

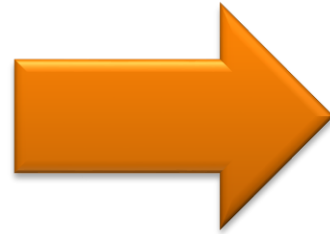
# What's New in MATLAB and Simulink for Signal Processing

**Jonas Rutström**  
**Senior Application Engineer**

**So, what's new?**

# NORDIC MATLAB EXPO 2014

*R2014b*



*R2016a*

*“What’s New in MATLAB and Simulink for Signal Processing”*

*Signal Processing*

*Audio*

*Antenna to Bits*

*WLAN/LTE*

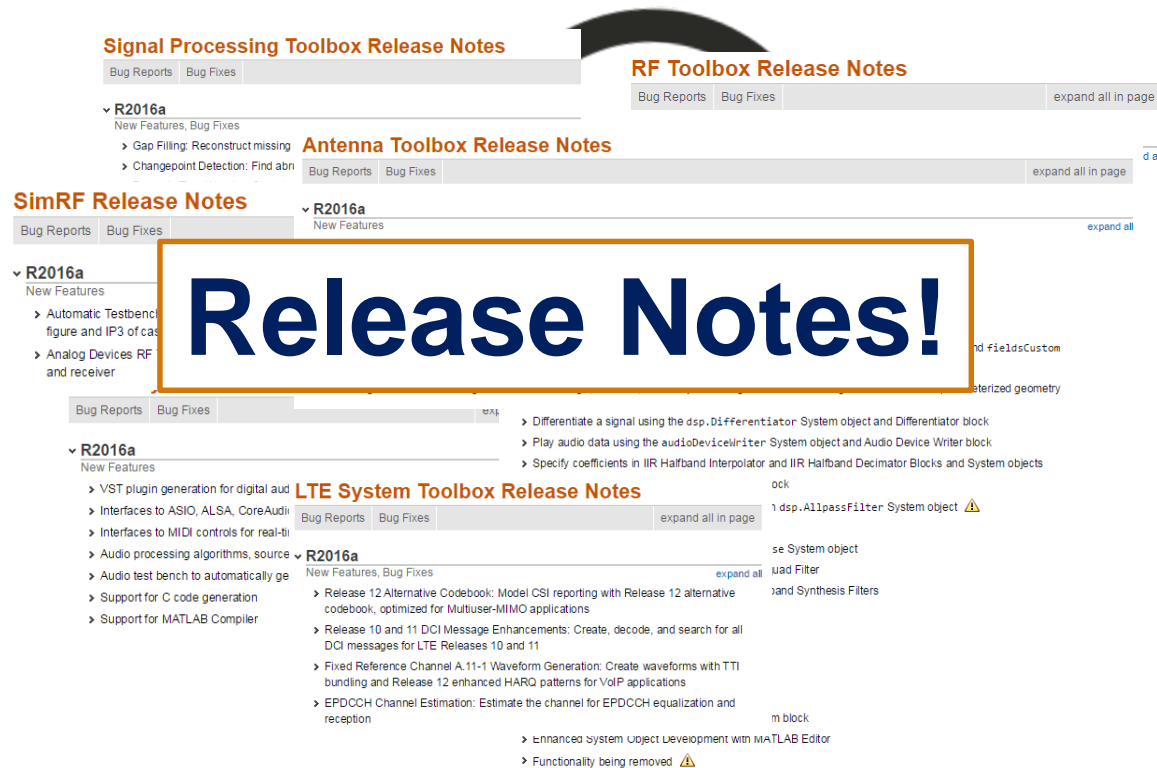
*Image and Video Processing*

# A few words about “What’s New?”



Details

# A few words about “What’s New?”



The screenshot shows the MATLAB release notes for R2016a, organized into sections for different toolboxes:

- Signal Processing Toolbox Release Notes**: Includes sections for Bug Reports and Bug Fixes.
- RF Toolbox Release Notes**: Includes sections for Bug Reports and Bug Fixes.
- Antenna Toolbox Release Notes**: Includes sections for Bug Reports and Bug Fixes.
- SimRF Release Notes**: Includes sections for Bug Reports and Bug Fixes. A large orange-bordered box with the text "Release Notes!" is overlaid on this section.
- LTE System Toolbox Release Notes**: Includes sections for Bug Reports and Bug Fixes.

Each section lists new features and bug fixes for the R2016a release. For example, the LTE System Toolbox notes include: "Release 12 Alternative Codebook: Model CSI reporting with Release 12 alternative codebook, optimized for Multiuser-MIMO applications", "Release 10 and 11 DCI Message Enhancements: Create, decode, and search for all DCI messages for LTE Releases 10 and 11", and "Fixed Reference Channel A.11-1 Waveform Generation: Create waveforms with TTI bundling and Release 12 enhanced HARQ patterns for VoIP applications".

*Signal Processing*

*Audio*

*Antenna to Bits*

*WLAN/LTE*

*Image and Video Processing*





*Signal Processing*

*Audio*

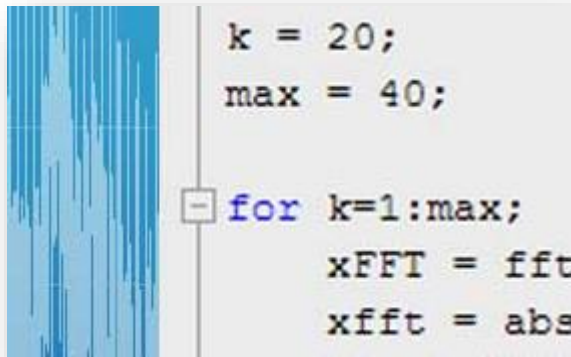
*Antenna to Bits*

*WLAN/LTE*

*Image and Video Processing*

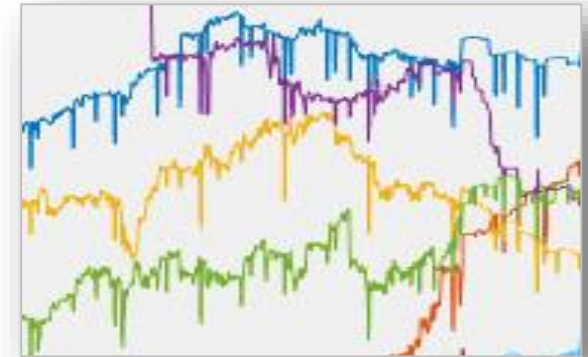
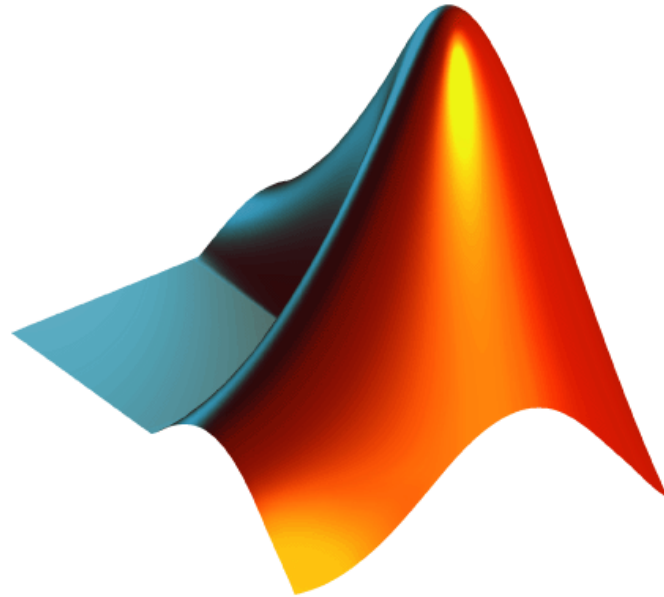
# Signal Processing Engineers...

# Signal Processing Engineers...



```
k = 20;  
max = 40;  
  
for k=1:max;  
    xFFT = fft  
    xfft = abs
```

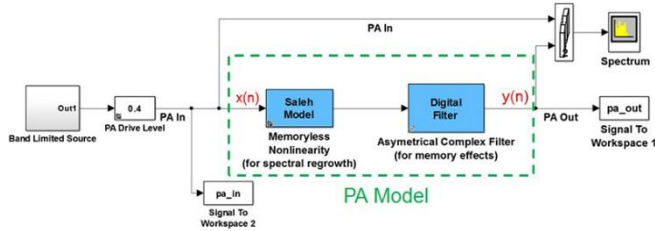
**Develop algorithms**



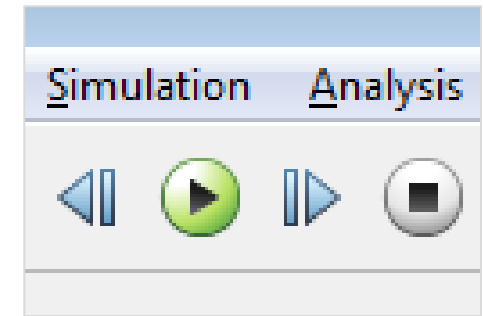
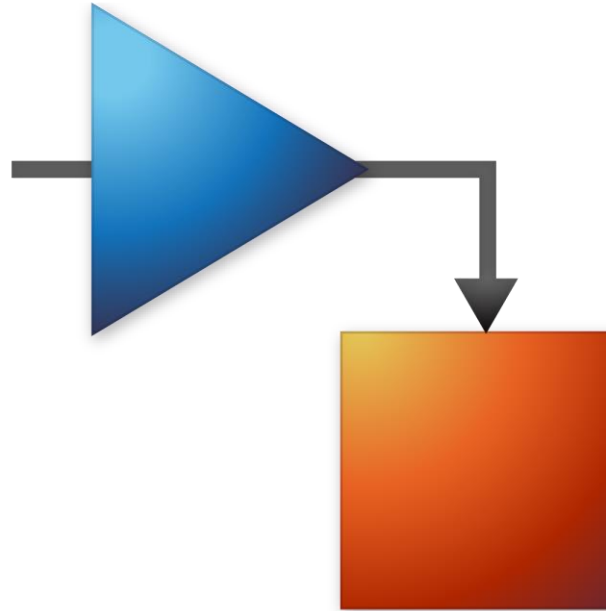
**Analyze data**

***write MATLAB code.***

# Signal Processing Engineers...



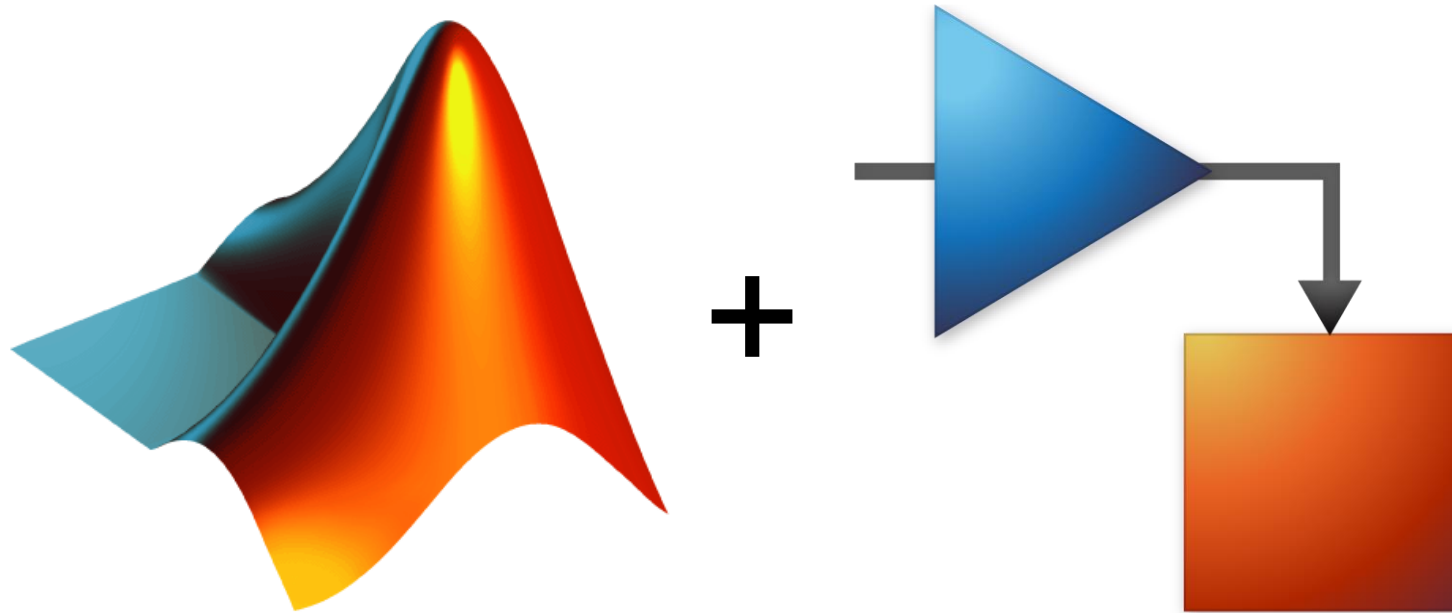
**Model systems**



**Run simulations**

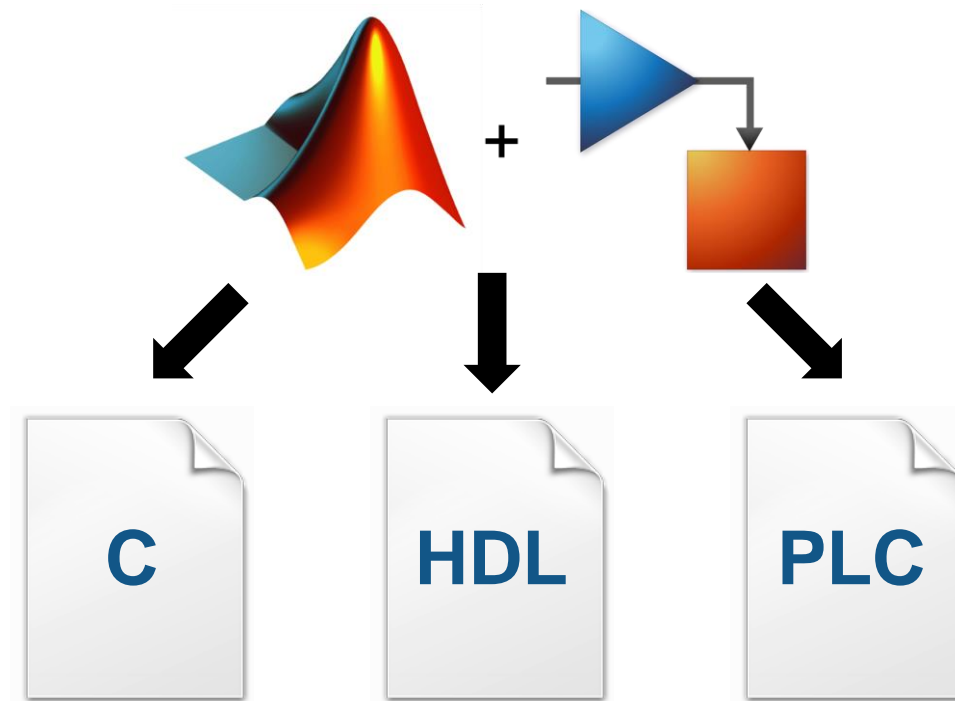
*build Simulink models.*

# Signal Processing Engineers...



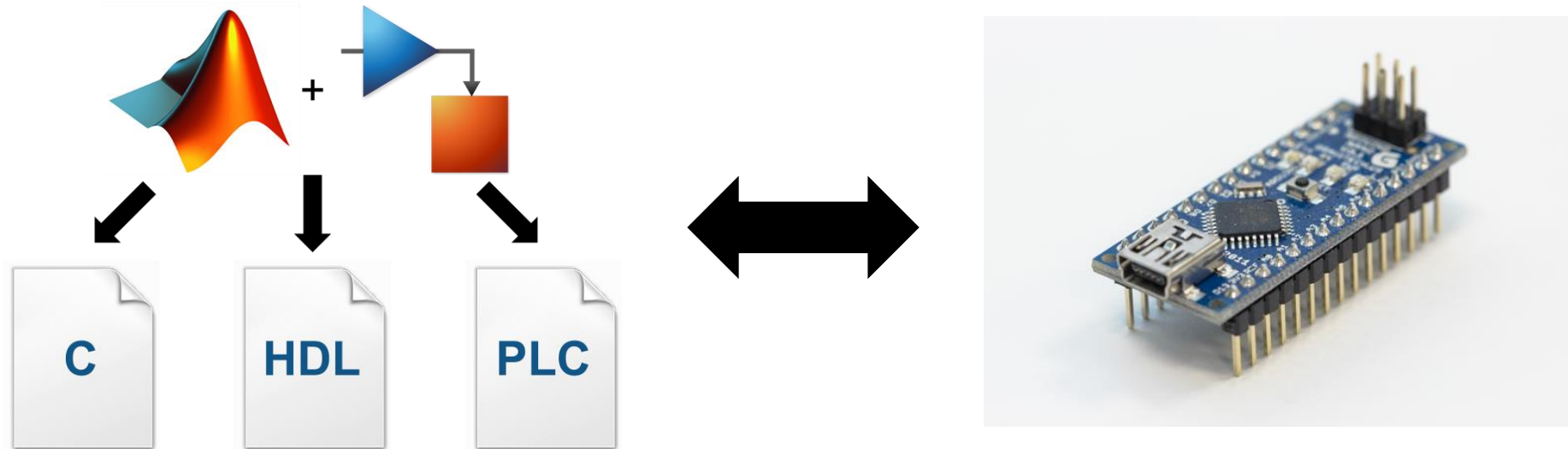
***combine MATLAB code and  
Simulink models together.***

# Signal Processing Engineers...



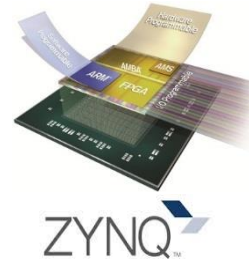
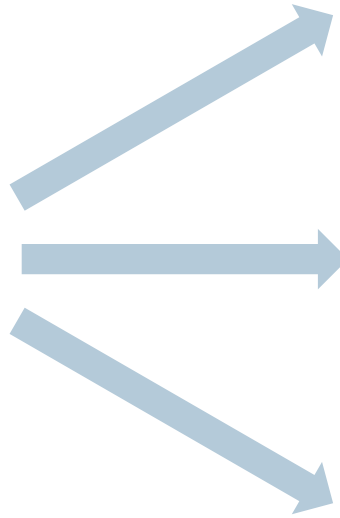
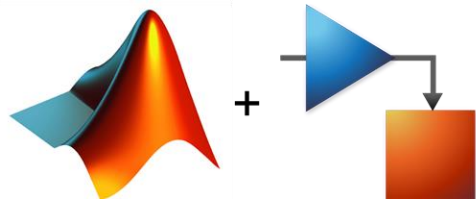
*generate code.*

# Signal Processing Engineers...



***connect software to hardware.***

# General trend... | *Idea to implementation*





# Increased support for code generation and fixed point design

## Functions and Objects Supported for C and C++ Code Generation — Category List

You can generate efficient C and C++ code for a subset of MATLAB® built-in functions and toolbox functions, classes, and System objects that you call from MATLAB code. These functions, classes, and System objects are listed by MATLAB category or toolbox category in the following tables.

**Signal Processing in MATLAB**

Function
chol
conv
fft
fft2
fftn
fftshift
filter
freqspace
ifft
ifft2
fftn
ifftshift
svd
zp2tf

**Signal Processing Toolbox**  
C and C++ code generation for the following functions  
Specifying Inputs in Code Generation from MATLAB

Note: Many Signal Processing Toolbox functions require the Signal Processing Toolbox.

Function
barthannwin
bartlett
besselap
bitrevorder
blackman
blackmanharris
bohmanwin
buttap
butter
buttord
cfirpm
cheblap
cheb2ap
cheblord
cheb2ord
chebwin
cheby1
cheby2
db2pow
dct

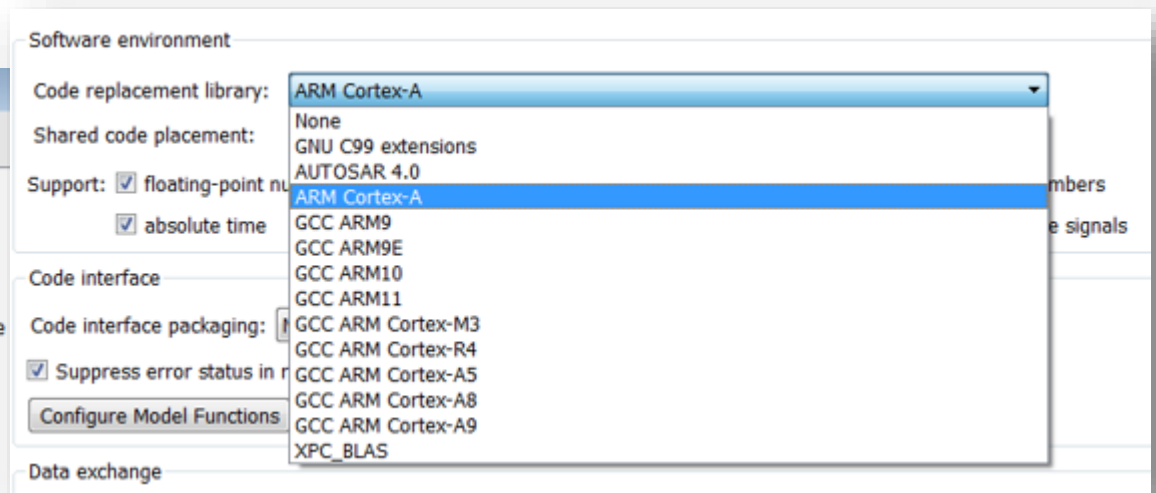
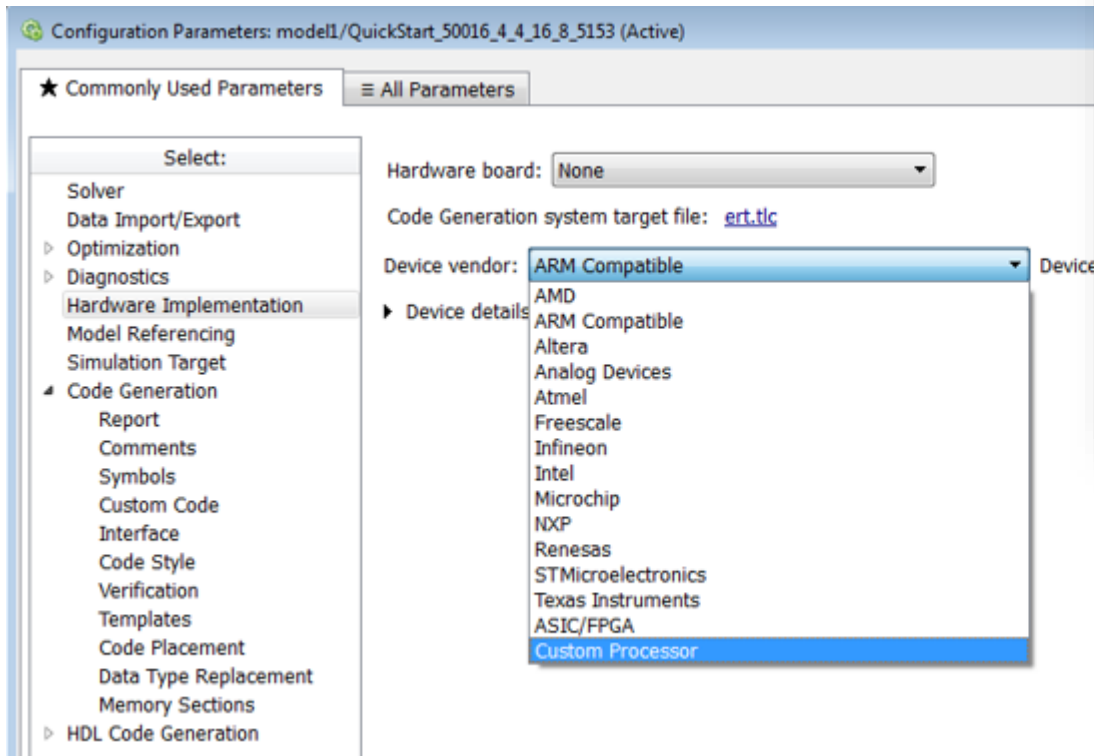
**DSP System Toolbox**  
C code generation for the following functions

Name
Estimation
dsp.BurgAREstimator
dsp.BurgSpectrumEstimator
dsp.CepstralToLPC
dsp.CrossSpectrumEstimator
dsp.LevinsonSolver
dsp.LPCToAutocorrelation
dsp.LPCToCepstral
dsp.LPCToLSF
dsp.LPCToLSP
dsp.LPCToRRC
dsp.LSFToLPC
dsp.LSPToLPC
dsp.RCToAutocorrelation
dsp.RCToLPC
dsp.SpectrumEstimator
dsp.TransferFunctionEstimator

# Optimized libraries for DSPs

## ARM Cortex-M and ARM Cortex-A Optimization R2016a

The DSP System Toolbox™ supports optimized C code generation for popular algorithms like FIR filtering and FFT on ARM® Cortex®-M and ARM Cortex-A processors.

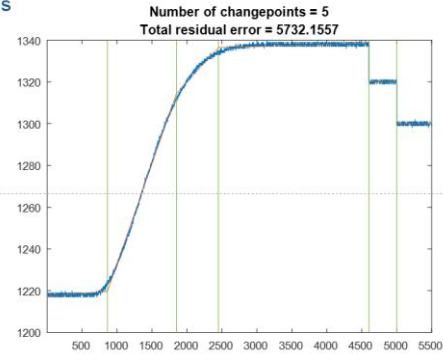


# Some interesting additions...

## Changepoint Detection

Find abrupt changes and statistical shifts in signals

- Determine "interesting" areas of an input signal
- Statistics supported
  - Mean
  - Variance
  - Mean and variance
  - Linear Regression

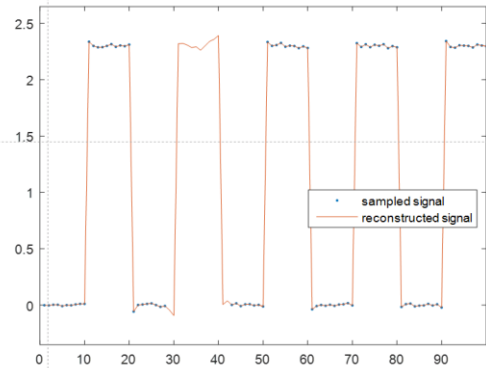


```
>> load('engineRPM.mat','x')
>> findchangepts(x,'Statistic','linear','MinThreshold',var(x))
```

## Gap Filling

Reconstruct missing samples using autoregressive modeling

- Allows finer prediction for many input signals.
- Automatic model selection via Akaike information criterion
- Multiple gaps.
- Optionally model non-stationary signals



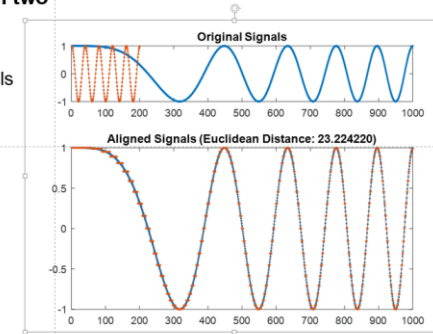
```
>> load clockex
>> x(29:42) = NaN;
>> fillgaps(x)
```

## Dynamic Time Warping

Stretch, align and compare signals with different time scales

Compare and align trajectories between two signals in space

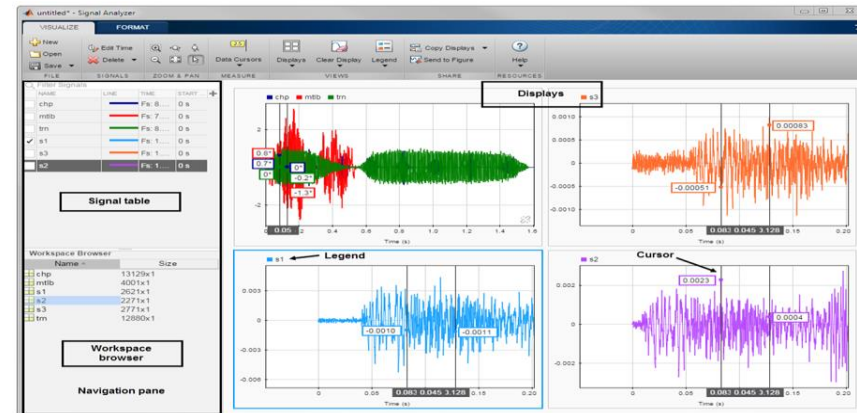
- Obtain a measure of similarity of two signals trajectories.
- Optional time alignment
- Popular distance metrics supported:
  - Euclidean
  - Squared Euclidean
  - Manhattan
  - Symmetric Kullback-Leibler



```
>> x = chirp(0:999,0,1000,1/100);
>> y = cos(2*pi*5*(0:199)/200);
>> dtw(x,y)
```

## Signal Analyzer App

Visualize and compare multiple signals



*Signal Processing*



*Audio*

*Antenna to Bits*

*WLAN/LTE*

*Image and Video Processing*

# Audio System Toolbox

*Design and test audio processing systems*

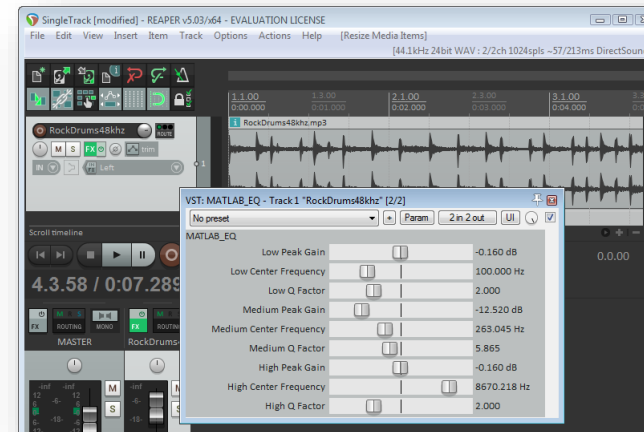
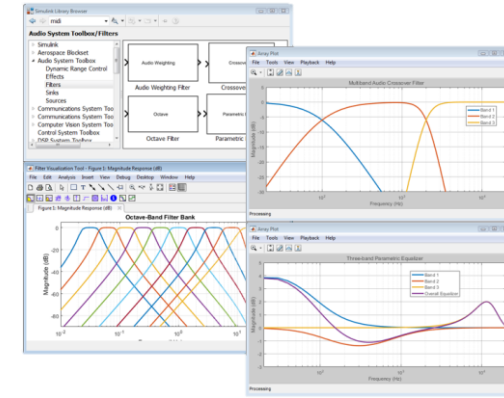


# Audio System Toolbox

*Design and test audio processing systems*



- Libraries of audio processing **algorithms** and examples
- Low-latency audio streaming** from and to standard audio interfaces (e.g. ASIO, CoreAudio, ALSA)
- Live-tuning** of MATLAB and Simulink via UI and MIDI controls
- VST** plugin generation to run on Digital Audio Workstations





# Audio System Toolbox

*Prototyping for product development*

MATLAB  
algorithm



Early validation  
(listening tests)

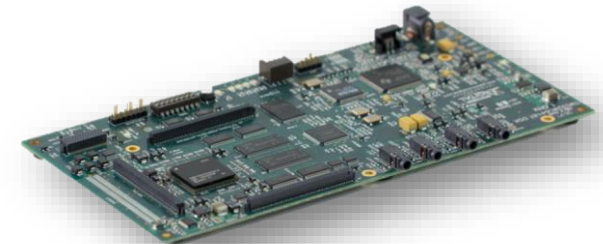
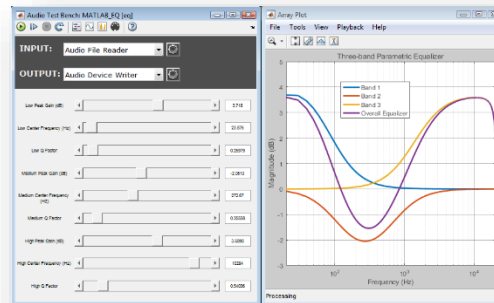


Advanced prototyping  
or production

```
classdef VolumeControl < AudioPlugin %codegen
    properties
        % Public properties that are not constant and not
        % become tunable plugin parameters and appear on
        % dialog. A default value is required.
        Volume = 1
    end
    properties (Constant)
        % Constant property 'DisplayName' specifies the
        % displayed by the DAW.
        DisplayName = 'Volume control'

        % Appearance and display of plugin parameters is
        % constant properties having the 'Display' suffix
        % name displayed on the dialog
        % units displayed on the dialog
        % minimum value
        % maximum value
        % scaling law; one of {'lin' 'log' 'fader'
        % scaling power (if law is 'pow')

        % Volume faders ordinarily vary the gain from so
        % silence (-inf dB, or UV/V). Here the maximum
        % The 'fader' law is cubic, which closely mimics
        % controls.
        VolumeDisplay = AudioPlugin.paramDisplay('Volume')
    end
    methods
        function out = process(plugin, in)
            % Processing to be applied to each frame of
            % Plugin input and output parameters are always
            % channel; use multiple parameters to get multiple
            % Frame size is not fixed, and may vary from
            % Thus arguments will always be var-sized
            % This method runs in the high priority audio
            % thread, so be as efficient as possible.
        end
    end
end
```

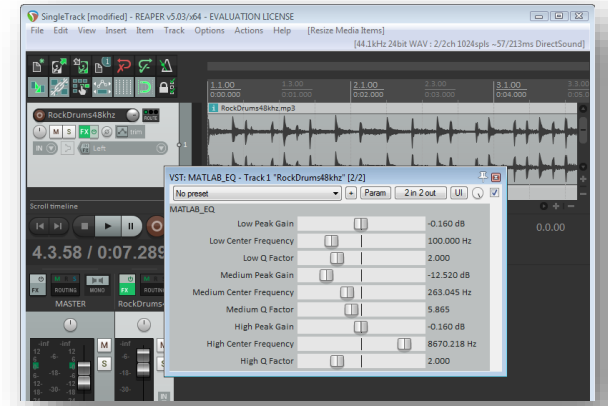


# Audio System Toolbox

## Use cases summary

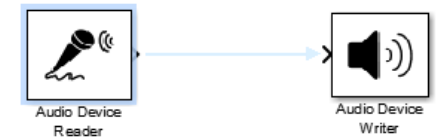
- Desktop prototyping and listening tests**

- Pain: prototyping costly and time-consuming
  - Solution: real-time audio streaming in MATLAB and VST plugin generation



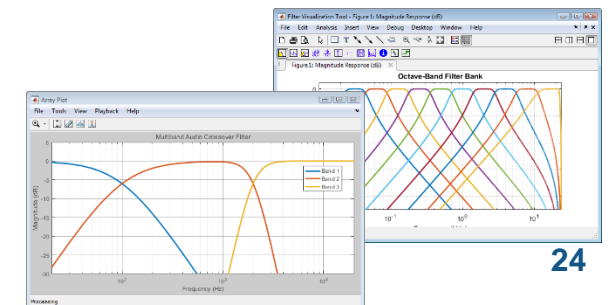
- Real-time custom measurements and signal analysis**

- Pain: test & measurement equipment not available or not customizable
  - Solution: real-time audio acquisition and *unlimited* custom analysis



- Audio algorithm design**

- Pain: re-inventing consolidated algorithms time-consuming
  - Solution: libraries of audio processing algorithms and examples





# Audio System Toolbox

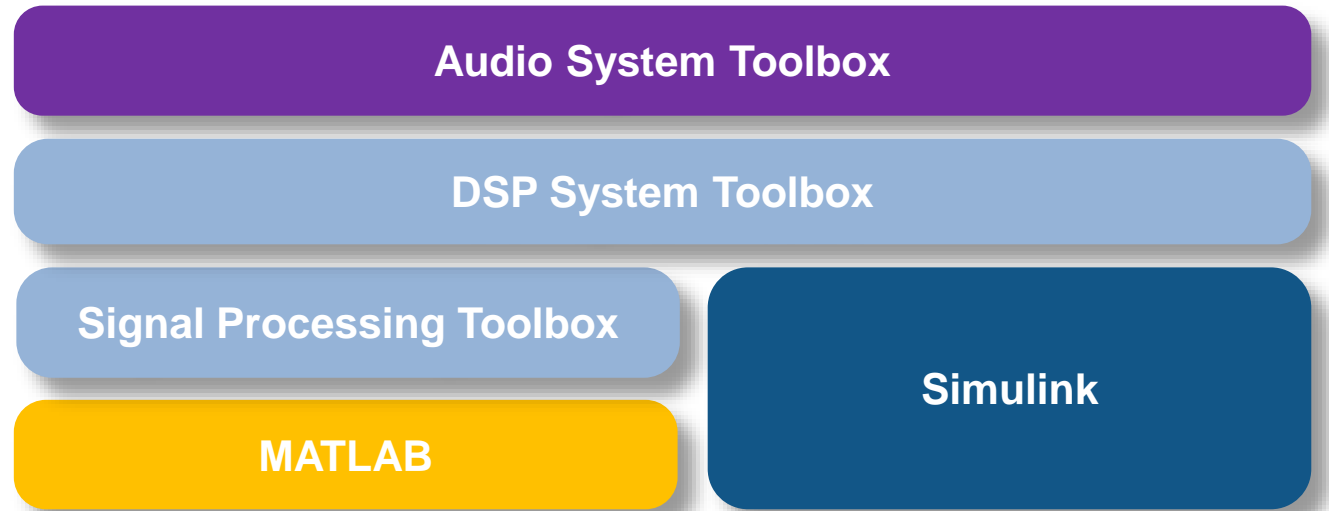
*Product ecosystem*

## Requires

- MATLAB
- Signal Processing Toolbox
- DSP System Toolbox

## Supports

- MATLAB
- Simulink
- C/C++ Code Generation



*Signal Processing*

*Audio*



*Antenna to Bits*

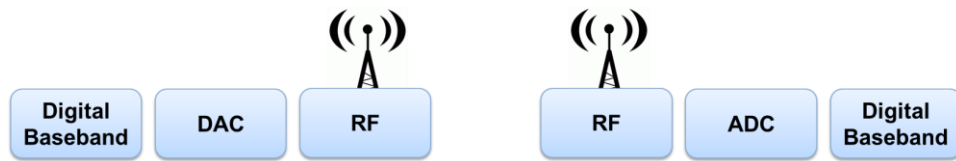
*WLAN/LTE*

*Image and Video Processing*

# Antenna to Bits

## System Design and Modelling

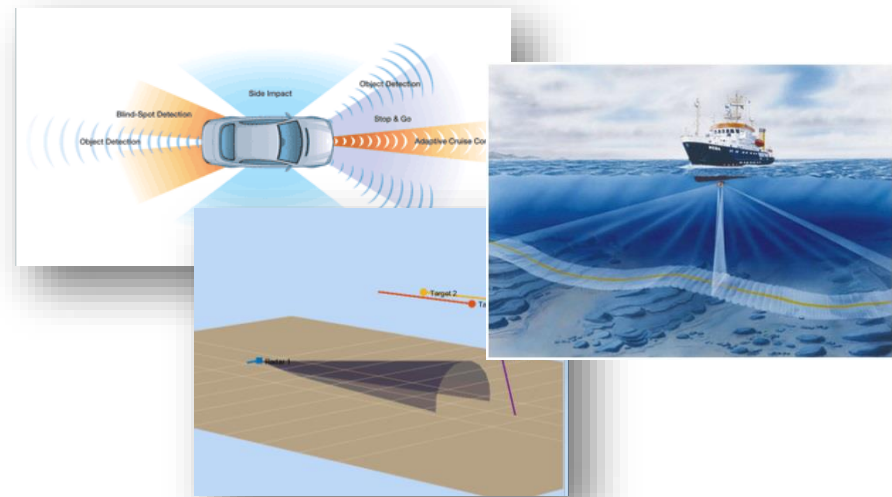
### Communications Systems



- System Partitioning
- Link Budget Simulations
- System Integration

- Elaborating RF Architecture
- Component Simulation
- RF Subsystem Simulation

### Radar / Sonar / Sensor Arrays



# Antenna to Bits

## System Design and Modelling

Antenna, Antenna arrays  
*type of element, # elements, coupling, edge effects*

- Antenna Toolbox
- Phased Array System Toolbox

Mixed-Signal  
*Continuous & discrete time*

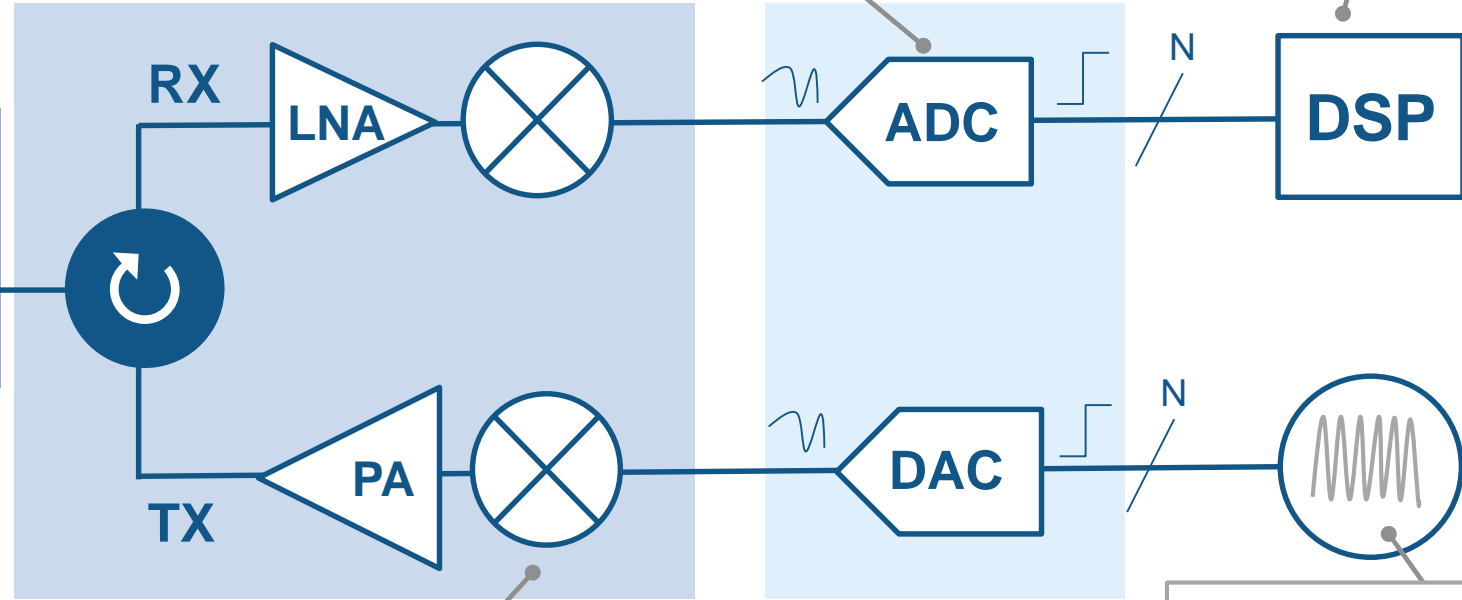
- Simulink (Simscape)
- DSP System Toolbox
- Control System Toolbox

Algorithms  
*beamforming, beamsteering, MIMO*

- Phased Array System Toolbox
- Communications System Toolbox

- Communications System Toolbox
- Phased Array System Toolbox

Channel  
*interference, clutter, noise*



- SimRF
- RF Toolbox

RF Impairments  
*frequency dependency, non-linearity, noise, mismatches*

- Phased Array System Toolbox
- Instrument Control Toolbox

Waveforms<sup>28</sup>

# Antenna to Bits

## System Design and Modelling

Antenna, Antenna arrays  
*type of element, # elements, coupling, edge effects*

- Antenna Toolbox
- Phased Array System Toolbox

Mixed-Signal  
*Continuous & discrete time*

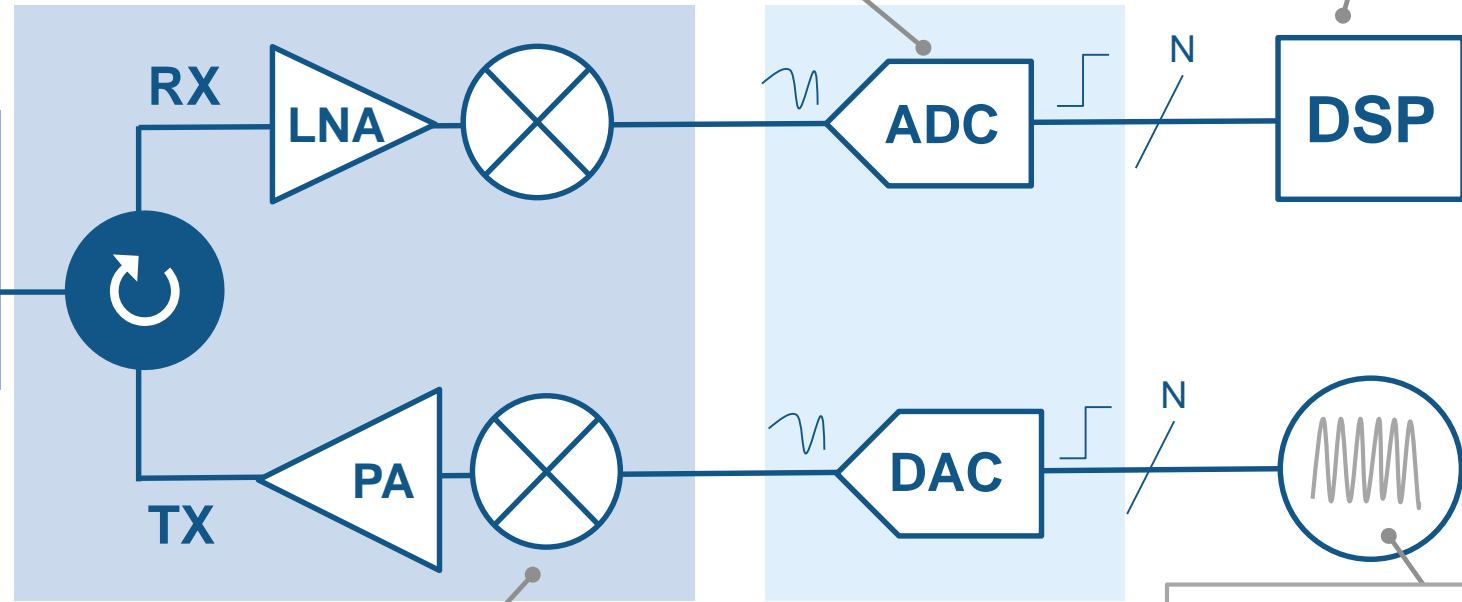
- Simulink (Simscape)
- DSP System Toolbox
- Control System Toolbox

Algorithms  
*beamforming, beamsteering, MIMO*

- Phased Array System Toolbox
- Communications System Toolbox

- Communications System Toolbox
- Phased Array System Toolbox

Channel  
*interference, clutter, noise*



- SimRF
- RF Toolbox

RF Impairments  
*frequency dependency, non-linearity, noise, mismatches*

- Phased Array System Toolbox
- Instrument Control Toolbox

Waveforms<sup>29</sup>

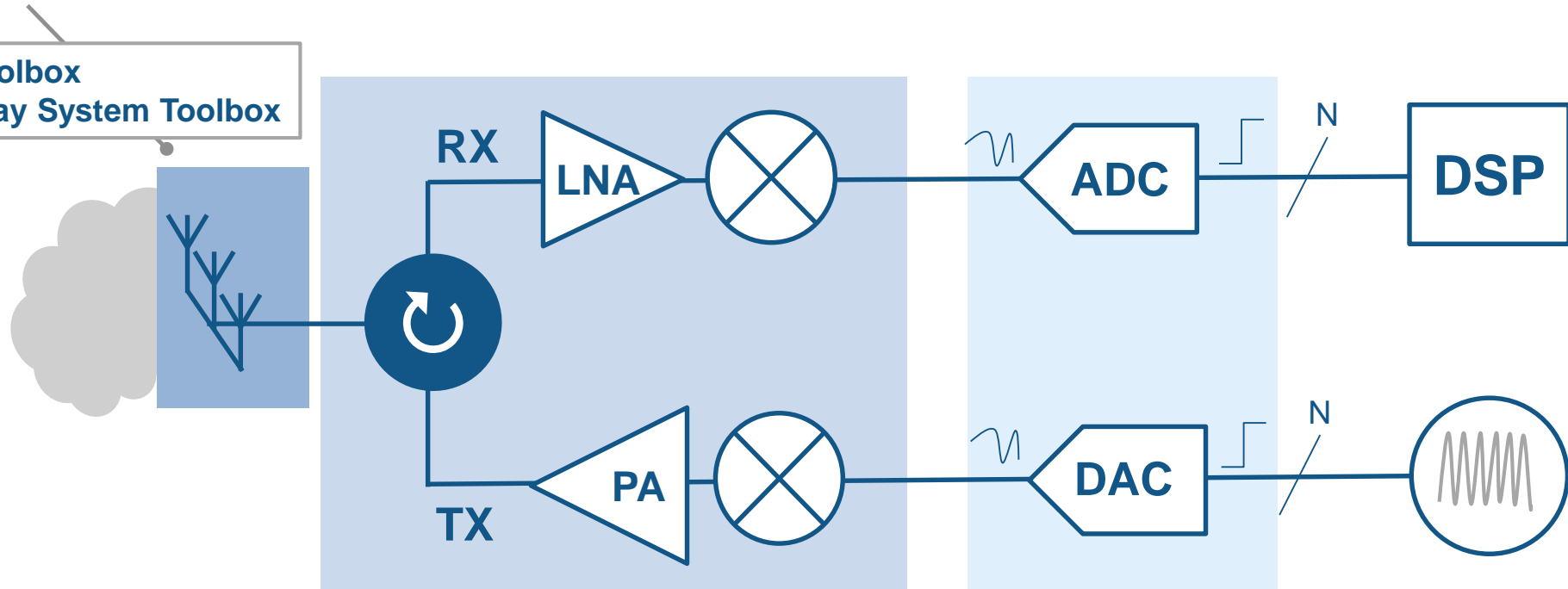
# Antenna to Bits

## System Design and Modelling

Antenna, Antenna arrays

*type of element, # elements, coupling, edge effects*

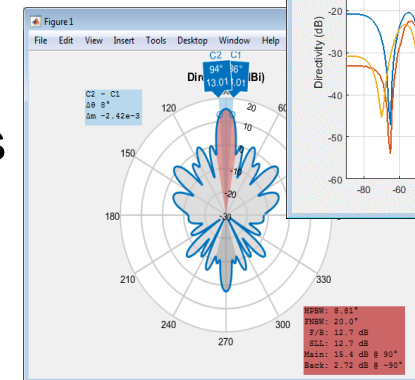
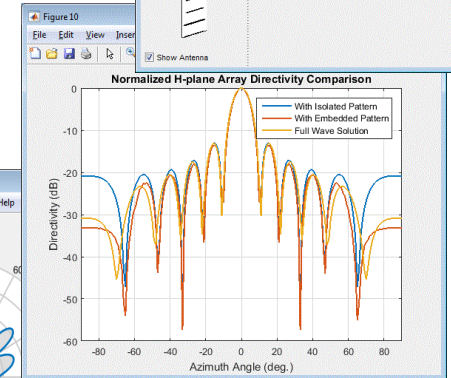
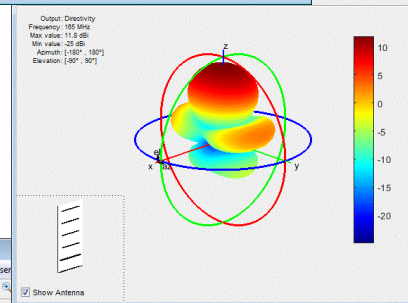
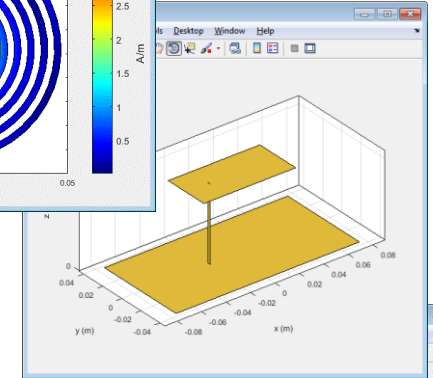
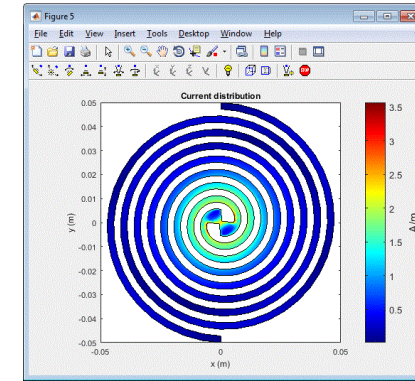
- Antenna Toolbox
- Phased Array System Toolbox



# Antenna Toolbox

## *Design, simulation and integration*

- **Easy design**
  - Library of parameterized antenna elements
  - Functionality for the design of linear and rectangular antenna arrays
  - No need for full CAD design
  
- **Rapid simulation setup**
  - Method of Moments field solver for port, field, and surface analysis
  - No need to be an EM expert
  
- **Seamless integration**
  - Model the antenna together with signal processing algorithms
  - Rapid iteration of different antenna scenarios for radar and communication systems design

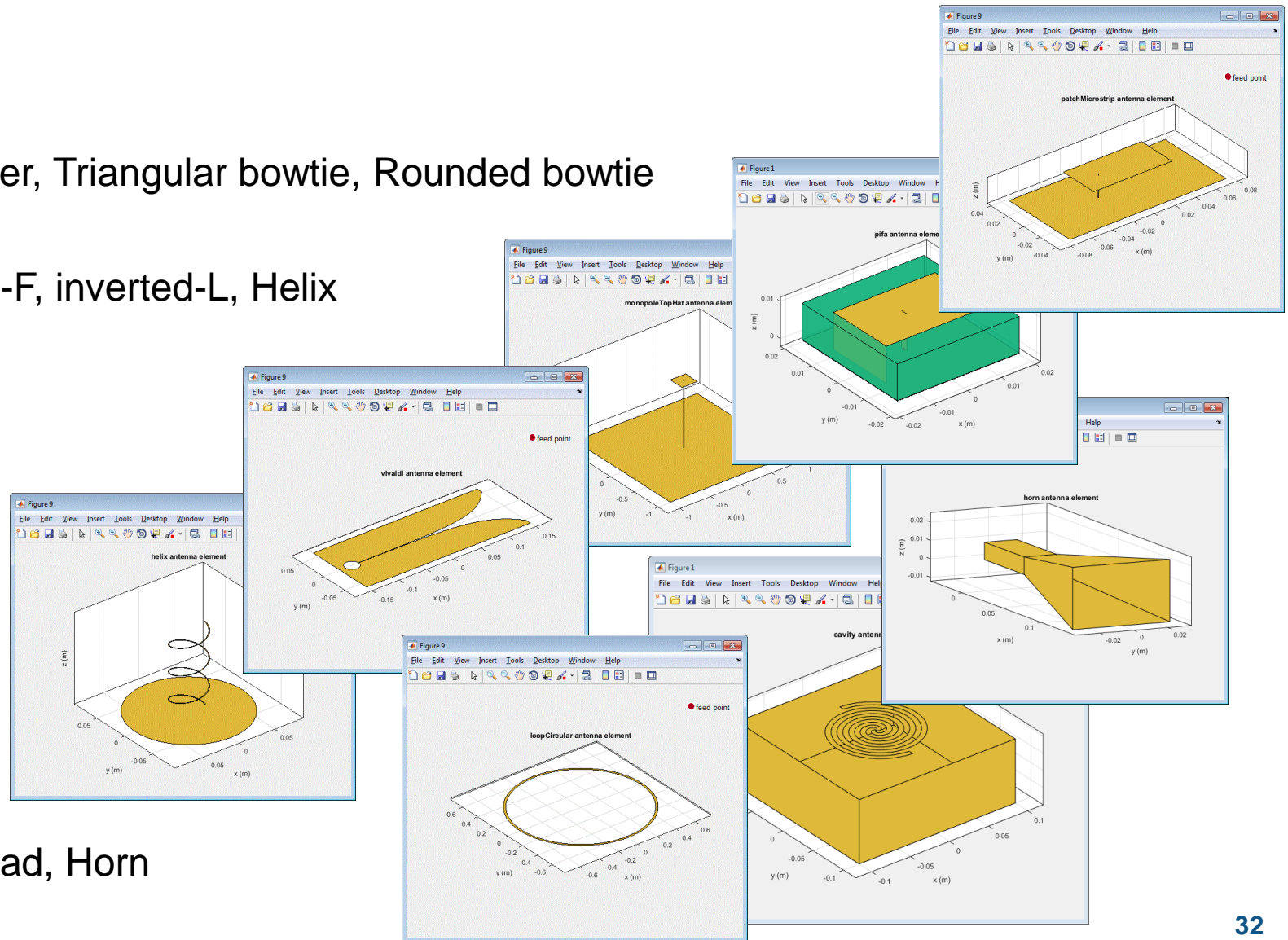




# Antenna Toolbox

## Library of Available Geometries

- Dipole antennas
  - Dipole, Vee, Folded, Meander, Triangular bowtie, Rounded bowtie
- Monopole antennas
  - Monopole, Top hat, Inverted-F, inverted-L, Helix
- Patch antennas
  - Microstrip patch, PIFA
- Spirals
  - Equiangular, Archimedean
- Loops
  - Circular, rectangular
- Backing structures
  - Reflector and cavity
- Other common antennas
  - Yagi Uda, Slot, Vivaldi, Biquad, Horn

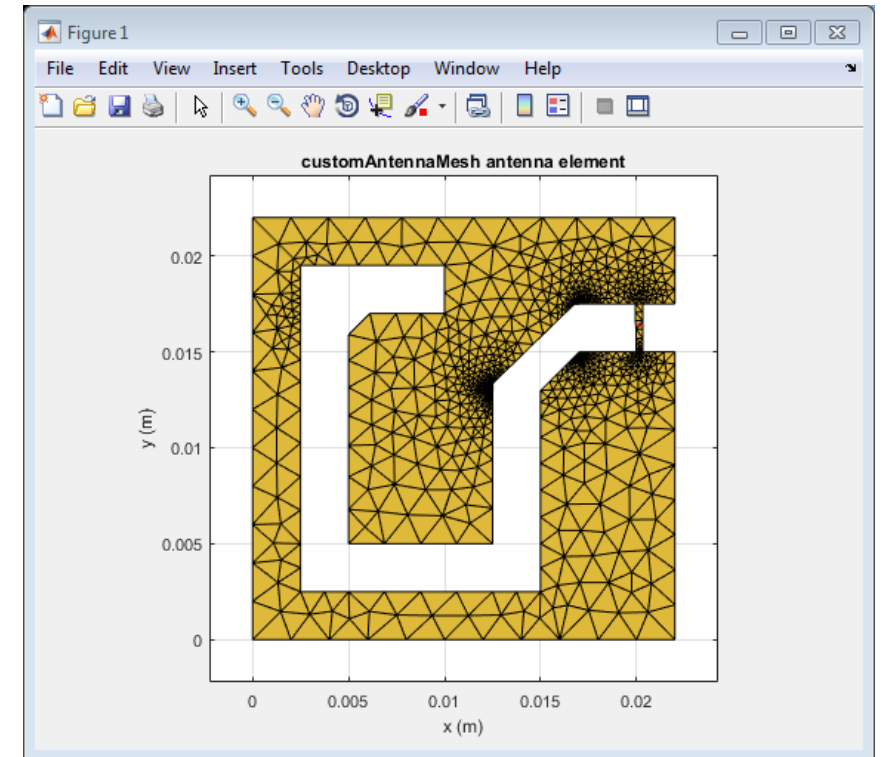




# Antenna Toolbox

## *Custom Antenna Element Design*

- Define your custom planar structure
  - Define the antenna geometry using PDE Toolbox
  - Define the mesh using MATLAB  
`deLaunayTriangulation`
  - Use third party tools to generate a mesh structure
- Import 2D mesh with Antenna Toolbox
  - Define the feeding point
  - Analyse the antenna



# Antenna Toolbox

## Dielectric Substrate Modelling

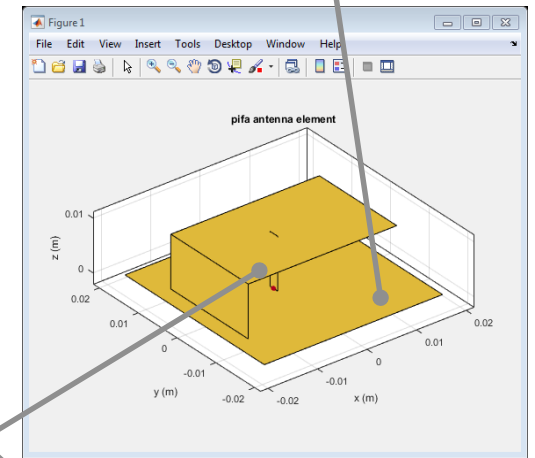
- Antenna are often mounted on **substrates**
- Dielectric properties:

Dielectric	Relative permittivity	Loss Tangent
Air	1	0
Other	>1 (typically <10)	>0 (typically ~1e-3)

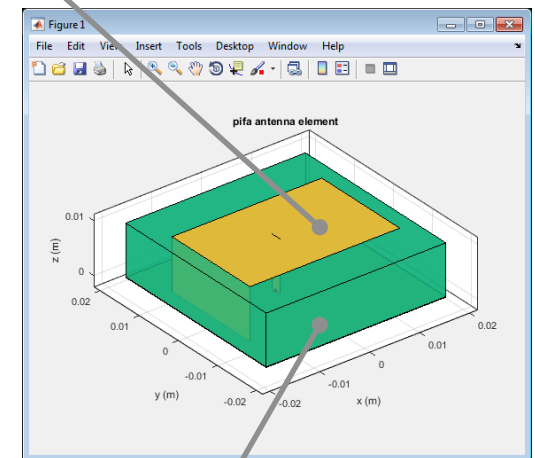
- Dielectric properties **affect resonance, bandwidth, efficiency, pattern ...**
- Use the dielectric catalogue listing existing materials
- Define your **own** dielectric material

“metal” antenna  
(ideal conductor)

Free space (isolation)



Name	Relative_Permittivity	Loss_Tangent	Frequency	Comments
1 Air	1	0	1.0000e+009	
2 FR4	4.8000	0.0280	100.0000e+009	
3 Teflon	2.1000	2.0000e-04	100.0000e+009	
4 Foam	1.0300	1.5000e-04	50.0000e+008	
5 Polystyrene	2.5500	1.0000e-04	100.0000e+009	
6 Plexiglas	2.5900	0.0068	10.0000e+009	
7 Fused quartz	3.7800	1.0000e-04	10.0000e+009	
8 E glass	6.2200	0.0023	100.0000e+009	
9 RO4725JXR	2.5500	0.0022	2.5000e+009	
10 RO4730JXR	3	0.0023	2.5000e+009	
11 TMM3	3.4500	0.0020	10.0000e+009	
12 TMM4	4.7000	0.0020	10.0000e+009	
13 TMM6	6.3000	0.0023	10.0000e+009	
14 TMM10	9.8000	0.0022	10.0000e+009	
15 TMM10a	9.9000	0.0020	10.0000e+009	
16 Taconic RF-35	3.5000	0.0018	1.9000e+009	



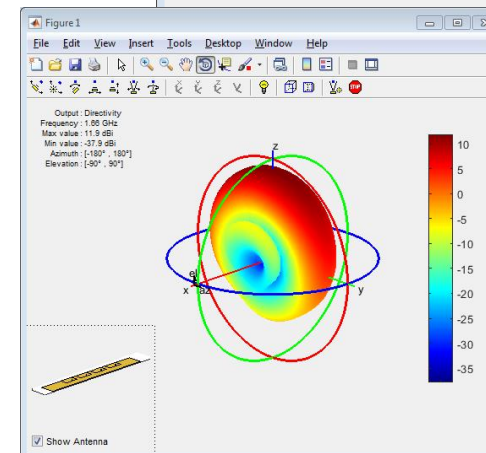
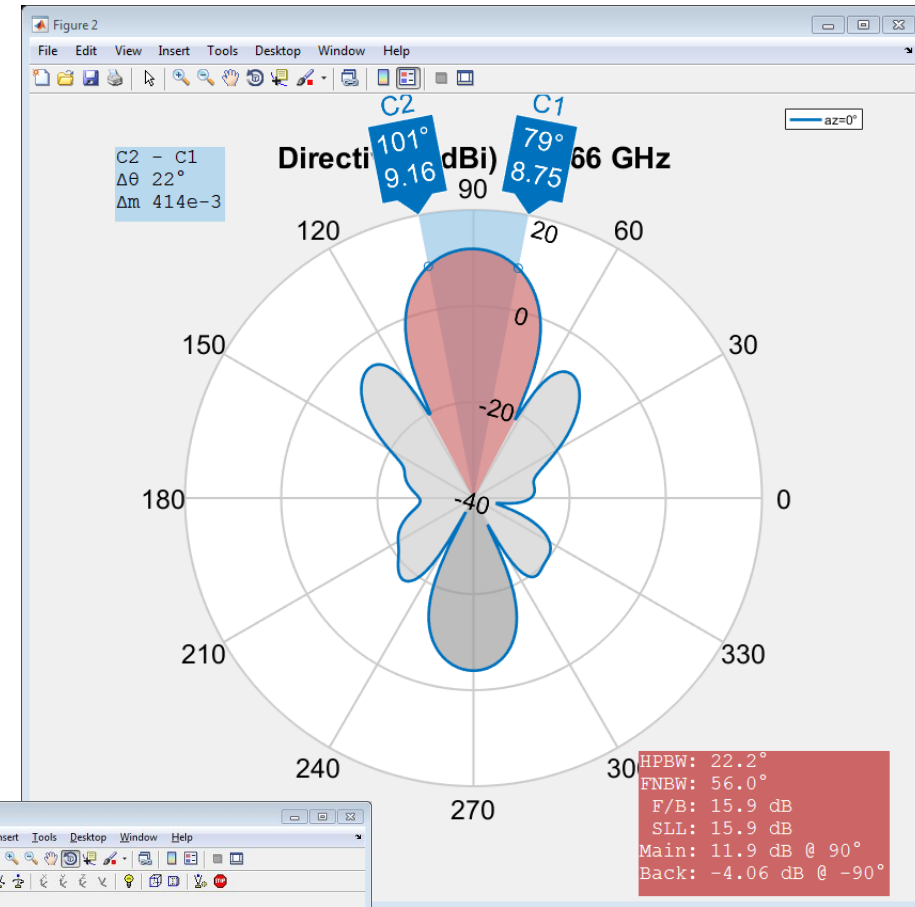
Dielectric substrate 34

*From antenna element to antenna array...*

# Phased Array System Toolbox

## Array Antenna Design

```
>> a = linearArray
>> a.Element = p;
>> a.ElementSpacing = 0.1;
>> a.NumElements = 4;
>> layout(a);
>> patternElevation(a, 1.66e9, 0);
```

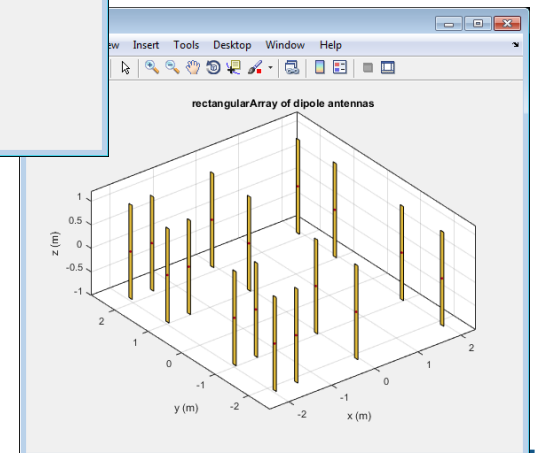
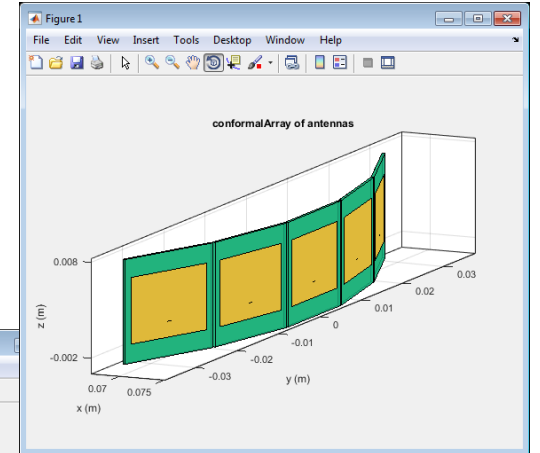
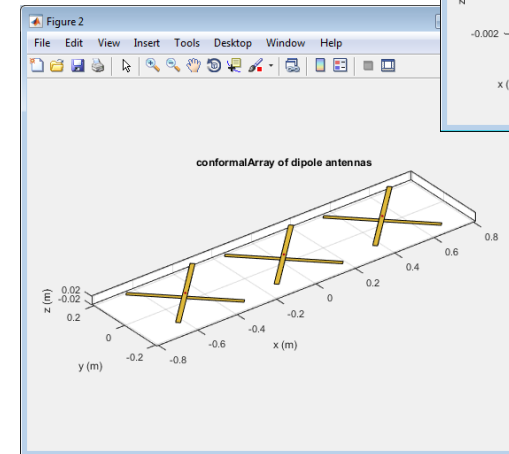


# Phased Array System Toolbox

## Custom Array Antenna Design

- Build regular arrays where you can change the **properties of individual elements** (rotation, size, tapering)
- Describe conformal (heterogeneous) arrays in terms of element type and arbitrary position

```
>> arr = conformalArray;
>> d = dipole;
>> b = bowtieTriangular;
>> arr.Element = {d, b};
>> arr.ElementPosition(1,:) = [0 0 0];
>> arr.ElementPosition(2,:) = [0 0.5 0];
```

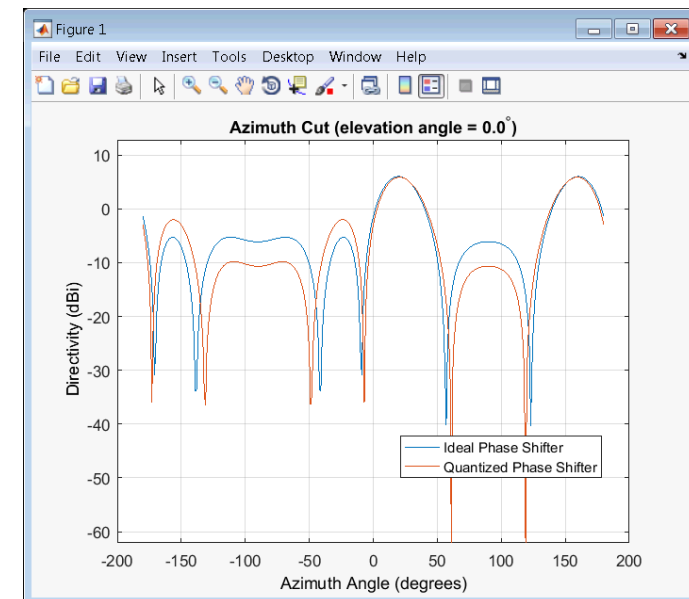
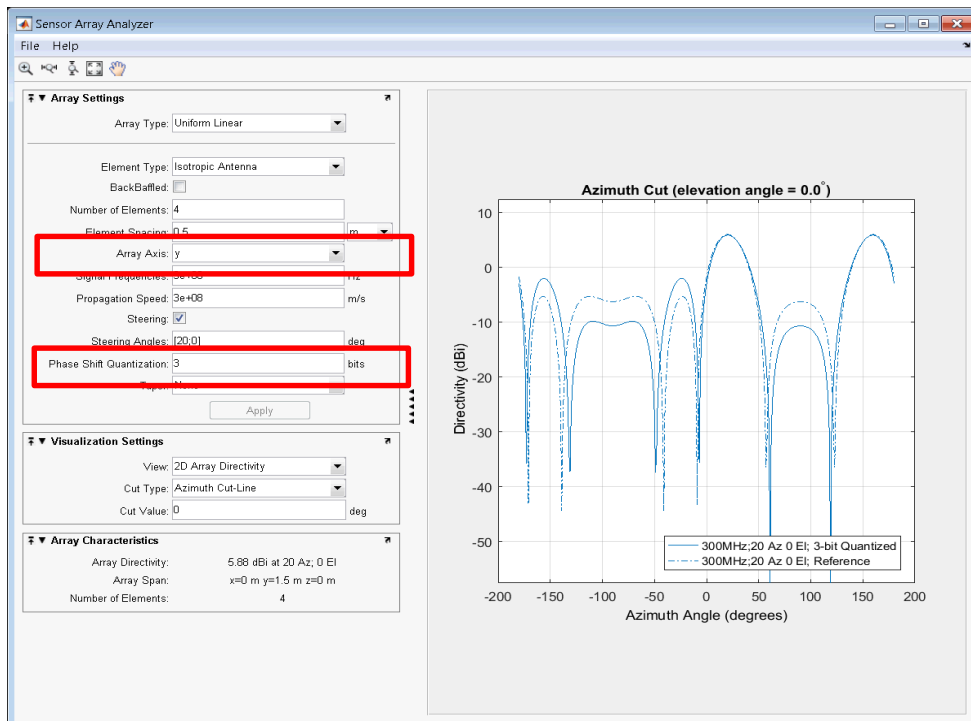


# Phased Array System Toolbox

## Model effects of quantized phase shift values on array patterns and responses

- Many phase shifters in real systems are quantized
- Allow customer to quickly see the effect of phase shifter quantization

```
ant = phased.ULA(4);
sv = phased.SteeringVector('SensorArray', ant);
w1 = step(sv, 3e8, [20;10]);
release(sv);
sv.NumPhaseShifterBits = 3;
w2 = step(sv, 3e8, [20;10]);
c = sv.PropagationSpeed;
pattern(ant, 3e8, -180:180, 0, 'PropagationSpeed', c, 'Weights', [w1 w2], ...
    'CoordinateSystem', 'rectangular');
legend('Ideal Phase Shifter', ...
    'Quantized Phase Shifter', 'Location', 'Best')
```



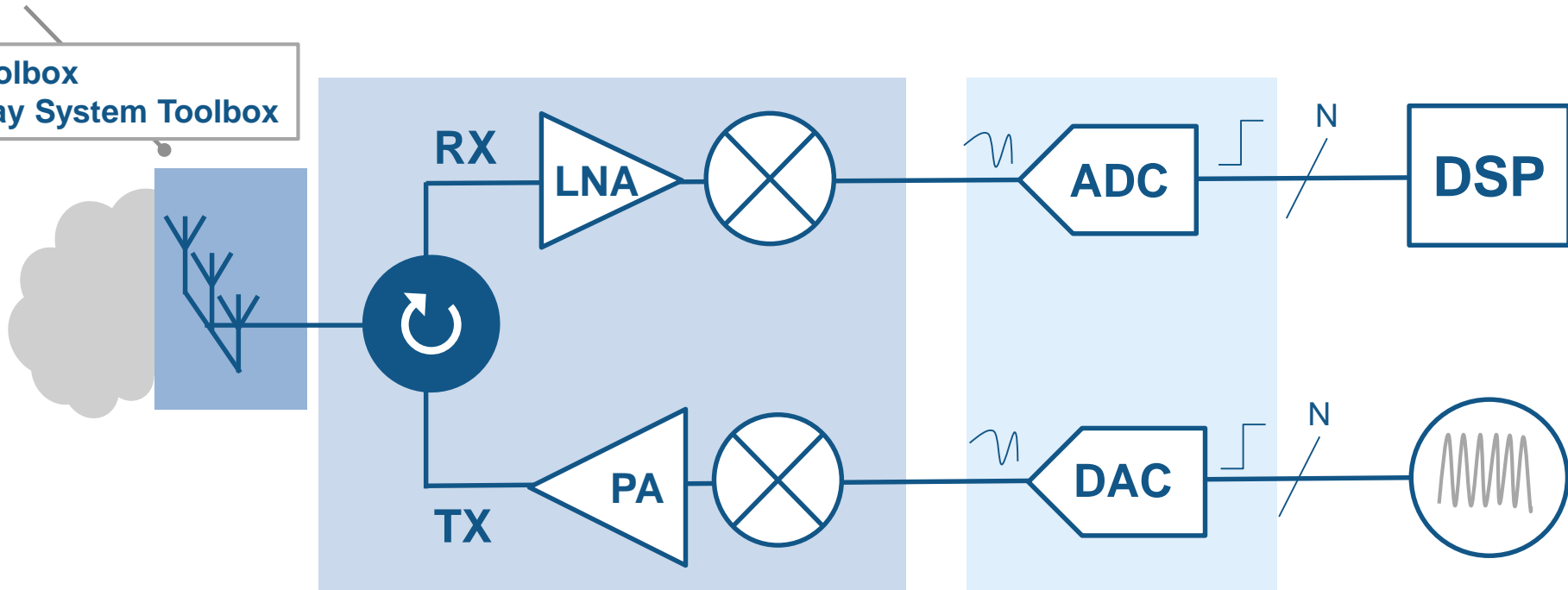
# Antenna to Bits

## System Design and Modelling

Antenna, Antenna arrays

*type of element, # elements, coupling, edge effects*

- Antenna Toolbox
- Phased Array System Toolbox

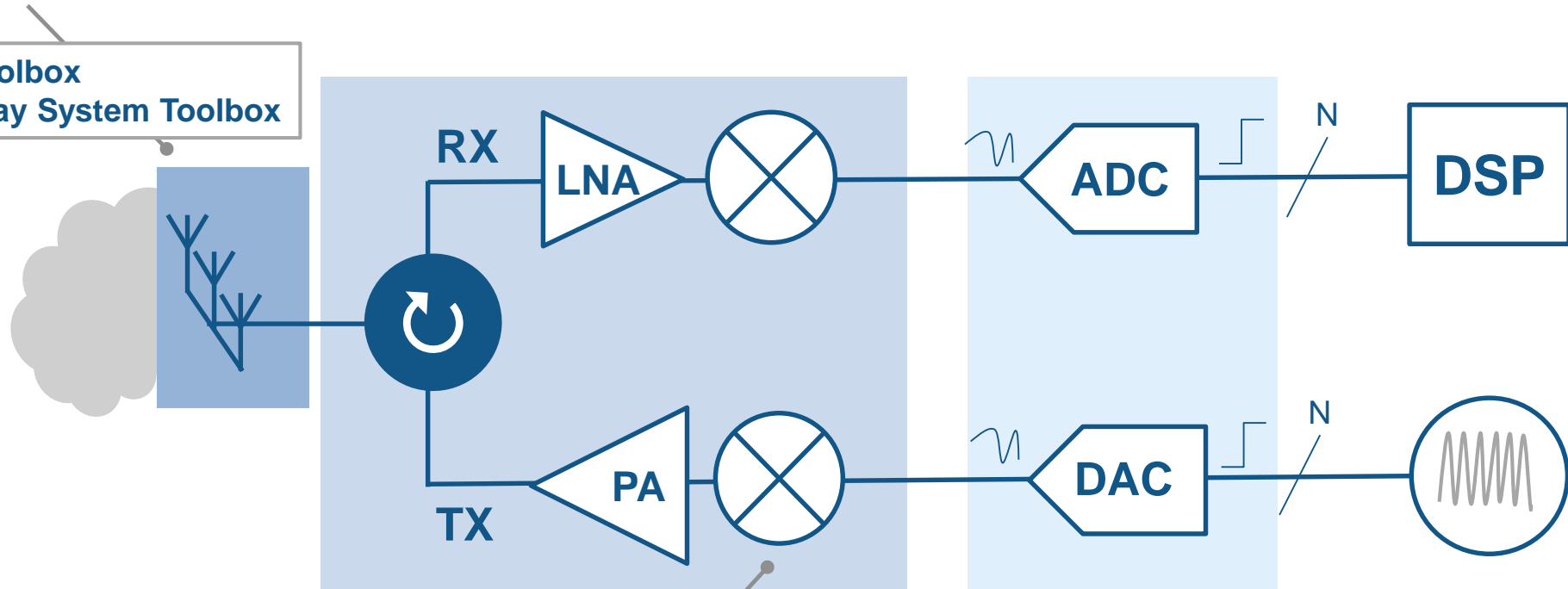


# Antenna to Bits

## System Design and Modelling

Antenna, Antenna arrays  
*type of element, # elements, coupling, edge effects*

- Antenna Toolbox
- Phased Array System Toolbox



- SimRF
- RF Toolbox

RF Impairments

*frequency dependency, non-linearity, noise, mismatches*



# RF Toolbox

## RF Budget Analyzer

- Analytically compute gain, noise figure, and IP3 for cascaded RF components
- Specify components in terms of data sheet parameters and S-parameters
- Analyse the RF chain taking into account impedance mismatches

Add RF components      Export to SimRF      RF Cascade

The screenshot shows the RF Budget Analyzer interface. The top toolbar includes buttons for 'Amplifier', 'Modulator', 'S-parameters', 'Generic', and 'Export'. The main workspace displays a block diagram of an RF chain with components: RF\_Filter, LNA, Demodula..., IF\_Filter, and IF\_Amplifier. Below the diagram are two tables: 'Stage' and 'Cascade'. The 'Element Parameters' panel on the left shows specifications for an LNA component.

**Component specifications**

System Parameters	
Input frequency:	10 GHz
Available input power:	-97.5 dBm
Signal bandwidth:	5.9958 MHz

Element Parameters	
Amplifier	
Name:	LNA
Available power gain:	12 dB
Noise figure:	1.9 dB
OIP3:	21 dBm
Input impedance:	50 Ohm
Output impedance:	50 Ohm

**Cascade Budget Analysis**

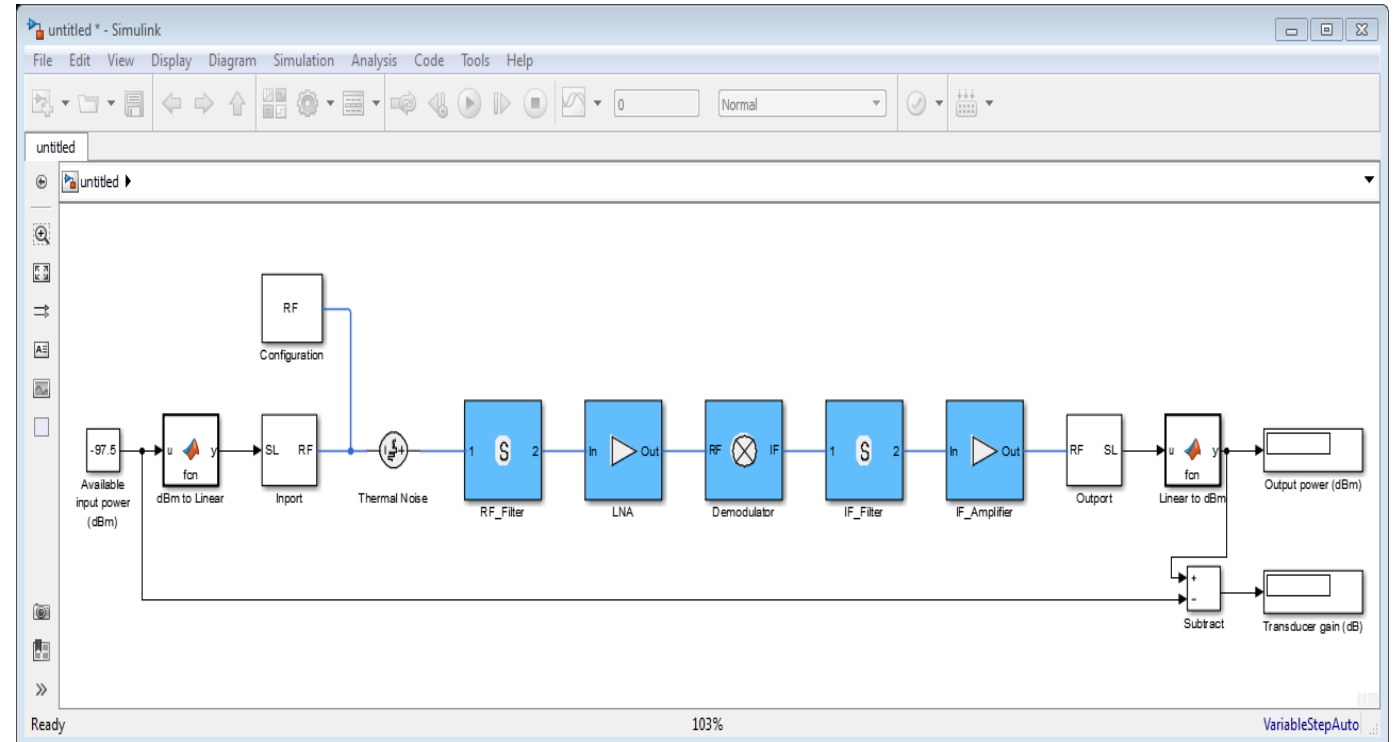
Stage	1	2	3	4	5
GainA (dB)	-2.583	12	-6	-0.1395	20
NF (dB)	2.583	1.9	5.4	0.1395	6
OIP3 (dBm)	Inf	21	Inf	Inf	Inf

Cascade	1	1..2	1..3	1..4	1..5
Fout (GHz)	10	10	0.07	0.07	0.07
Pout (dBm)	-100.4	-88.35	-94.35	-94.49	-74.49
GainT (dB)	-2.854	9.146	3.146	3.006	23.01
NF (dB)	2.583	4.581	5.013	5.035	6.702
OIP3 (dBm)	Inf	21	15	14.86	34.86
SNR (dB)	6.114	4.116	3.684	3.662	1.995

# RF Toolbox

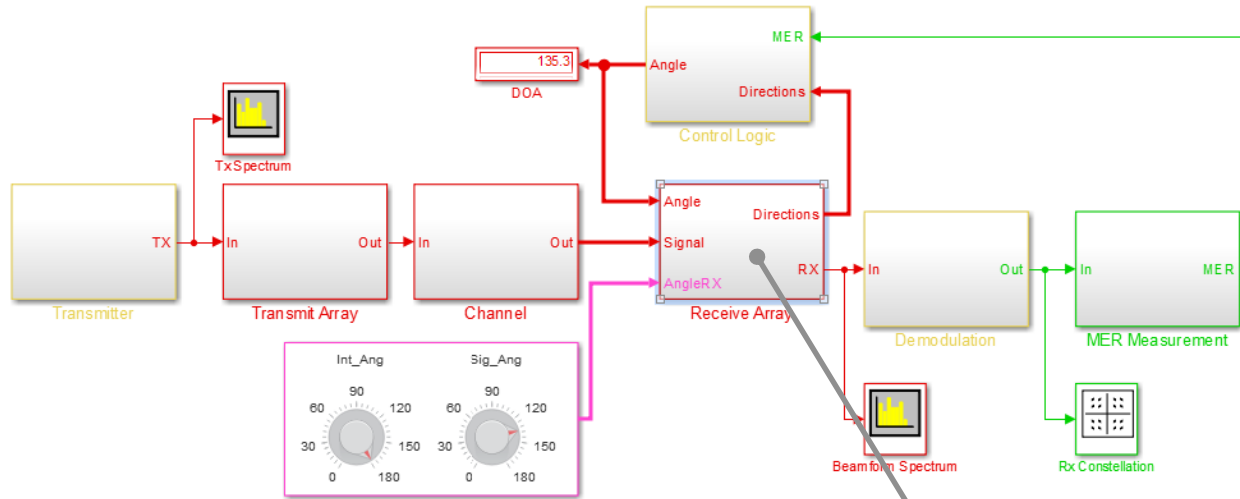
## RF Budget Analyzer | Export to Sim RF

- Automatic testbench and SimRF model generation using the RF Budget Analyser App
- Validate simulation results using analytical computations
- Rapidly get started with Circuit Envelope simulation



# Sim RF

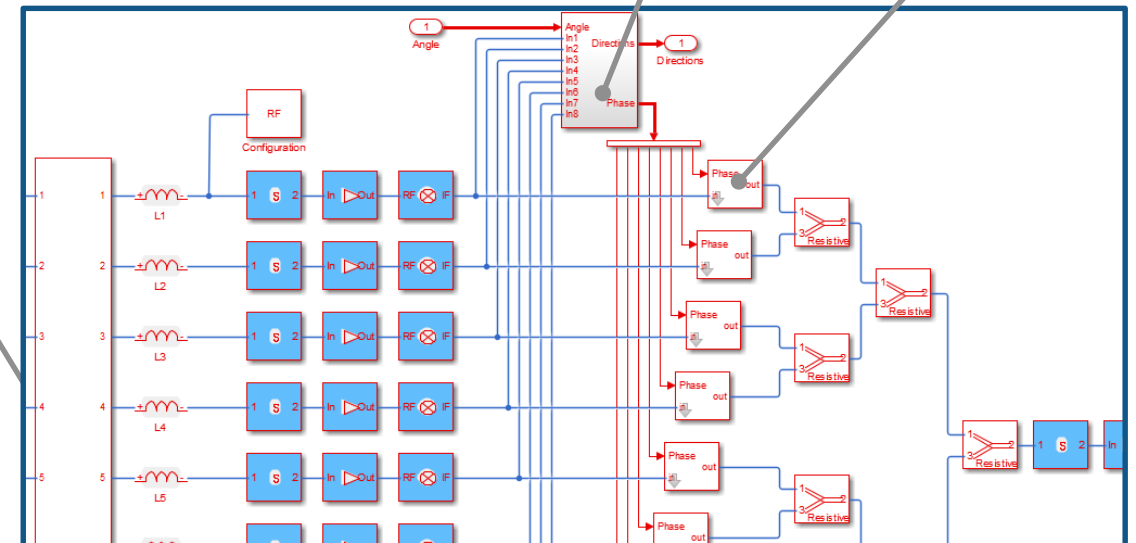
## Example | MIMO Front End with RF Beamforming



Estimation of direction of arrival

RF phase shifting

- Antenna coupling and loading (S-parameters)
- Antenna matching network
- RF and IF Filters described with Touchstone files
- IF demodulation with image rejection
- Non-linearity of the amplifiers
- Thermal Noise
- RF phase shifting and signal combiners



# Sim RF

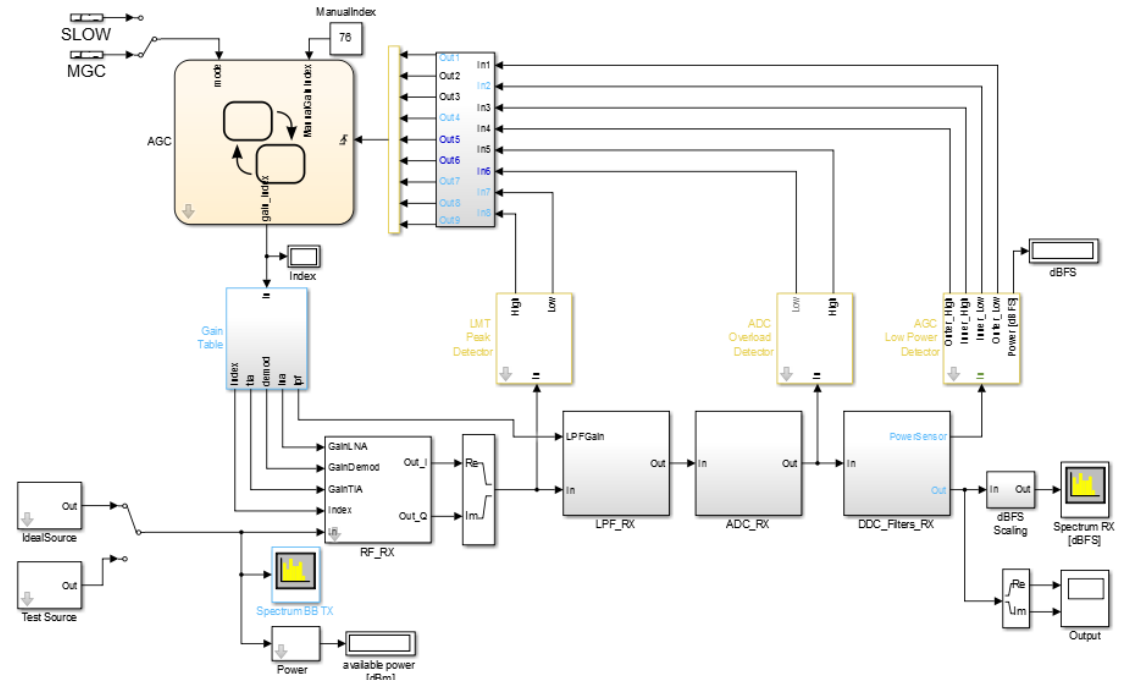
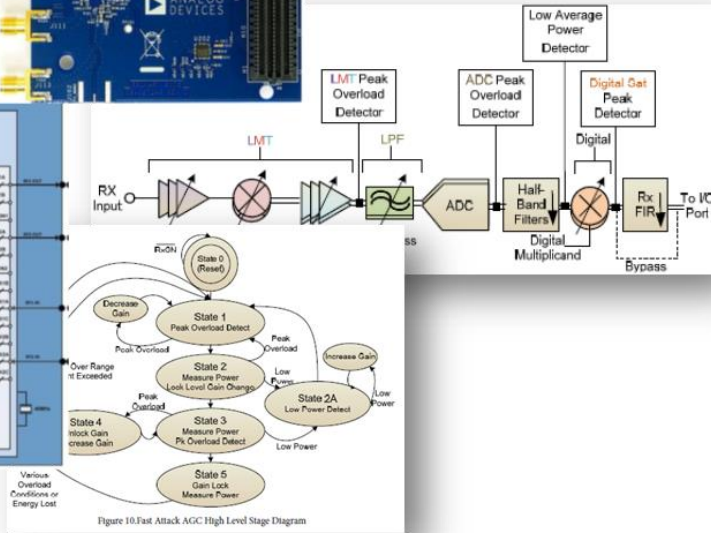
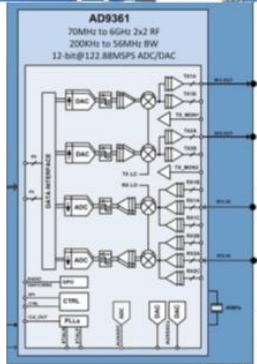
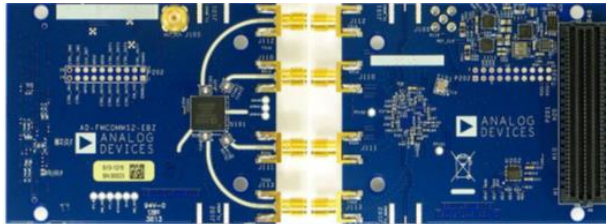
*New and faster implementation of the AD9361 transmitter and receiver*

## AD9361

RF Agile Transceiver™

70 MHz – 6000 MHz Tuning range

200kHz – 56 MHz RF channel Bandwidth



<http://www.mathworks.com/adi-rf>

# Antenna to Bits

## System Design and Modelling

Antenna, Antenna arrays  
type of element, # elements, coupling, edge effects

- Antenna Toolbox
- Phased Array System Toolbox

Mixed-Signal  
Continuous & discrete time

- Simulink (Simscape)
- DSP System Toolbox
- Control System Toolbox

Algorithms  
beamforming, beamsteering, MIMO

- Phased Array System Toolbox
- Communications System Toolbox

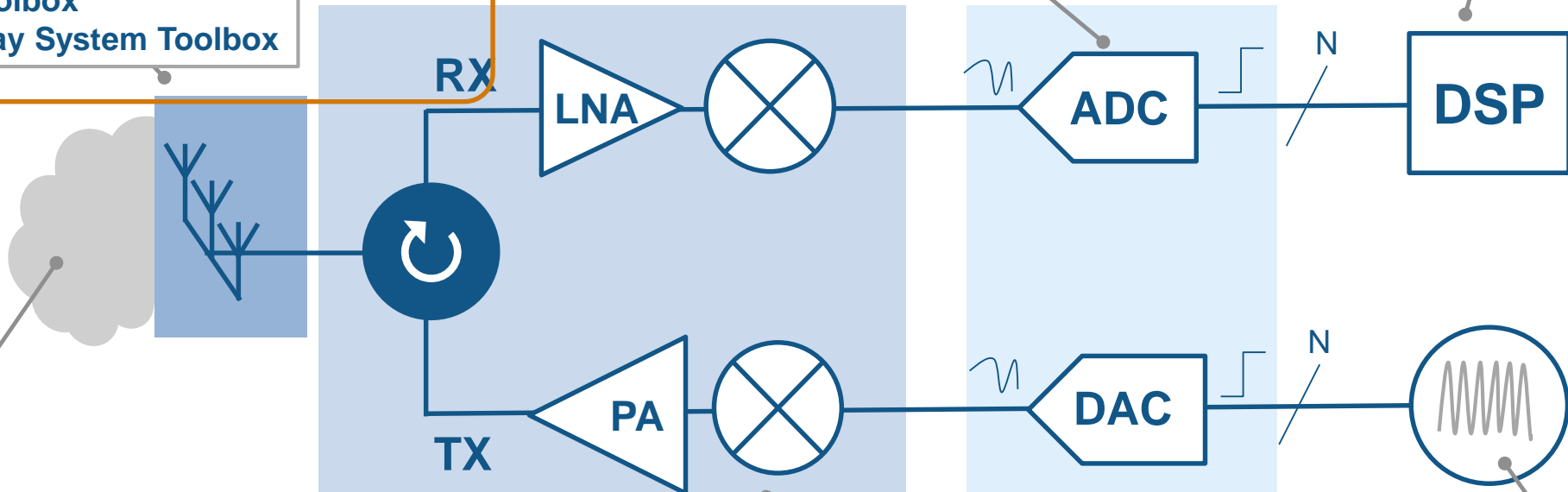
- Communications System Toolbox
- Phased Array System Toolbox

Channel  
interference, clutter, noise

RF Impairments  
frequency dependency, non-linearity, noise, mismatches

- Phased Array System Toolbox
- Instrument Control Toolbox

Waveforms<sup>45</sup>



*Signal Processing*

*Audio*

*Antenna to Bits*



*WLAN/LTE*

*Image and Video Processing*

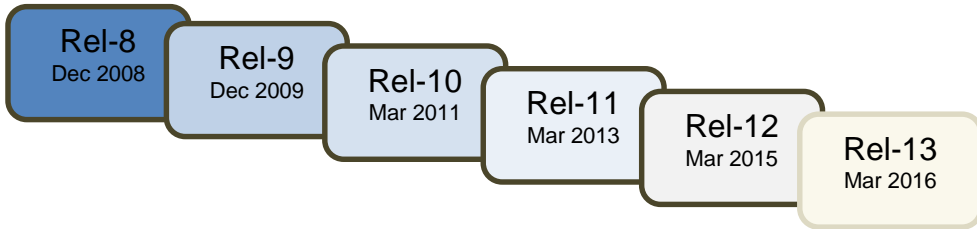
# WLAN/LTE and beyond...

## Evolution of Air Interface Technologies

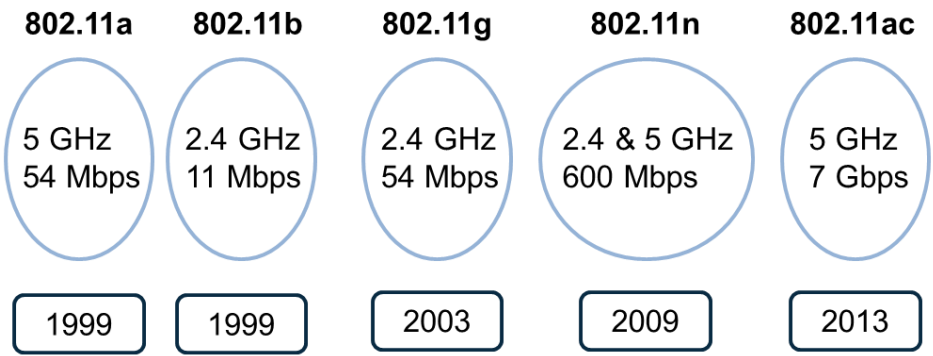
### 4G



### 3GPP LTE, LTE-A



### IEEE 802.11 WLAN standards



### 5G?



5G  
standardization

#### Requirements

- Higher data rates
- More flexible spectrum use
- Spatial resource
- Low delay & link adaptability
- Reliable service everywhere

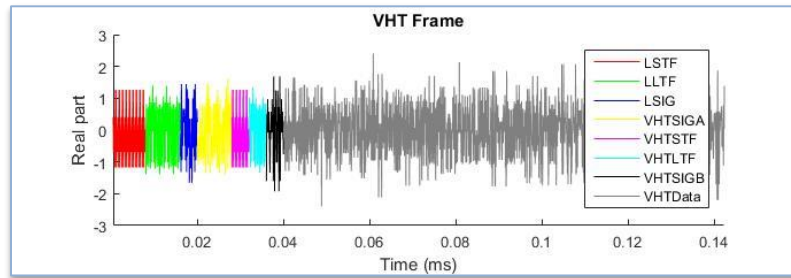
#### Proposed enabling technologies

- Massive MIMO
- Small Cell, HetNet
- New Modulations
- New Frequency bands

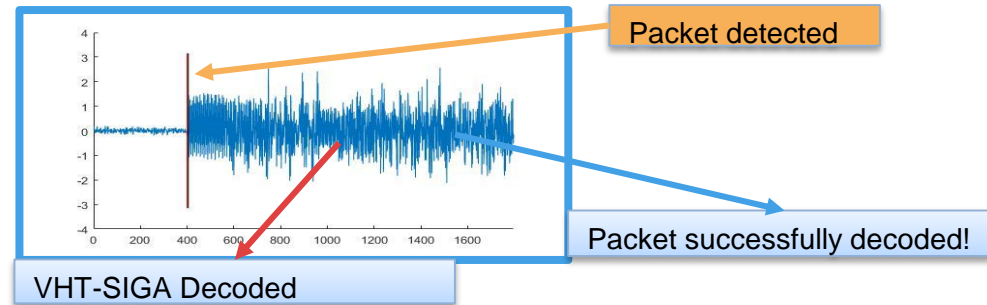
# WLAN/LTE

## Workflow/Use-cases of wireless designers

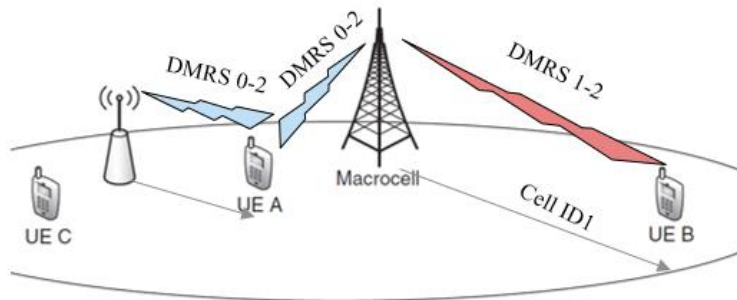
### Signal Generation



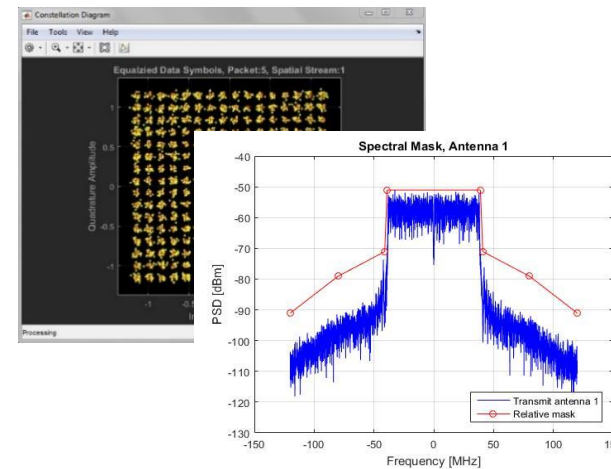
### Signal Detection



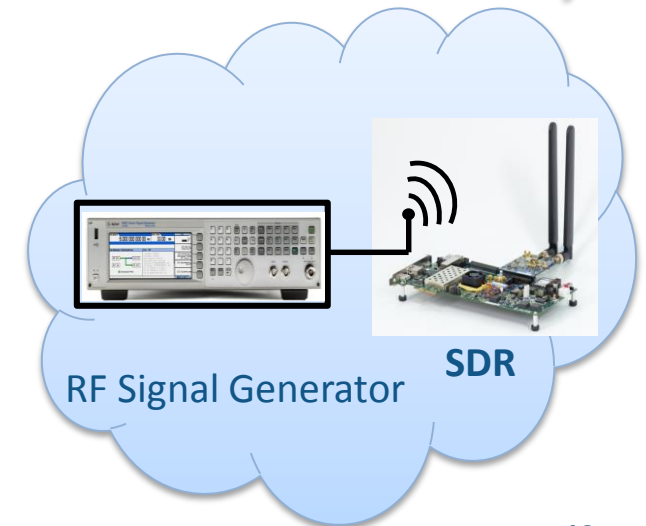
### End-to-End Simulations



### Measurements



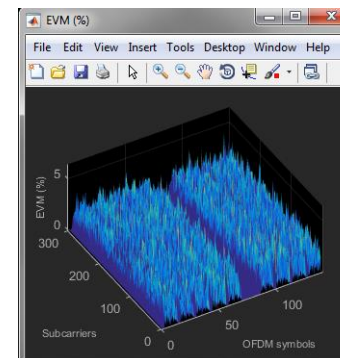
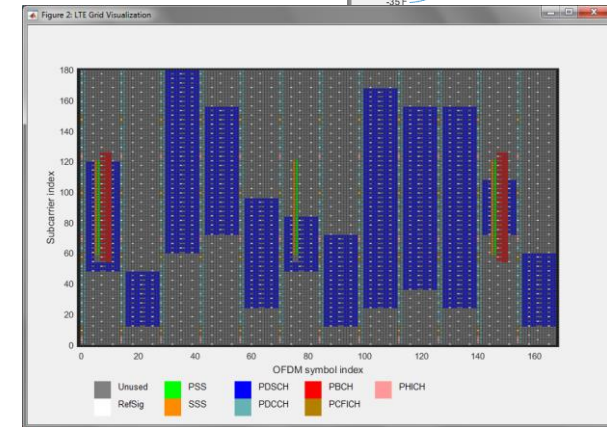
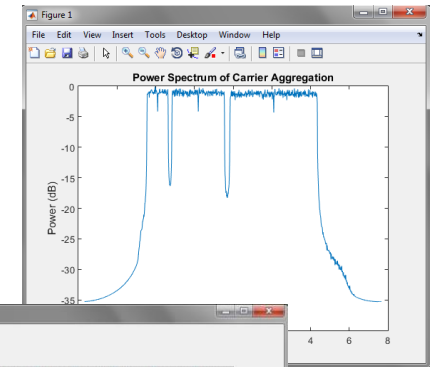
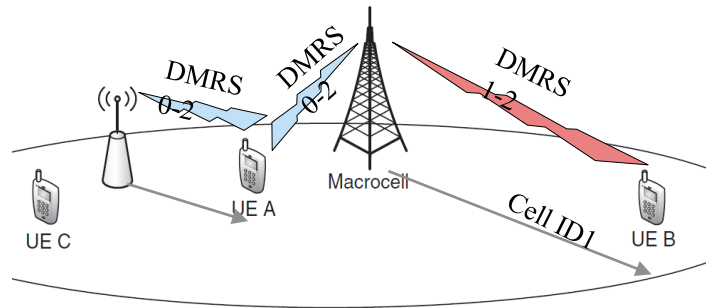
### HW & Radio Connectivity





# LTE System Toolbox

- LTE and LTE-Advanced (Rel-8 through **Rel-12**)
- Scope
  - FDD/TDD
  - Uplink/Downlink
  - Transmitter/Receiver
- ~200 functions for physical layer (PHY) modeling
- Signal generation for LTE & UMTS
- ACLR/EVM measurement
- Conformance Tests



# LTE System Toolbox | *More information...*

- Consult LTE Product Page
  - [www.mathworks.com/products/lte-system/](http://www.mathworks.com/products/lte-system/)
  - Provides overview of LTE/LTE-A capabilities
  - Organized based on use-cases
  
- Consult Wireless Communications Page
  - [www.mathworks.com/wireless](http://www.mathworks.com/wireless)
  - Provides overview of today's MATLAB® for Wireless System Design
  
- For details: Attend Recorded Webinar:
  - “Introducing LTE System Toolbox”

MathWorks | Accelerating the pace of engineering and science

United States | Contact Us | How To Buy | Search MathWorks | Create Account | Log In

Products Solutions Academia Support Community Events Company

## LTE System Toolbox

Simulate, analyze and test the physical layer of LTE and LTE-Advanced wireless communications systems.

Overview Features Code Examples Videos Webinars Related Products What's New Product Trial

LTE System Toolbox™ provides standard-compliant functions and apps for the design, simulation, and verification of LTE and LTE-Advanced communications systems. The system toolbox accelerates LTE algorithm and physical layer (PHY) development, supports golden reference verification and conformance testing, and enables test waveform generation. With the system toolbox, you can configure, simulate, measure, and analyze end-to-end communications links. You can also create and reuse a conformance test bench to verify that your designs, prototypes, and implementations comply with the LTE standard.

- ▶ Key Features
- ▶ Design Verification
- ▶ End-to-End Simulation
- ▶ Signal Generation and Analysis
- ▶ Signal Information Recovery
- ▶ Conformance Testing

Product Overview

Documentation fx Functions Technical Articles Videos Webinars Examples

Learn to Generate LTE Waveforms and Build an End-to-End LTE Model  
» View course info

Try LTE System Toolbox  
» Get trial software

**TRY OR BUY**  
Contact Sales  
Product Trial  
Pricing and Licensing

**What's New**  
From Houman Zarrinkoub, LTE System Toolbox Technical Expert  
Introducing LTE System Toolbox  
» Email Houman

**Technical Resources**  
Support  
Technical Articles  
System Requirements  
**User Community**  
Answers  
File Exchange  
Link Exchange

# WLAN System Toolbox

- Physical layer (PHY) modeling

Standard-compliant functions for the **design, simulation, analysis**, and testing of wireless LAN communications systems

- Transmitter & Receiver

L-SIG, HT-SIG, VHT-SIG-A, VHT-SIG-B  
OFDM, MIMO Equalization, STBC Combining  
Packet detection, symbol timing correction  
Coarse and fine frequency offset estimation  
Preamble signal decoders for L-SIG, HT-SIG, VHT-SIG-A, VHT-SIG-B fields

- Propagation Channel

– TGn, TGac



- Measurements

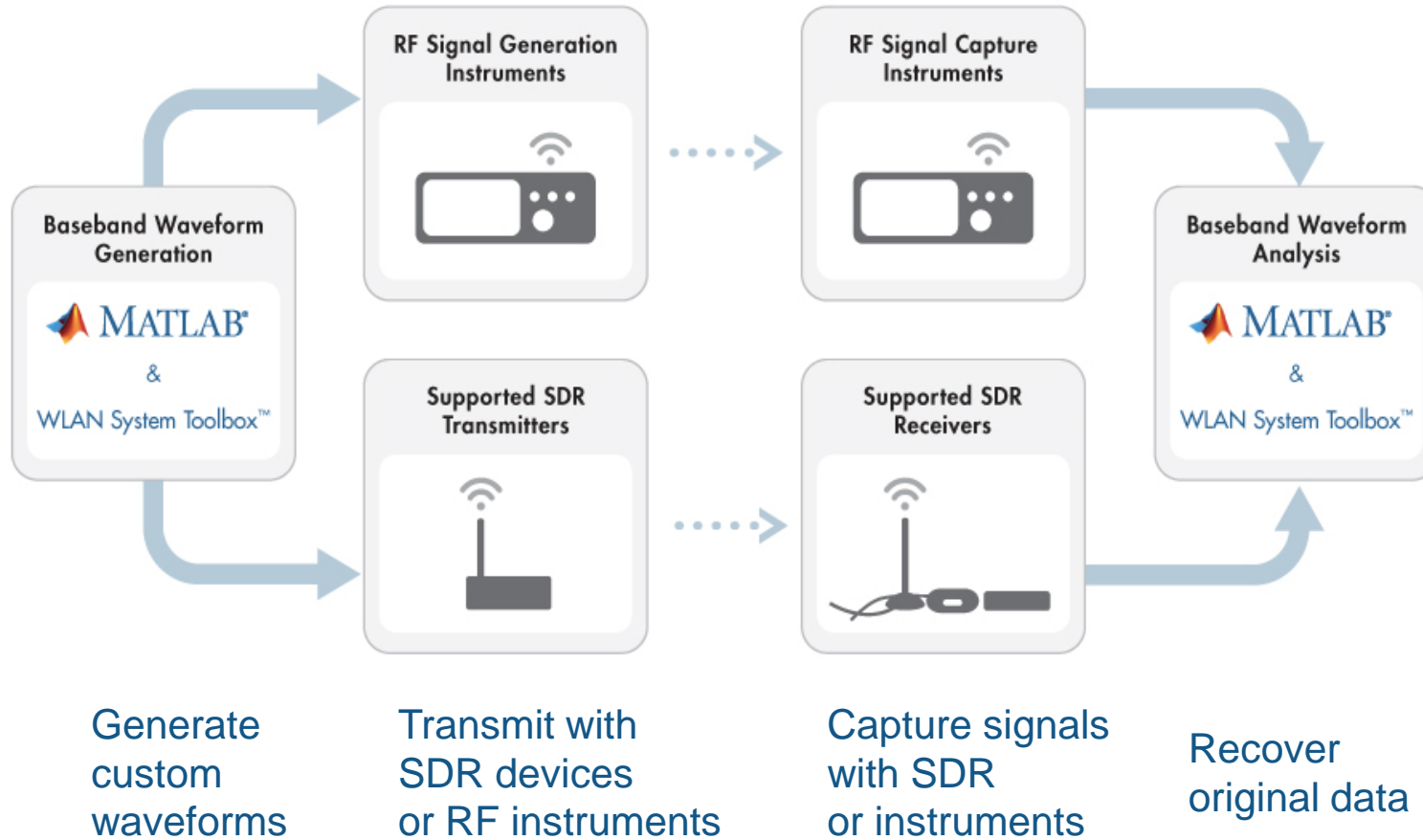
– Packet Error Rate, EVM, Spectral Emissions

- Features

– Open, customizable MATLAB code  
– C-code generation with MATLAB Coder

# WLAN System Toolbox

## Hardware & Radio Connectivity



**Range of supported hardware**

-   
RF Signal Generator
-   
Spectrum Analyzer
-   
Zynq Radio SDR
-   
USRP SDR

# WLAN System Toolbox | *More information...*

- Consult WLAN Product Page
  - [www.mathworks.com/products/wlan-system/](http://www.mathworks.com/products/wlan-system/)
  - Provides overview of WLAN capabilities
  - Organized based on use-cases
  
- Consult Wireless Communications Page
  - [www.mathworks.com/wireless](http://www.mathworks.com/wireless)
  - Provides overview of today's MATLAB® for Wireless System Design
  
- For details: Attend Recorded Webinar:
  - “Introducing WLAN System Toolbox”

**WLAN System Toolbox** NEW PRODUCT

Simulate, analyze, and test the physical layer of WLAN communications systems

[Overview](#) [Features](#) [Code Examples](#) [Related Products](#) [Product Trial](#)

WLAN System Toolbox™ provides standard-compliant functions for the design, simulation, analysis, and testing of wireless LAN communications systems. The system toolbox provides configurable physical layer waveforms for IEEE 802.11ac and 802.11n/g/a/b standards. It also provides transmitter, channel modeling, and receiver operations, including channel coding, modulation (OFDM, DSSS, and CCK), spatial stream mapping, channel models (TGac and TGn), and MIMO receivers.

You can generate very-high-throughput (VHT), high-throughput (HT-mixed), and legacy (non-HT) signals, and demodulate VHT, HT-mixed, and non-HT OFDM signals. You can also perform measurements such as channel power, spectrum mask, and occupied bandwidth, and create test benches for the end-to-end simulation of WLAN communications links.

The system toolbox provides reference designs to help you explore baseband specifications and study the effects of RF designs and interference sources on system performance. Using WLAN System Toolbox with hardware support packages, you can connect your transmitter and receiver models to radio devices and verify your designs via over-the-air transmission and reception.

- ▶ Key Features
- ▶ Signal Generation
- ▶ Signal Measurement
- ▶ Link-Level Simulation and Throughput Analysis
- ▶ Radio Connectivity
- ▶ Open, Customizable Algorithms

[Documentation](#) [fx Functions](#) [Examples](#) [Hardware Support](#)

**TRY OR BUY**

[Contact Sales](#)  
[Product Trial](#)  
[Pricing and Licensing](#)

**What's New**

From Houman Zarrinkout  
 System Toolbox Technica  
[See videos and webinars](#)  
[» Email Houman](#)

**Technical Resources**

[Support](#)  
[Technical Articles](#)  
[System Requirements](#)

**User Community**

[Answers](#)  
[File Exchange](#)  
[Link Exchange](#)

*Signal Processing*

*Audio*

*Antenna to Bits*

*WLAN/LTE*

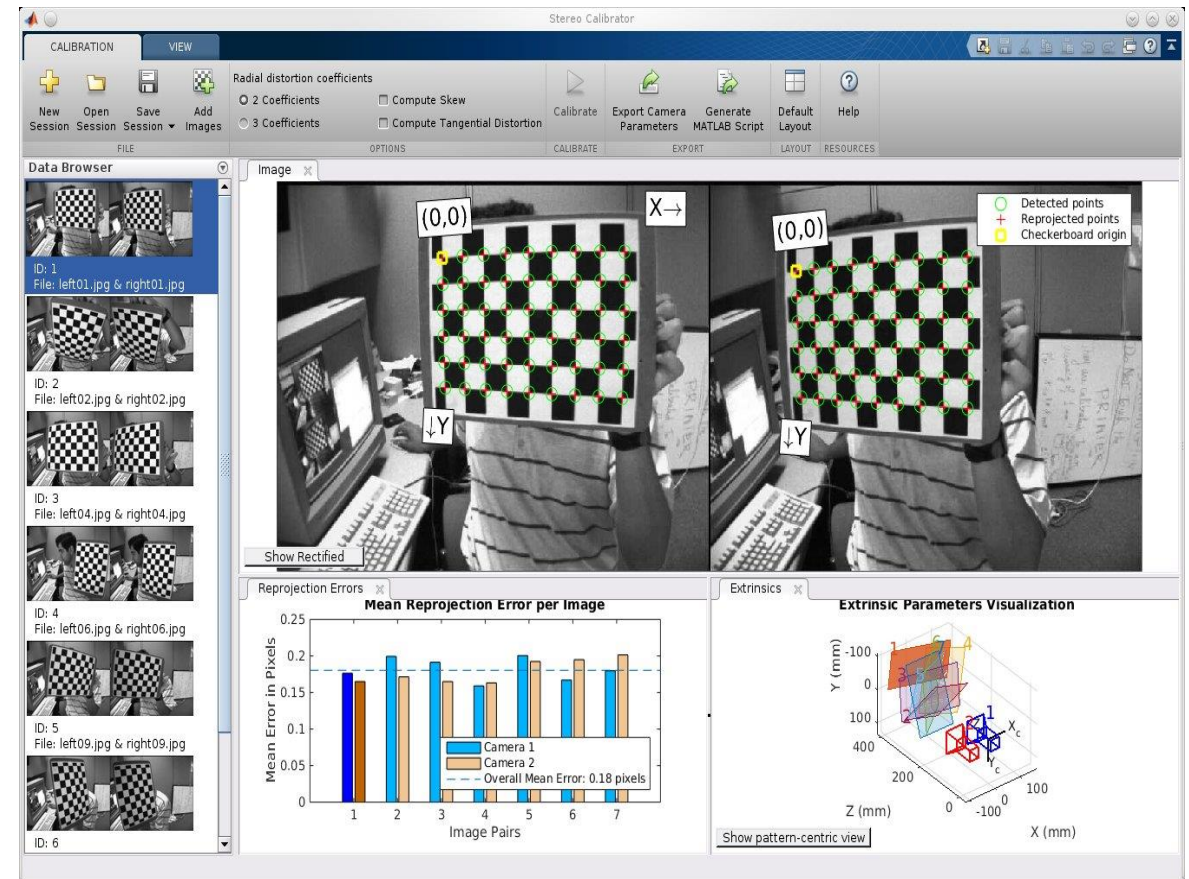


*Image and Video Processing*



# Image and Video Processing

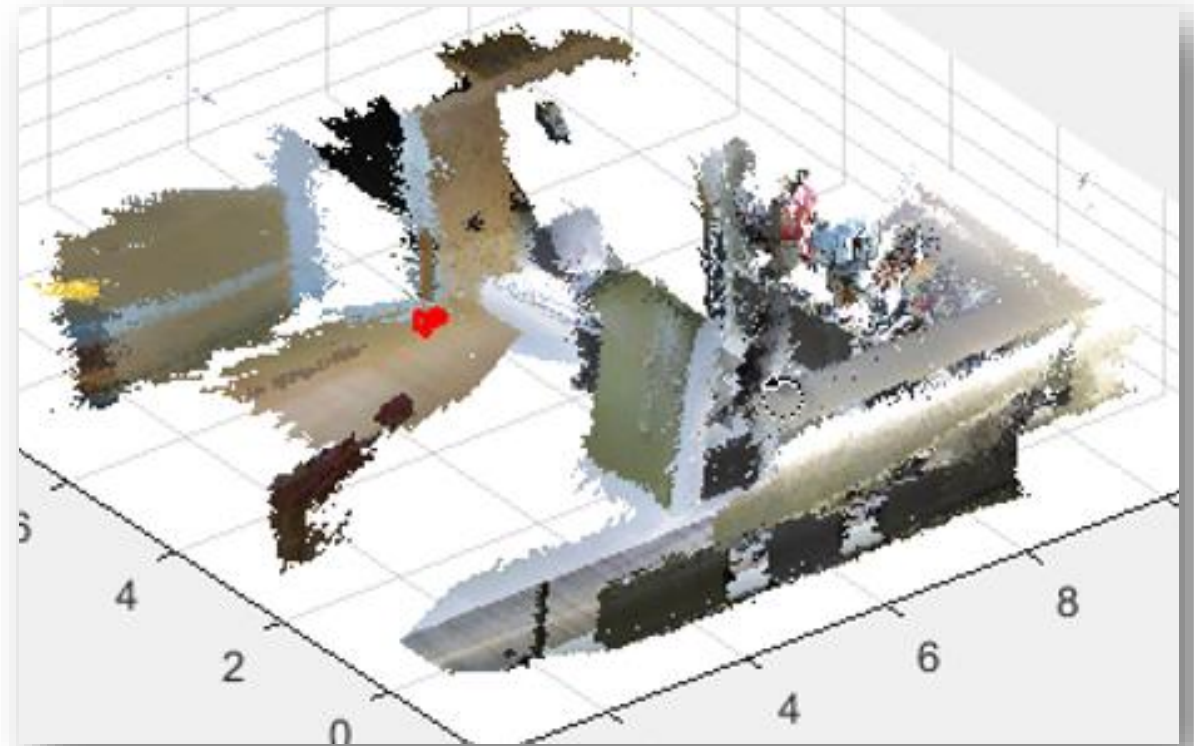
- Stereo Camera Calibration **R2014b**
  - Lens distortion correction
  - Rectification
- Depth estimation **R2014a**
- 3D Scene reconstruction **R2014a**
- Code generation **R2015a**



# Image and Video Processing | *Stereo Vision*

R2016a

- Enables autonomous systems to **map and measure** the world
- Supports workflows for **ADAS**, autonomous driving, and robotics
- New functionality to support:
  - 3D **point cloud** processing
  - Structure from motion

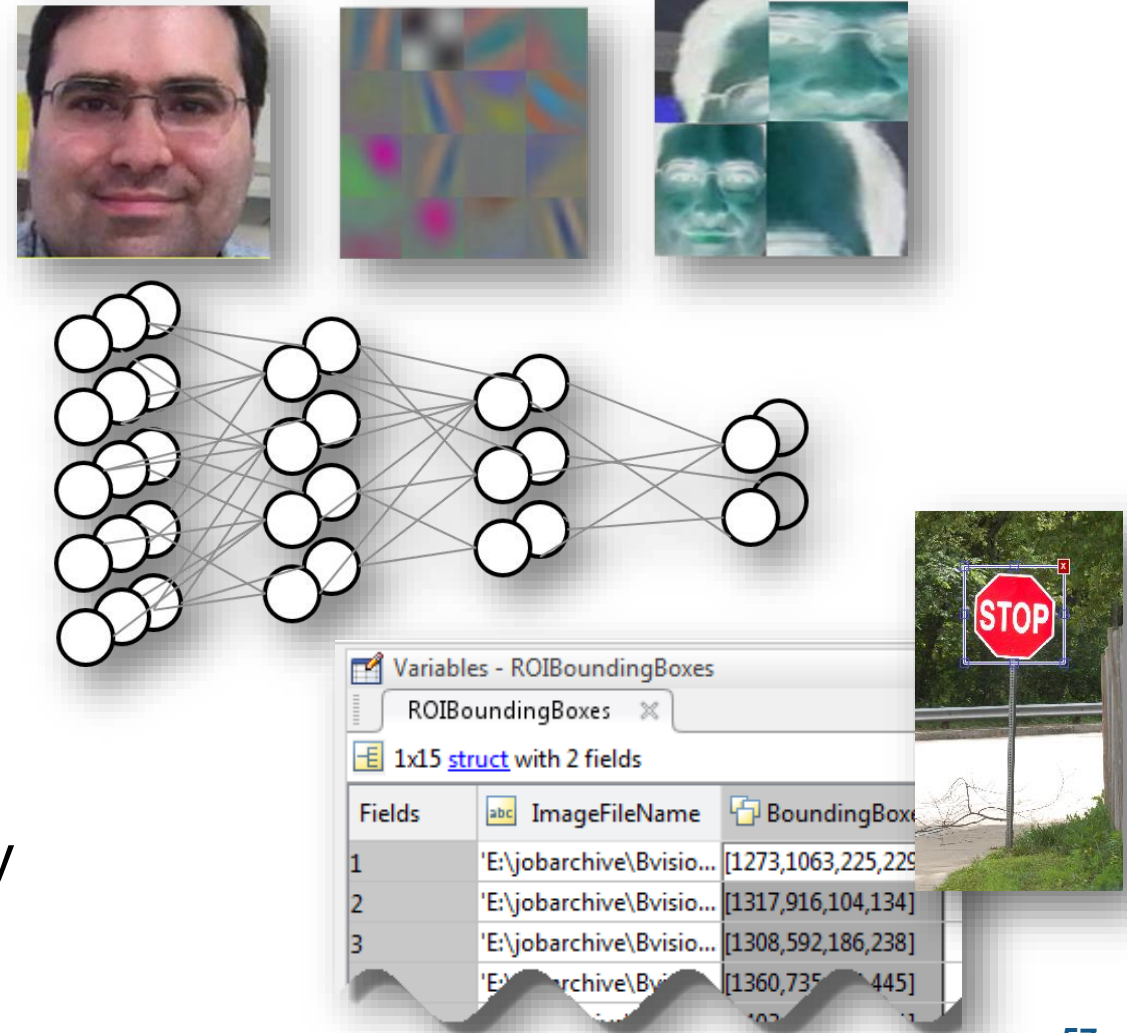




# Image and Video Processing | *Deep Learning*

R2016a

- Perform fast, accurate image classification
- Enables recognition workflows in autonomous robotics and ADAS
- Convolutional neural network (CNN) algorithm added to Neural Network Toolbox
- Uses cuDNN (a GPU-accelerated library from NVIDIA)  
(requires Parallel Computing Toolbox)





*Signal Processing*



*Audio*



*Antenna to Bits*



*WLAN/LTE*



*Image and Video Processing*

**That's, what's new!**