

Release Notes for Phased Array System Toolbox™

How to Contact MathWorks



www.mathworks.com Web
comp.soft-sys.matlab Newsgroup
www.mathworks.com/contact_TS.html Technical Support



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



508-647-7000 (Phone)



508-647-7001 (Fax)



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

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

Release Notes for Phased Array System Toolbox™

© COPYRIGHT 2011–2012 by The MathWorks, Inc.

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

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

Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Patents

MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

R2012b

Acceleration of clutter model simulation with parfor or GPUs	2
FMCW waveforms	3
CFAR detector, supporting SOCA, GOCA, and order statistic thresholds	4
Function to simulate signals received by an array	5
Range-Doppler estimation	6
Propagation model that now supports intrapulse Doppler modeling	7
Visualization of array geometries	8
Random number stream usage for parallel computing	9
save and load for System objects	11

R2012a

Replicated Subarrays and Partitioned Array Apertures ..	14
Stretch Processing	16
U/V Space and Phi/Theta Angles	17
Multiple-Beam Beamformers	18
Support for Wideband Beam Pattern Analysis	19
Beamformer Option to Preserve Power	20
Symmetric Sweeping of Linear FM Waveform	21
Toolbox Location of dutycycle Function	22
New System Object Option on File Menu	23
Variable-Size Input Support for System Objects	24
Data Type Support for System Objects	25
New Property Attribute to Define States	26
New Methods to Validate Properties and Get States from System Objects	27
matlab.system.System changed to matlab.System	28

R2011b

Constant Gamma Clutter Modeling	30
Clutter Modeling Utilities	31
Phase-Coded Waveforms	32
Spectrum Weighting Options in Matched Filter	33
Expanded Lattice Options in Uniform Rectangular Array	35
Enhanced Plots Show Multiple Frequency Responses	36
Custom Antenna Removes Restriction on Radiation Pattern	37
Storing States When Saving or Cloning Objects	38
Custom System Objects	39
Conversion of Error and Warning Message Identifiers ...	40

R2011a

Introducing the Phased Array System Toolbox	42
Features	43

R2012b

Version: 1.3
New Features: Yes
Bug Fixes: No

Acceleration of clutter model simulation with parfor or GPUs

A change in the `phased.ConstantGammaClutter` System object™ facilitates performing Monte Carlo simulations with Parallel Computing Toolbox™ constructs, such as `parfor`. For details about this change, see the description of random number stream usage that follows.

The new `phased.gpu.ConstantGammaClutter` System object simulates clutter on a GPU. This System object typically runs faster than `phased.ConstantGammaClutter`. Using `phased.gpu.ConstantGammaClutter` requires a license for Parallel Computing Toolbox software.

FMCW waveforms

The `phased.FMCWWaveform` System object models a frequency modulated continuous wave (FMCW) waveform.

These functions help you determine appropriate property values for `phased.FMCWWaveform`:

- `time2range`
- `range2time`
- `range2bw`

These functions help you simulate and analyze systems that process FMCW waveforms:

- `dechirp`
- `beat2range`
- `range2beat`
- `rdcoupling`

CFAR detector, supporting SOCA, GOCA, and order statistic thresholds

Compatibility Considerations: Yes

The phased.CFARDetector System object provides a new property named Method. Using this property, you can choose among four CFAR detection algorithms: cell averaging (default), smallest-of cell averaging (SOCA), greatest-of cell averaging (GOCA), and order statistic.

Compatibility Considerations

In R2012b, if you save a CFAR detector variable in a MAT-file, you cannot load that variable from the MAT-file in an earlier version. Instead, re-create the variable in the earlier version.

Function to simulate signals received by an array

The new `sensorsig` function simulates plane wave signals received at a phased array. This function facilitates statistical analysis and testing of direction-of-arrival algorithms. The function does not require you to simulate an entire phased array system.

Range-Doppler estimation

The new `phased.RangeDopplerResponse` System object generates and plots range-Doppler maps.

Propagation model that now supports intrapulse Doppler modeling

Compatibility Considerations: Yes

The `phased.FreeSpace System` object models the Doppler shift both within a pulse (fast time) and between successive pulses (slow time). In previous releases, the object modeled only the slow-time Doppler shift.

This enhanced Doppler shift modeling is especially useful for observing range-Doppler coupling in a radar system that uses a linear FM waveform.

Compatibility Considerations

The `step` method of the `phased.FreeSpace System` object has two additional input parameters:

- `origin_vel`, the velocity of the signal origin
- `dest_vel`, the velocity of the signal destination

To update legacy code that uses this `step` method, use one of these approaches:

- In some cases, the signal origin or signal destination is stationary. If so, set the corresponding velocity input argument to `[0; 0; 0]`.
- In other cases, the signal origin or signal destination is moving, and you are using `phased.Platform` to model the moving platform. In this situation, obtain the velocity vector as an additional output argument from the `step` method of `phased.Platform`. Then, specify this velocity vector as an input argument in the `step` method of `phased.FreeSpace`. For example, compare the R2012a and R2012b versions of an example, and notice the introduction of the `txvel` variable in the R2012b version.

Visualization of array geometries

You can call `viewArray` on a phased array to plot the positions, normal directions, and element indices of the elements in the array. For arrays containing subarrays, `viewArray` can also graphically highlight one or more of the subarrays.

The System objects affected are:

- `phased.ULA`
- `phased.URA`
- `phased.ConformalArray`
- `phased.ReplicatedSubarray`
- `phased.PartitionedArray`

For more information, see the reference pages for:

- `viewArray` for arrays without subarrays, such as `phased.ULA`
- `viewArray` for arrays containing subarrays, such as `phased.PartitionedArray`

Random number stream usage for parallel computing

Compatibility Considerations: Yes

System objects in Phased Array System Toolbox™ software that rely on a random number generator now behave differently when you set the `SeedSource` property to 'Auto':

- The object uses the global stream of random numbers instead of a private stream. This change is useful if you are performing the computations in a set of Monte Carlo trials involving Parallel Computing Toolbox software. In that situation, the global stream is more suitable than a stream that the System object manages internally.
- The `reset` method does not reset the random number generator.

The System objects affected by this change are:

- `phased.BarrageJammer`
- `phased.ConstantGammaClutter`
- `phased.ReceiverPreamp`
- `phased.ReceiverPreamp`
- `phased.Transmitter`

Compatibility Considerations

The following compatibility considerations apply to System objects that existed in earlier releases, if your legacy code configures the objects with the `SeedSource` property set to 'Auto'.

- In operations involving the System objects, the specific random numbers in R2012b differ from the random numbers in earlier releases.
- If your code later performs an arbitrary operation that uses the global random number stream, this operation also uses different random numbers compared to earlier releases.
- In some cases, your code may rely on operations in earlier releases that reset or restore the random number generator while loading, cloning, or

resetting objects. If so, you should update your code to reset or restore the global stream yourself.

save and load for System objects

You can create your own save and load methods for a System object you create. To do so, use the `saveObjectImpl` and `loadObjectImpl`, respectively, in your class definition file.

R2012a

Version: 1.2
New Features: Yes
Bug Fixes: No

Replicated Subarrays and Partitioned Array Apertures

Compatibility Considerations: Yes

A *subarray* is an accessible subset of array elements. The following new System objects enable you to create arrays that contain subarrays:

- `phased.ReplicatedSubarray`
- `phased.PartitionedArray`

The following table lists existing System objects that now support operations on arrays that contain subarrays. Each of the objects in the table has a property called `SensorArray` or `Sensor`. You can set that property to an array object that contains subarrays. Also, some of the objects in the table support subarray steering through a new input argument, `STEERANGLE`, in the `step` method.

System Object	SensorArray or Sensor Can Contain Subarrays	step Syntax Can Include Subarray Steering
<code>phased.AngleDopplerResponse</code>	Yes	Not applicable
<code>phased.ArrayGain</code>	Yes	Yes
<code>phased.ArrayResponse</code>	Yes	Yes
<code>phased.Collector</code>	Yes	Yes
<code>phased.ConstantGammaClutter</code>	Yes	Yes
<code>phased.MVDRBeamformer</code>	Yes	Not applicable
<code>phased.PhaseShiftBeamformer</code>	Yes	Not applicable
<code>phased.Radiator</code>	Yes	Yes
<code>phased.STAPSMIBeamformer</code>	Yes	Not applicable
<code>phased.SteeringVector</code>	Yes	Yes
<code>phased.SubbandPhaseShiftBeamformer</code>	Yes	Not applicable
<code>phased.WidebandCollector</code>	Yes	Yes

For more information about using subarrays, see [Subarrays Within Arrays](#) or [Subarrays in Phased Array Antennas](#).

Compatibility Considerations

The `IncludeElementResponse` property of the `phased.SteeringVectorSystem` object is no longer tunable in V1.2 (R2012a). This change facilitates support for arrays containing subarrays.

You may have code from an earlier version that tunes the value of the `IncludeElementResponse` property of a locked steering vector object. If so, the code will produce an error message in R2012a. You can avoid the error message by calling `release` to unlock the object, or by not changing the value of the `IncludeElementResponse` property.

Stretch Processing

The following new features help you perform pulse compression on linear frequency modulation (FM) waveforms using stretch processing:

- `phased.StretchProcessor` System object
- `stretchfreq2rng` function
- `getStretchProcessor` method of `phased.LinearFMWaveform` System object

Stretch processing is sometimes called *deramping* or *dechirping*.

For more information about using stretch processing, see [Stretch Processing or Range Estimation Using Stretch Processing](#).

U/V Space and Phi/Theta Angles

Several enhancements facilitate performing operations in the u/v coordinate system or in a spherical coordinate system that describes angles using φ and θ instead of azimuth and elevation:

- Visualize radiation patterns in u/v space using the `plotResponse` method for arrays, antenna elements, and microphone elements. To use this feature, include 'Format', 'UV' in the `plotResponse` syntax.
- Convert coordinates from one coordinate system to another using these new functions:
 - `uv2azel`
 - `azel2uv`
 - `phitheta2azel`
 - `azel2phitheta`
 - `uv2phitheta`
 - `phitheta2uv`
- Convert antenna radiation patterns from one coordinate system to another using these new functions:
 - `uv2azelpat`
 - `azel2uvpat`
 - `phitheta2azelpat`
 - `azel2phithetapat`
 - `uv2phithetapat`
 - `phitheta2uvpat`

For examples, see [Antenna Radiation Pattern in U/V Coordinates](#) and [Antenna Array Analysis with Custom Radiation Pattern](#). For background information about the coordinate systems, see [Spherical Coordinates](#).

Multiple-Beam Beamformers

Compatibility Considerations: Yes

The following beamformers support multiple beamforming directions:

- `phased.PhaseShiftBeamformer`
- `phased.SubbandPhaseShiftBeamformer`
- `phased.MVDRBeamformer`

You can use this capability to model switched-beam systems.

To indicate multiple beamforming directions, use a matrix instead of a vector for the `Direction` property of the beamformer object or the `ANG` input argument of `step`. In earlier versions, the value required a vector. Now, when you specify multiple beamforming directions, the `Y` and `W` outputs of `step` have an extra matrix dimension.

Compatibility Considerations

In V1.2 (R2012a), you can create a MAT-file that stores a beamformer variable specifying multiple directions in the `Direction` property. However, you cannot load that variable from the MAT-file in an earlier version. As an alternative, you can re-create the variable in the earlier version and specify only one beamforming direction.

Support for Wideband Beam Pattern Analysis

The `plotResponse` method for arrays, antenna elements, and microphone elements has enhancements for use with wideband beamforming applications.

- For arrays or elements, `plotResponse` can plot multiple frequency responses in a three-dimensional waterfall plot. To use this feature, include `'OverlayFreq', false` in the `plotResponse` syntax. The `OverlayFreq` argument is new.
- For arrays, `plotResponse` can apply weights independently to each frequency in the plot. For example, you can use beamformer weights as in [Visualization of Wideband Beamformer Performance](#). To use this feature, include `'Weights', Value` in the `plotResponse` syntax, where `Value` is a vector or matrix. R2011b required that the weights be the same for all frequencies in the plot and that `Value` be a vector.

In the `phased.ArrayResponse` and `phased.ArrayGain` System objects, the `step` method permits the `WEIGHTS` input argument to be a vector or a matrix. In earlier releases, `WEIGHTS` is a vector.

Beamformer Option to Preserve Power

Compatibility Considerations: Yes

The phase shift beamformer offers options for normalizing the beamformer weights. To select an option, set the new `WeightsNormalization` property of the `phased.PhaseShiftBeamformer` object to one of these values:

- 'Distortionless' — The gain toward the beamforming direction is 0 dB. This choice is the default and matches the behavior in earlier versions.
- 'Preserve power' — The norm of the weights is 1.

Compatibility Considerations

In V1.2 (R2012a), if you save a phase shift beamformer variable in a MAT-file, you cannot load that variable from the MAT-file in an earlier version. Instead, re-create the variable in the earlier version.

Symmetric Sweeping of Linear FM Waveform

You can create a linear FM waveform to sweep in an interval that is symmetric about 0 or positive only. To choose the location of the FM sweep interval, set the new `SweepInterval` property of the `phased.LinearFMWaveform` object to one of these values:

- 'Positive' — The waveform sweeps between 0 and B , where B is the sweep bandwidth. This choice is the default and matches the behavior in earlier versions.
- 'Symmetric' — The waveform sweeps between $-B/2$ and $B/2$.

Toolbox Location of dutyCycle Function

The `dutyCycle` function in the Signal Processing Toolbox™ product replaces the earlier `dutyCycle` function in the Phased Array System Toolbox product. The new function includes both the capabilities of the earlier function and additional new capabilities.

New System Object Option on File Menu

The File menu on the MATLAB® desktop now includes a **New > System object** menu item. This option opens a System object class template, which you can use to define a System object class.

Variable-Size Input Support for System Objects

System objects that you define now support inputs that change size at runtime.

Data Type Support for System Objects

System objects that you define now support all MATLAB data types as inputs and outputs.

New Property Attribute to Define States

R2012a adds the new `DiscreteState` attribute for properties in your System object class definition file. Discrete states are values calculated during one step of an object's algorithm that are needed during future steps.

New Methods to Validate Properties and Get States from System Objects

The following methods have been added:

- `validateProperties` – Checks that the System object is in a valid configuration. This applies only to objects that have a defined `validatePropertiesImpl` method
- `getDiscreteState` – Returns a struct containing a System object's properties that have the `DiscreteState` attribute

matlab.system.System changed to matlab.System

Compatibility Considerations: Yes

The base System object class name has changed from `matlab.system.System` to `matlab.System`.

Compatibility Considerations

The previous `matlab.system.System` class will remain valid for existing System objects. When you define new System objects, your class file should inherit from the `matlab.System` class.

R2011b

Version: 1.1
New Features: Yes
Bug Fixes: No

Constant Gamma Clutter Modeling

The new `phased.ConstantGammaClutter` System object helps you model surface clutter using the constant gamma model. You can use this object when simulating a radar system or estimating its performance statistically.

For more information, see these resources:

- Clutter Modeling
- `phased.ConstantGammaClutter`
- Ground Clutter Mitigation with Moving Target Indication (MTI) Radar

Clutter Modeling Utilities

These new utility functions can help you implement custom clutter models:

- `billingsleyicm`
- `depressionang`
- `effearthradius`
- `grazingang`
- `horizonrange`
- `surfclutterrcs`
- `surfacegamma`

Phase-Coded Waveforms

The new `phased.PhaseCodedWaveform` System object generates samples of a phase-coded pulse waveform. This object supports these code types:

- Barker
- Frank
- P1
- P2
- P3
- P4
- Px
- Zadoff-Chu

For more information, see [Phase-Coded Waveforms](#) and `phased.PhaseCodedWaveform`.

Spectrum Weighting Options in Matched Filter

Compatibility Considerations: Yes

The `phased.MatchedFilter` System object supports spectrum weighting using these window types:

- Hamming
- Chebyshev
- Hann
- Kaiser
- Taylor

You can also specify a custom window. To do so, write a function that takes the window length as an input argument and returns window coefficients in an output argument.

For more information, see [Matched Filtering](#) or `phased.MatchedFilter`.

Compatibility Considerations

If you save a `phased.MatchedFilter` object in a MAT-file in V1.1 (R2011b) and then load the MAT-file in V1.0 (R2011a), the object does not perform spectrum weighting. The Command Window shows this warning:

```
Warning: While loading an object of class 'phased.MatchedFilter':  
No public field SpectrumWindow exists for class phased.MatchedFilter.
```

If you write code in V1.1 (R2011b) that sets or reads any of the following properties of `phased.MatchedFilter` object, the code produces an error message in V1.0 (R2011a).

- `SpectrumWindow`
- `CustomSpectrumWindow`
- `SpectrumRange`
- `SampleRate`
- `SidelobeAttenuation`

- Beta
- Nbar

Expanded Lattice Options in Uniform Rectangular Array

Compatibility Considerations: Yes

The `phased.URA` System object supports both triangular lattices and rectangular lattices. You use the `Lattice` property to select the lattice type.

In V1.0 (R2011a), `phased.URA` supports only rectangular lattices and does not have a `Lattice` property.

Compatibility Considerations

If you save a `phased.URA` object in a MAT-file in V1.1 (R2011b) and then load the MAT-file in V1.0 (R2011a), the object uses a rectangular lattice. The Command Window shows this warning:

```
Warning: While loading an object of class 'phased.URA':  
No public field Lattice exists for class phased.URA.
```

If you write code in V1.1 (R2011b) that sets or reads the `Lattice` property of a `phased.URA` object, the code produces an error message in V1.0 (R2011a).

Enhanced Plots Show Multiple Frequency Responses

The `plotResponse` method can plot multiple frequency responses along an azimuth cut or elevation cut. This method is available for the System objects for array design, antenna elements, and microphone elements. To create a plot of multiple frequency responses, use a `plotResponse` syntax in which:

- `FREQ` is a row vector.
- `RespCut` either does not appear explicitly, or has the value `'Az'` or `'El'`.

The affected System objects are:

- `phased.ConformalArray`
- `phased.CosineAntennaElement`
- `phased.CustomAntennaElement`
- `phased.CustomMicrophoneElement`
- `phased.IsotropicAntennaElement`
- `phased.OmnidirectionalMicrophoneElement`
- `phased.ULA`
- `phased.URA`

In V1.0 (R2011a), `FREQ` must be a scalar. The resulting plot shows one frequency response.

Custom Antenna Removes Restriction on Radiation Pattern

The phased.CustomAntennaElement System object now permits more general radiation patterns. The main beam of the pattern is no longer required to point to 0 degrees azimuth and 0 degrees elevation.

Storing States When Saving or Cloning Objects

Compatibility Considerations: Yes

The `save` and `clone` operations now store all states of the System objects in the phased package. As a result, calling the `step` method on a loaded or cloned object resumes processing from the state where the original object left off. In V1.0 (R2011a), the loaded or cloned object is unlocked and uninitialized.

Compatibility Considerations

If your legacy code exploits the unlocked, uninitialized state of a loaded or cloned object, you should update the code in V1.1 (R2011b). You can use the `release` method to unlock objects.

Custom System Objects

You can now create custom System objects in MATLAB. This capability allows you to define your own System objects for time-based and data-driven algorithms, I/O, and visualizations. The System object API provides a set of implementation and service methods that you incorporate into your code to implement your algorithm. See [Define New System Objects in the DSP System Toolbox™](#) documentation for more information.

Conversion of Error and Warning Message Identifiers

Compatibility Considerations: Yes

For version 1.1 (R2011b), some error and warning message identifiers have changed in Phased Array System Toolbox software.

Compatibility Considerations

If you have scripts or functions that use message identifiers that changed, you must update the code to use the new identifiers. Typically, message identifiers are used to turn off specific warning messages, or in code that uses a try/catch statement and performs an action based on a specific error identifier.

For example, the 'phased:phased:RootWSFEstimator:ZeroSourceNumber' identifier and the 'phased:phased:RootMUSICEstimator:ZeroSourceNumber' identifier have both changed to 'phased:phased:doa:ZeroSourceNumber'. If your code checks for one of the earlier values, you must update it to check for 'phased:phased:doa:ZeroSourceNumber' instead.

To determine the identifier for a warning, run the following command just after you see the warning:

```
[MSG,MSGID] = lastwarn;
```

This command saves the message identifier to the variable MSGID.

To determine the identifier for an error, run the following command just after you see the error:

```
exception = MException.last;  
MSGID = exception.identifier;
```

Note Warning messages indicate a potential issue with your code. While you can turn off a warning, a suggested alternative is to change your code so it runs warning-free.

R2011a

Version: 1.0
New Features: Yes
Bug Fixes: No

Introducing the Phased Array System Toolbox

Phased Array System Toolbox provides algorithms and tools for the design, simulation, and analysis of phased array signal processing systems. These capabilities are provided as MATLAB functions and MATLAB System objects. The system toolbox includes algorithms for waveform generation, beamforming, direction of arrival estimation, target detection, and space-time adaptive processing. The system toolbox lets you build monostatic, bistatic, and multistatic architectures for a variety of array geometries. You can model these architectures on stationary or moving platforms. Array analysis and visualization tools help you evaluate spatial, spectral, and temporal performance. The system toolbox lets you model an end-to-end phased array system or use individual algorithms to process acquired data.

Features

Key features of Phased Array System Toolbox Version 1.0 include:

- Algorithms available as MATLAB functions and MATLAB System objects
- Monostatic, bistatic, and multistatic phased array system modeling
- Array analysis and 3D visualization; physical array modeling for uniform linear arrays, uniform rectangular arrays, and arbitrary conformal arrays on platforms with motion
- Broadband and narrowband digital beamforming functions, including MVDR/Capon, LCMV, time delay, Frost, time delay LCMV, and subband phase shift
- Space-time adaptive processing algorithms, including displaced phase center array (DPCA), adaptive DPCA, sample matrix inversion (SMI) beamforming, and angle-Doppler response visualization
- Direction of arrival algorithms, including MVDR, ESPRIT, Beamscan, Root MUSIC, and monopulse tracking
- Waveform synthesis functions for pulsed CW, linear FM, stepped FM, and staggered PRF signals, and waveform visualization tools for ambiguity function and matched filter response
- Algorithms for TVG, pulse compression, coherent and non-coherent integration, CFAR processing, plotting ROC curves, and estimating range and Doppler