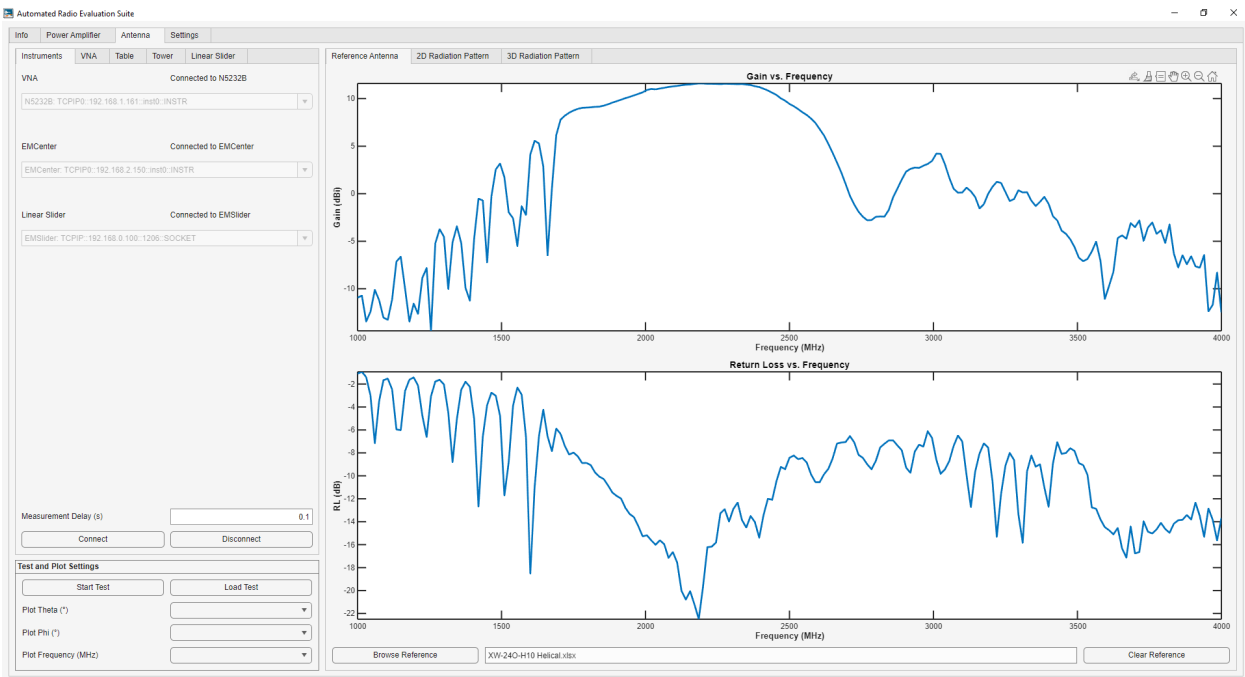
Output Parameters

- None

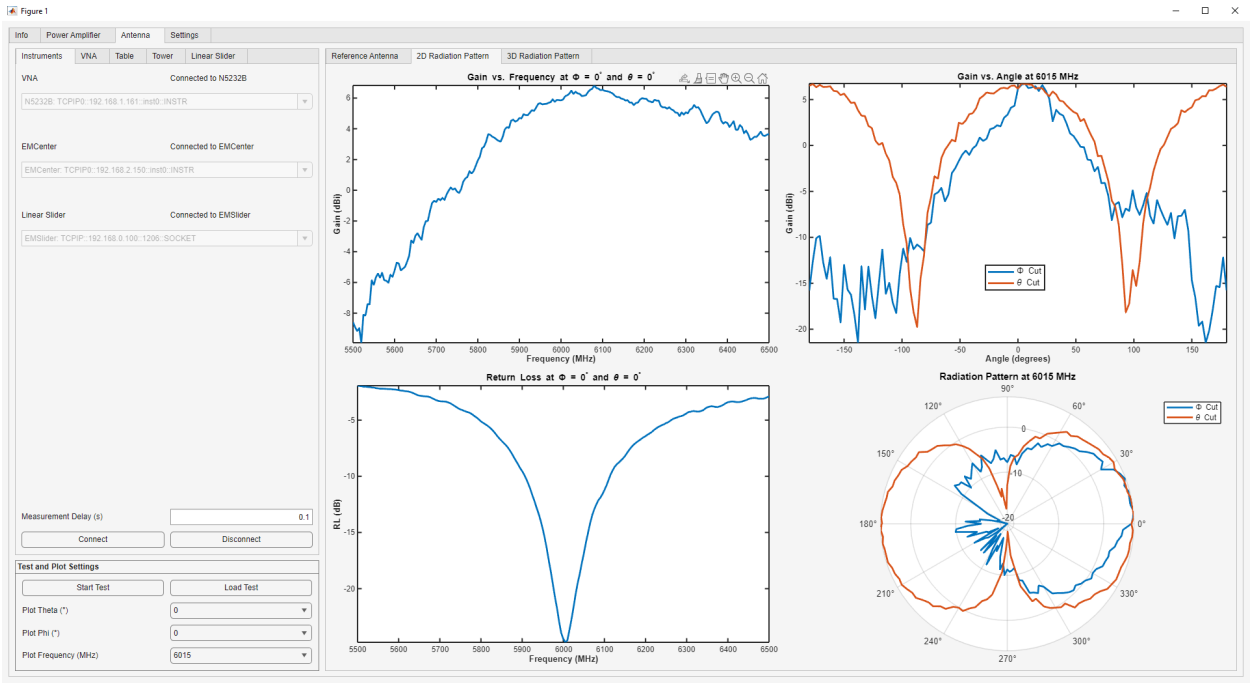
EXAMPLE GALLERY

5.1 Antenna Example Data

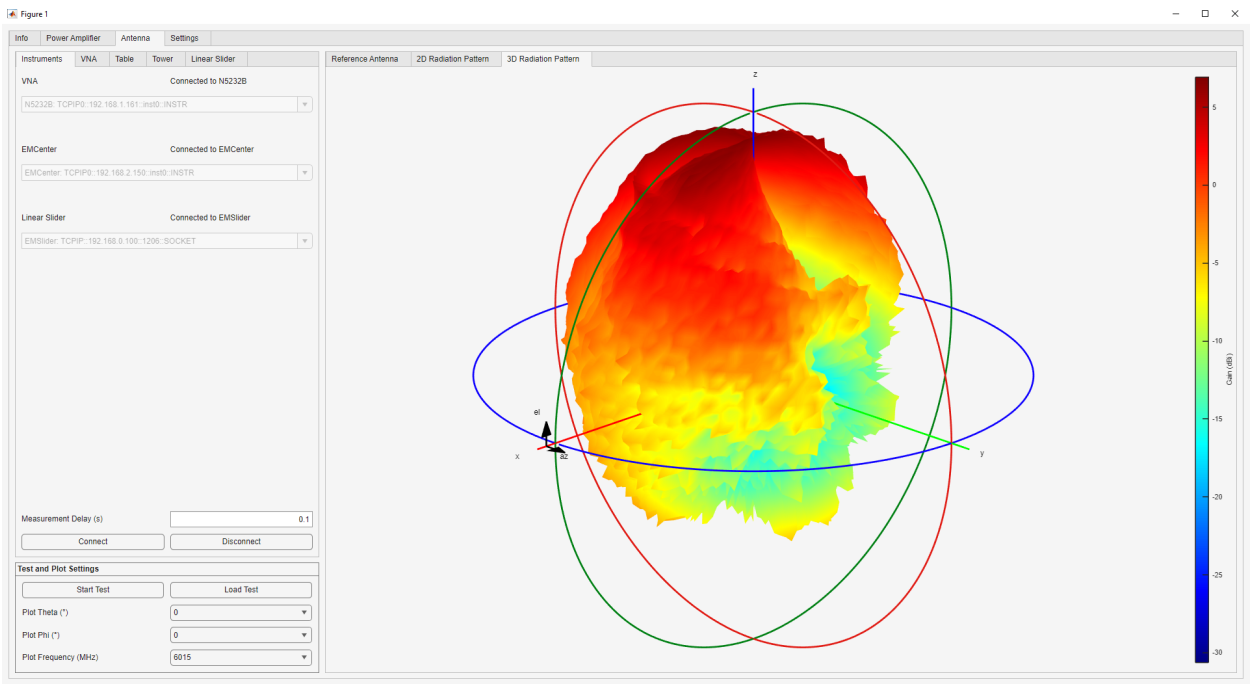
Reference Antenna



Measured Antenna 2D Radiation Pattern

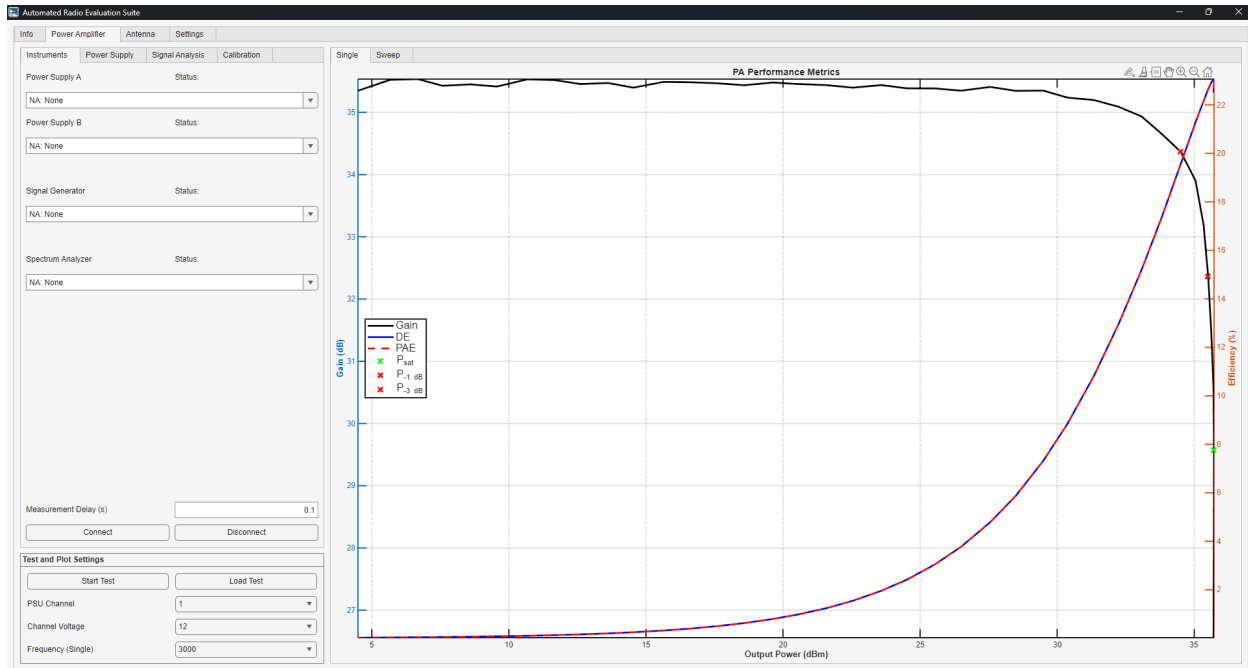


Measured Antenna 3D Radiation Pattern

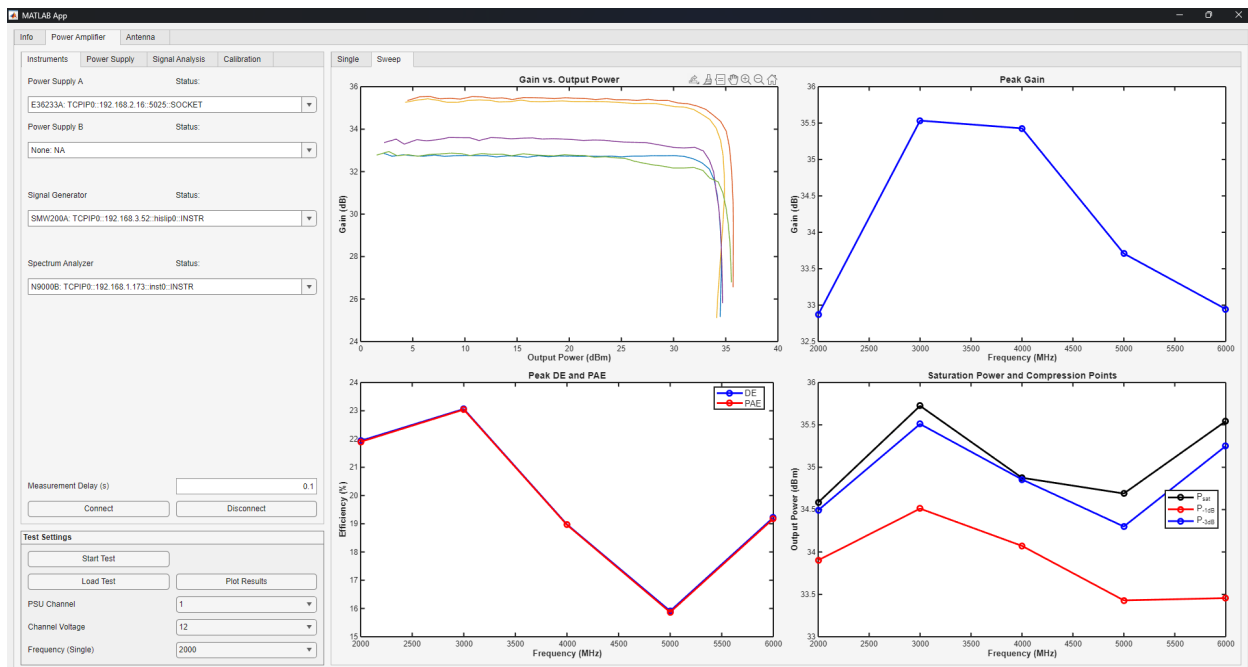


5.2 PA Example Data

Single Results View



Sweep Results View



6.1 Do I need to install anything else to use ARES?

Most features in ARES work out of the box.

For TikZ export:

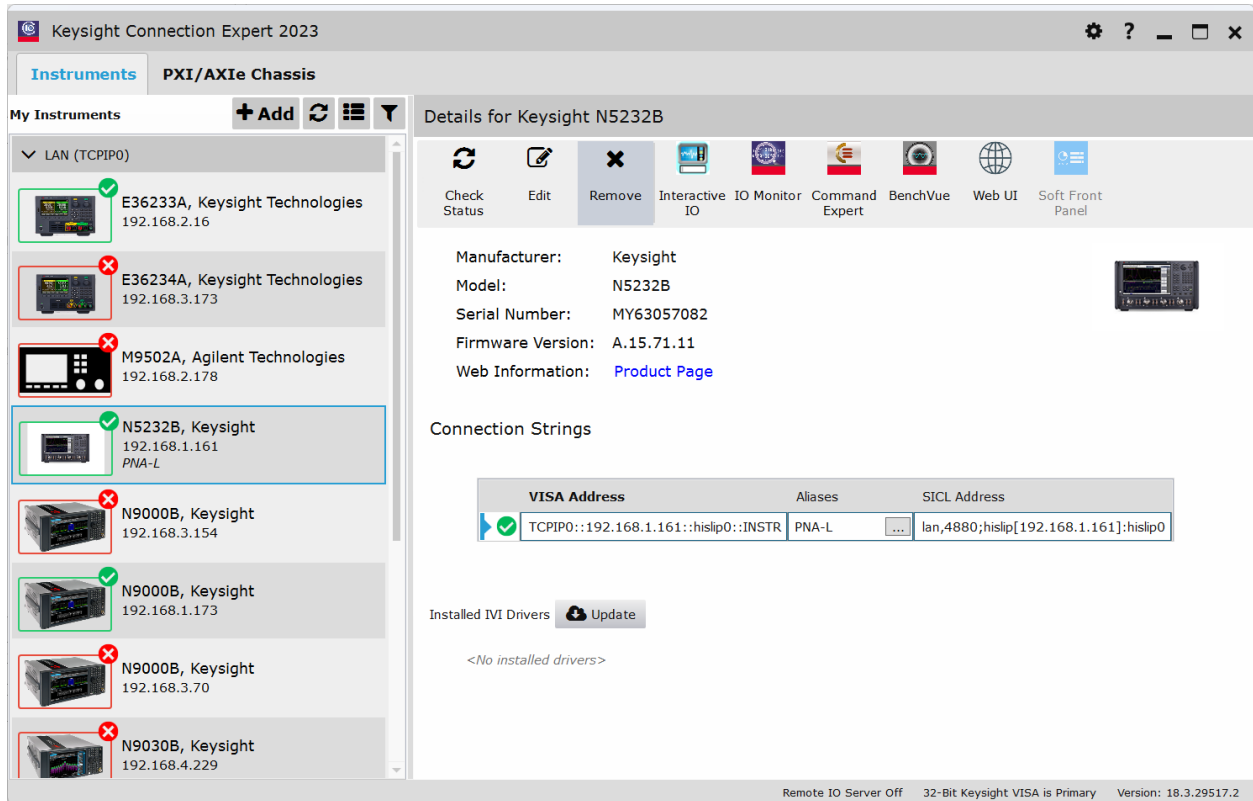
- The required **matlab2tikz** package is already bundled inside the ARES src folder.
- No external installation is required.

To ensure your environment is correctly configured, visit the [Getting Started](#) page for setup steps and required MATLAB toolboxes.

6.2 How do I find my instrument's VISA address?

You can obtain the instrument's VISA address in several ways:

- On the instrument front panel (check LAN or IO settings).
- Use **Keysight Connection Expert** or similar software to scan your local network for supported devices.



6.3 My instrument is not connecting. What should I check?

If ARES is unable to connect to your instrument:

- Make sure the instrument and computer are on the same network.
- Verify that the IP addresses and subnet masks are compatible.
- Test the connection by pinging the device using the command line:

```
ping XXX.XXX.XXX.XXX
```

- Ensure the VISA resource string is correctly formatted and entered in the instrument database.

6.4 Can I edit the instrument list manually?

Yes, the instrument database is a .csv file located at:

```
<userpath>/ARES/instrument_database.csv
```

You can open it in Excel, VS Code, or a text editor. Just make sure you preserve the format:

- Column headers: *Manufacturer, Model, Address*
- One row per instrument

6.5 Where are my saved test results and logs stored?

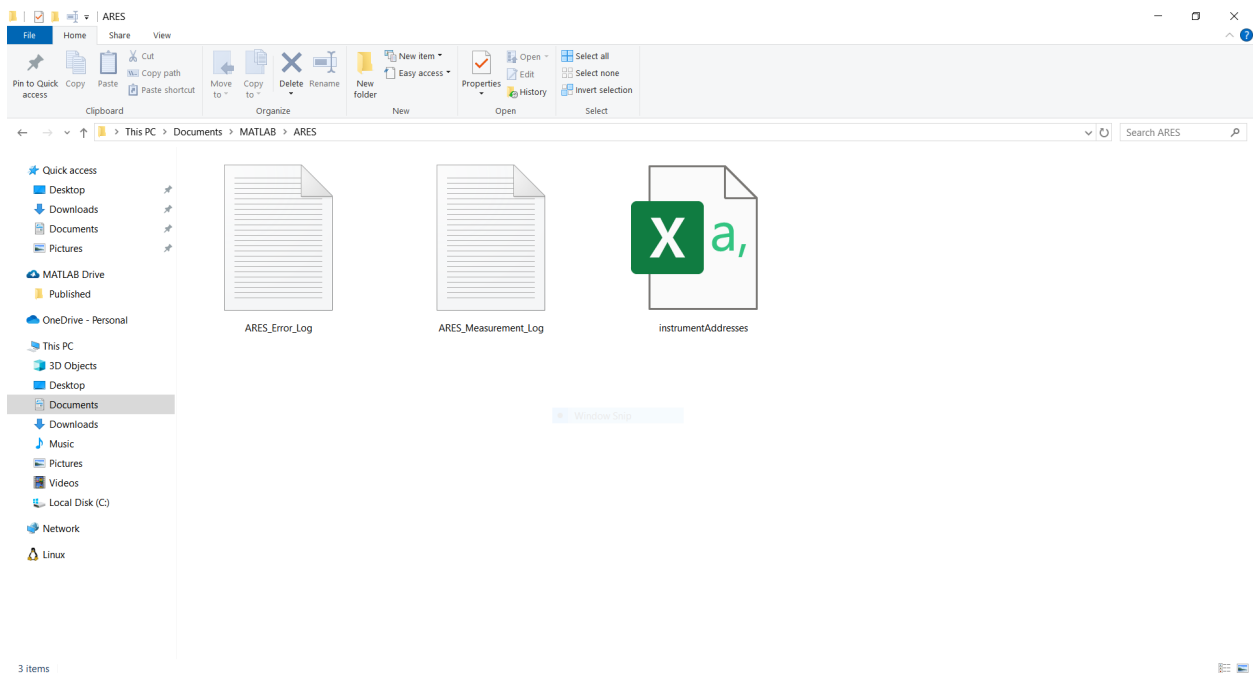
All logs and configuration files are stored in your MATLAB user path under the ARES directory:

```
<userpath>/ARES/
```

This includes:

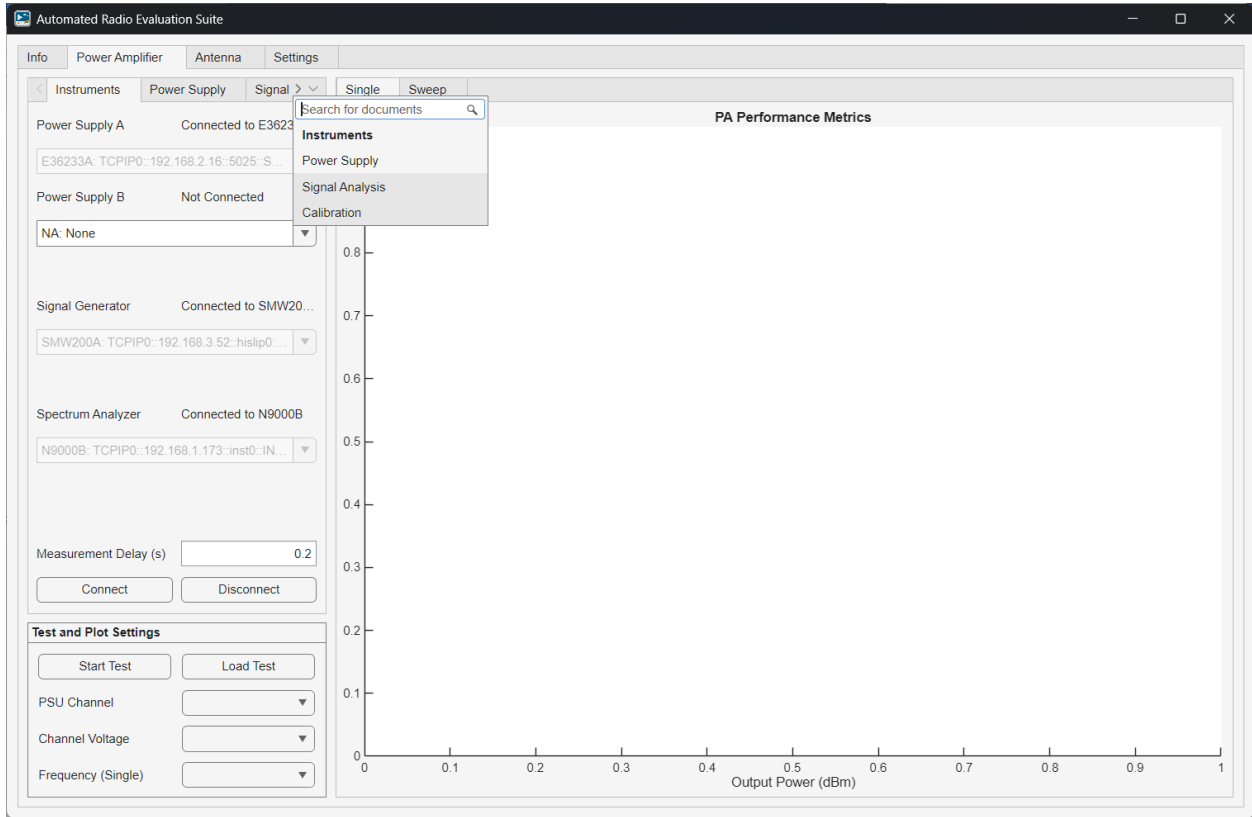
- **ARES_Error_Log.txt** – logs internal app errors
- **ARES_Measurement_Log.txt** – logs the duration of each test
- **instrument_database.csv** – stores instrument addresses

Measurement results are saved wherever you choose when prompted after a test run. ARES opens a folder selection dialog, and you select the location.



6.6 How do I show hidden tabs in the app?

If you don't see all available tabs, right-click and hold on the tab menu bar to expand hidden modules.



6.7 Why can't I export a plot as PDF or TikZ?

Not all plot types support every export format:

- TikZ only works for Cartesian 2D plots.
- Polar plots cannot be exported using TikZ.
- PDF export is not supported for 3D radiation plots.

6.8 VNA isn't returning expected values, what should I do?

Unexpected or incorrect measurements are usually caused by instrument misconfiguration, cabling issues, or calibration mismatches.

Here's a checklist to troubleshoot:

- Verify instrument calibration:
 - Use a calibration kit or eCal module to calibrate the VNA at the measurement plane (where the DUT and reference antenna connect).

- Calibration should be done before connecting the VNA to ARES, using the same frequency range and number of sweep points you plan to use.
- Don't change the VNA frequency sweep settings after calibration:
 - Once calibration is complete, do not adjust *Start/Stop Frequency* or *Sweep Points* in the app.
 - ARES reads the sweep settings from the VNA to preserve the calibrated state.
- Check cables and terminations.
- Inspect connectors for damage or wear.