MATLAB EXPO

May 28, 2024 | Beijing

使用 MATLAB 和 Simulink 开发音频系统

马朝辉, MathWorks

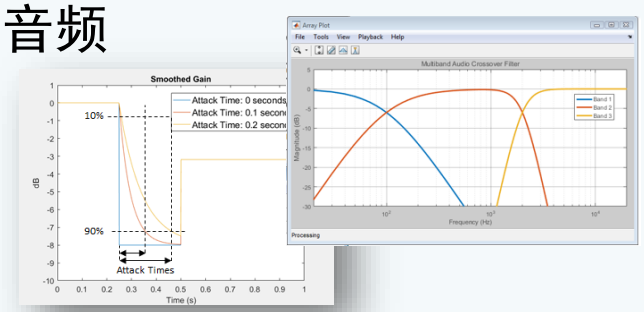
MathWorks

提纲

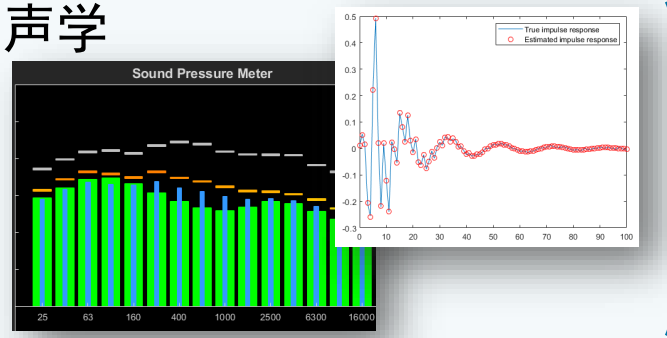
- Audio Toolbox
- 使用 MATLAB 进行音频算法开发
- 使用 Simulink 进行音频系统开发
  - 基于模型的音频设计
  - 智能音箱 Demo

Audio Toolbox

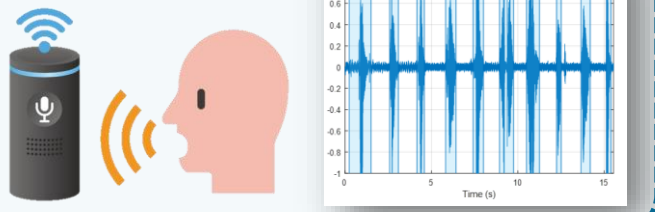
音频



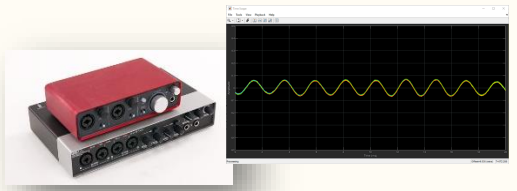
声学



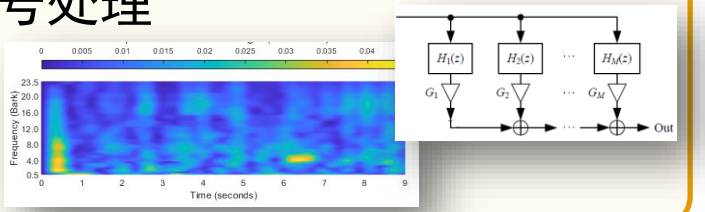
语音



音频 I/O



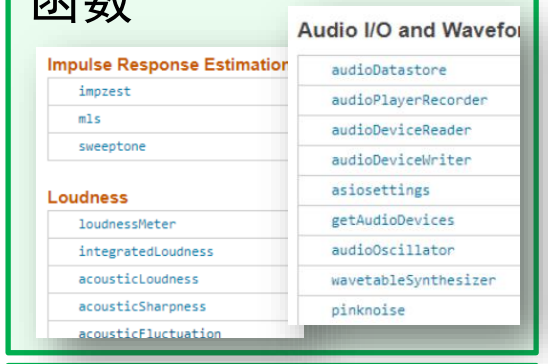
信号处理



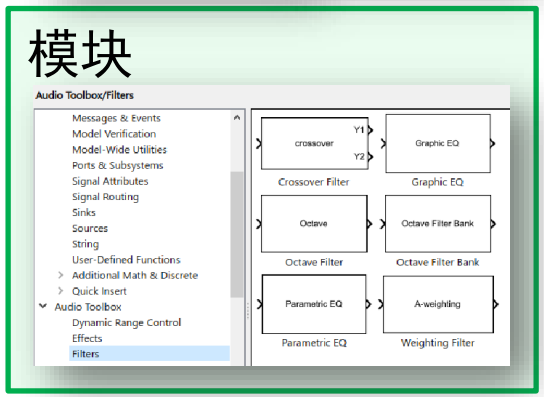
AI

音频插件

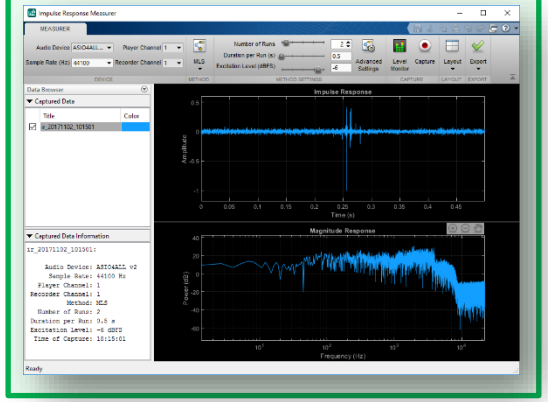
函数



模块



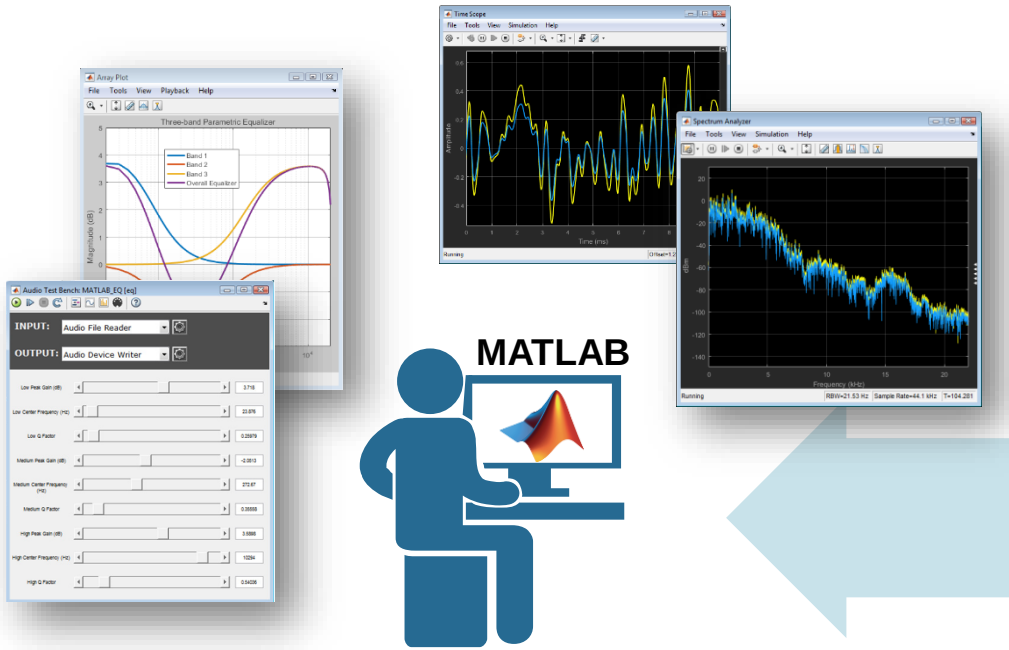
应用



使用 MATLAB 进行音频算法开发

音频 I/O

与标准音频驱动程序之间的连接

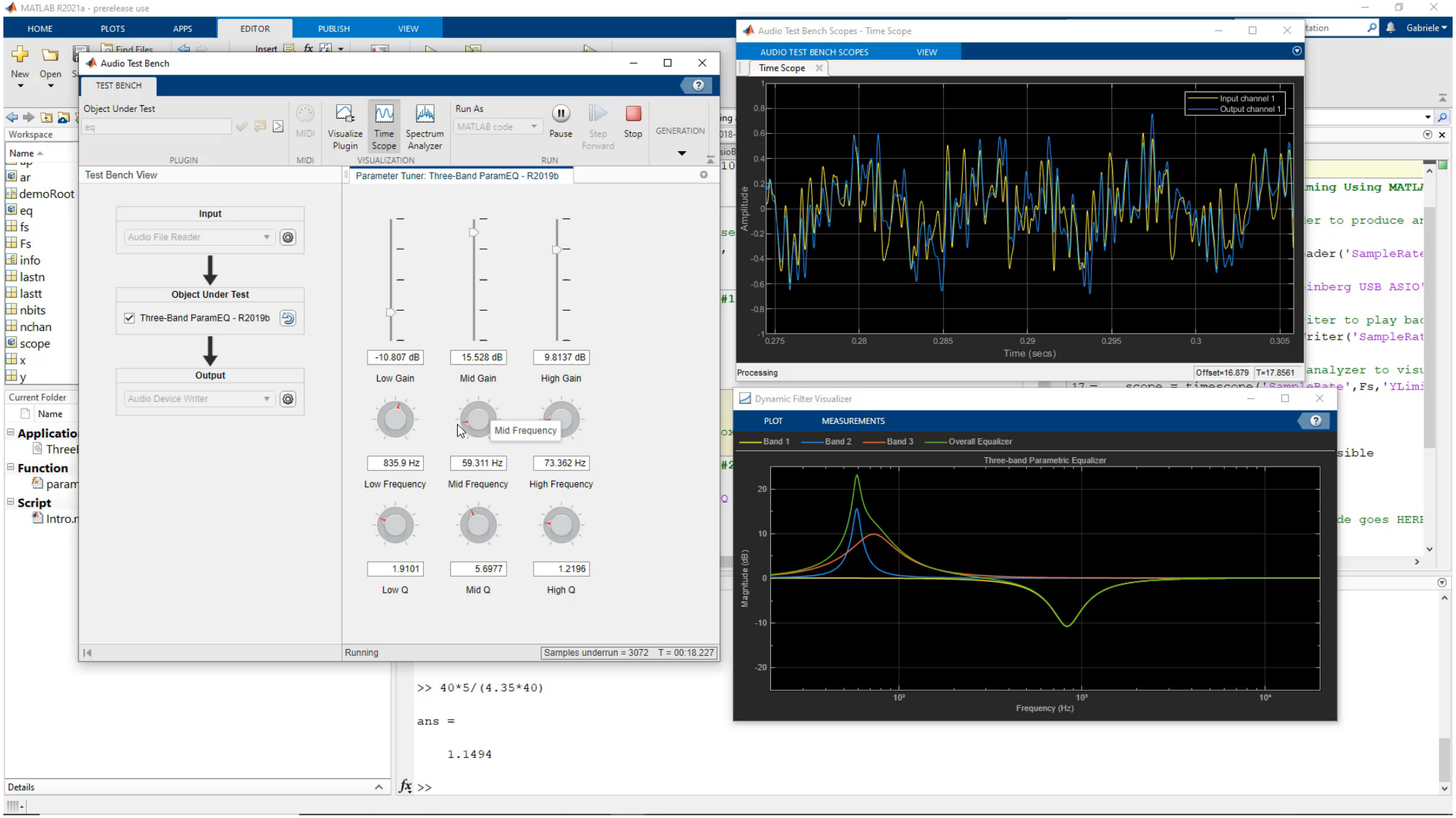


ASIO
Core Audio
ALSA
DirectSound
WASAPI

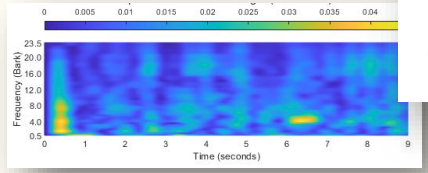
I/O 音频流

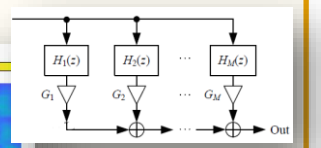
MIDI

Audio Test Bench App 中的实时音频 I/O



信号处理


0
0.005
0.01
0.015
0.02
0.025
0.03
0.035
0.04
23.5
20.0
16.0
12.0
8.0
4.0
0.5
Frequency (Bark)
0
1
2
3
4
5
6
7
8
9
Time (seconds)


$H_1(z)$
$H_2(z)$
$H_M(z)$
G_1
G_2
G_M
Out

预处理和特征提取

特征提取

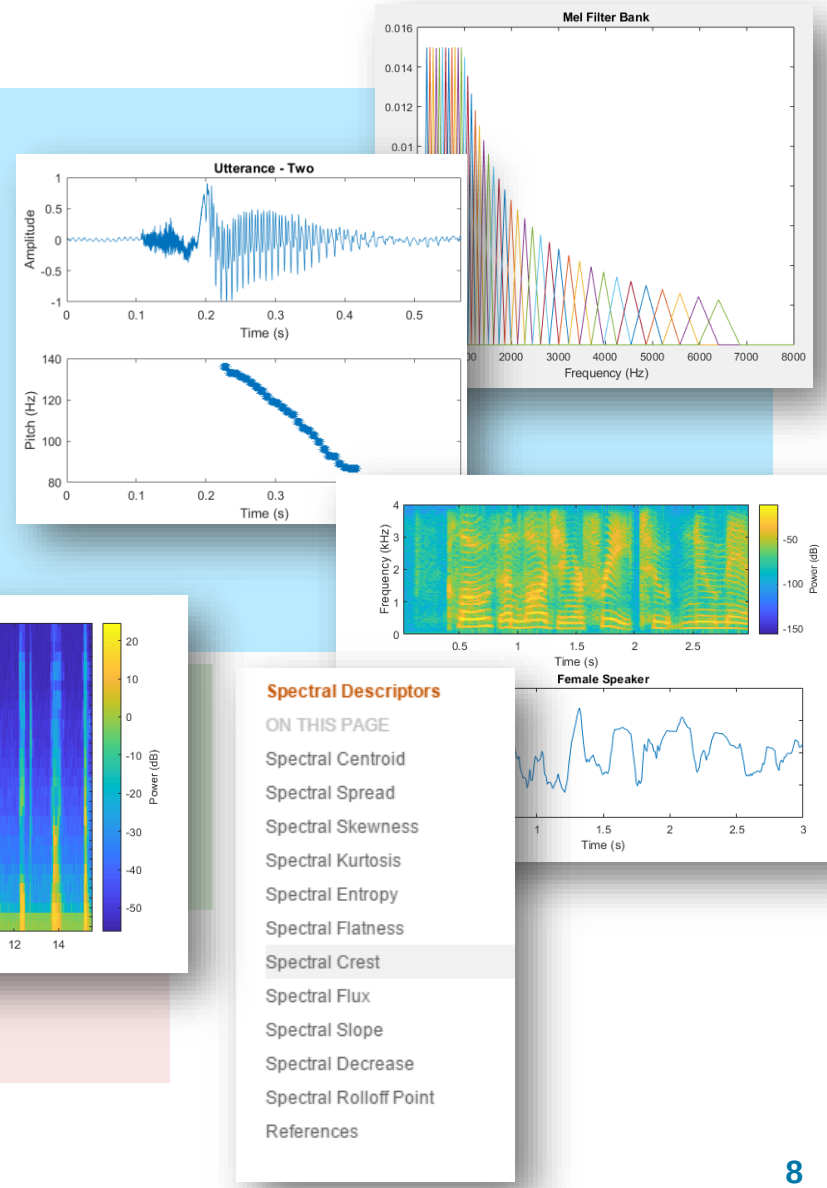
- MFCC – Mel Frequency Cepstral Coefficients
- GTCC – Gammatone Cepstral Coefficients
- Pitch and Harmonicity
- Spectral Descriptors

变换

- Mel-Spaced Spectrogram
- Gammatone and Octave Filter Banks

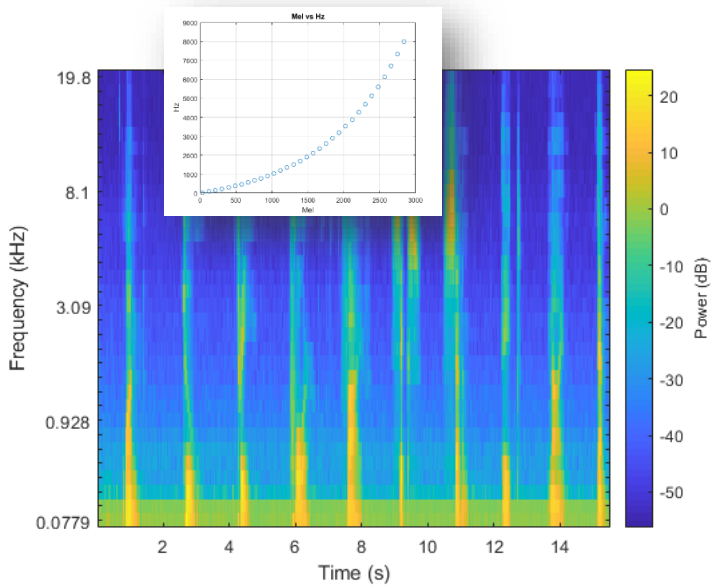
分割

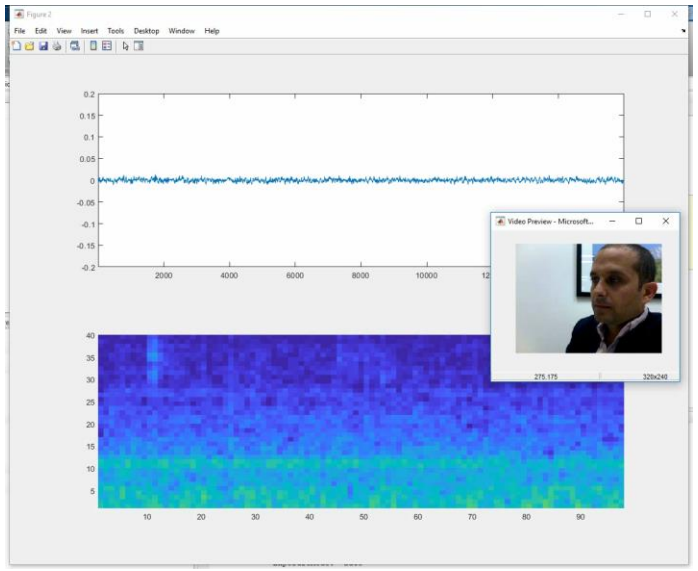
- Voice Activity Detection



时频变换 – Mel-Spaced Spectrogram

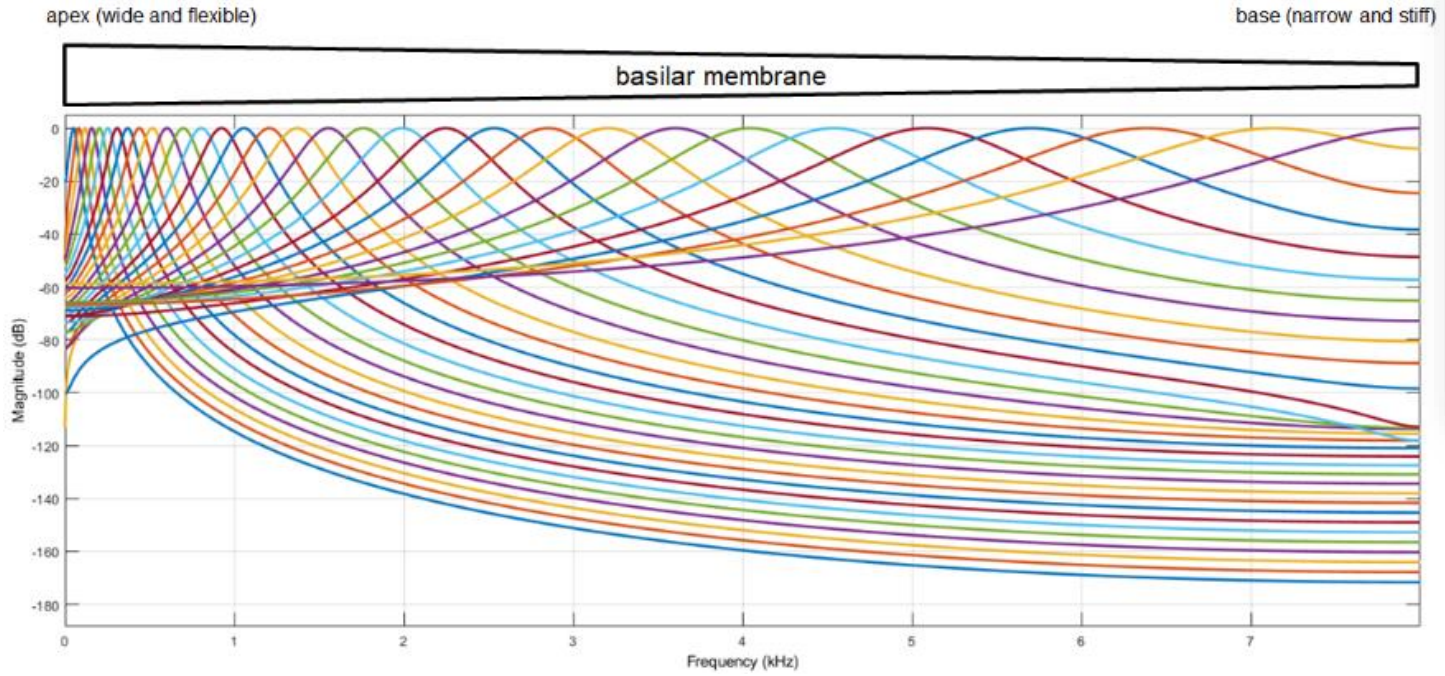
- **`melSpectrogram`** 函数
- 随时间变化的感知缩放频谱：
  - 人耳最敏感的地方分辨率更高
  - 将音频和语音转换为2D的紧凑方法
- 基于FFT，计算效率高
- 将音频、语音和声学信号作为CNN的输入
- Featured in “语音命令识别”, and “声音场景识别" examples

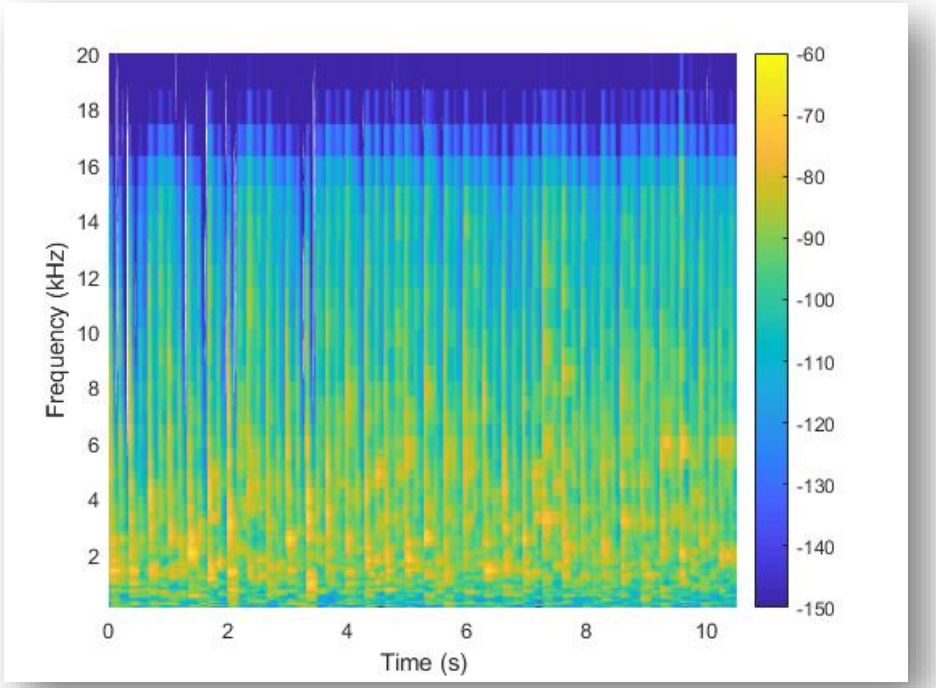




时频变换 – Filter Banks

- Gammatone Filter Bank (**`gammatoneFilterBank`**)
  - Perceptually-spaced (ERB) filters
  - Optionally use to create ERB-spaced spectrogram

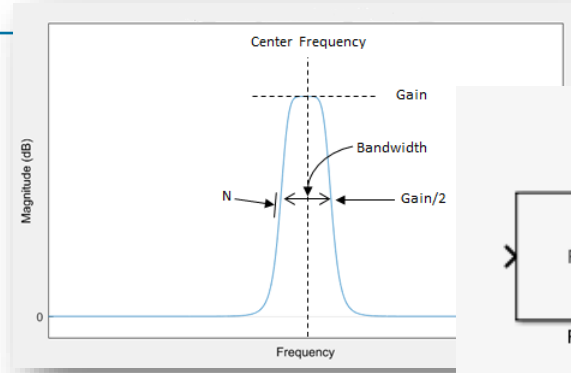


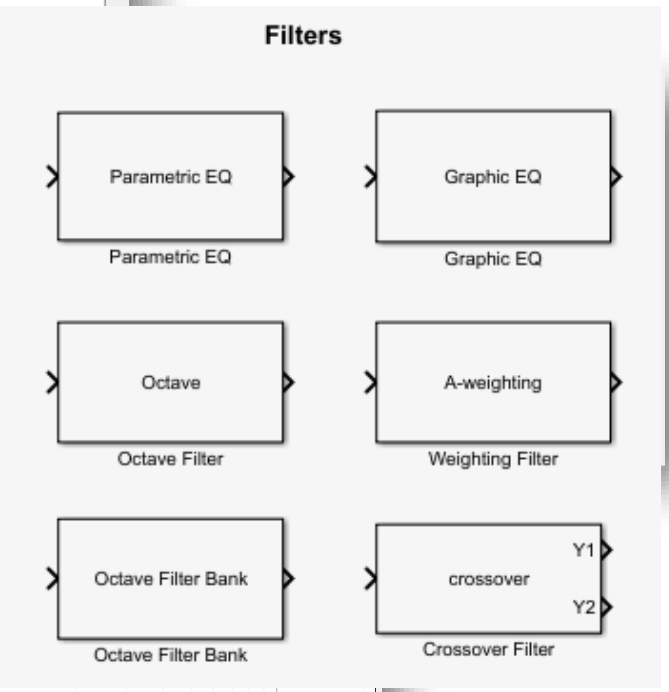


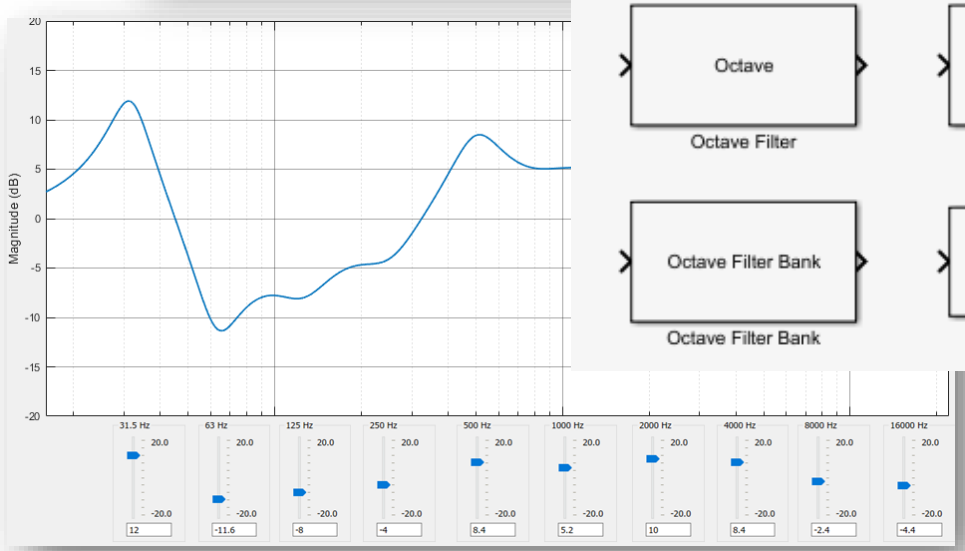
音频滤波和均衡

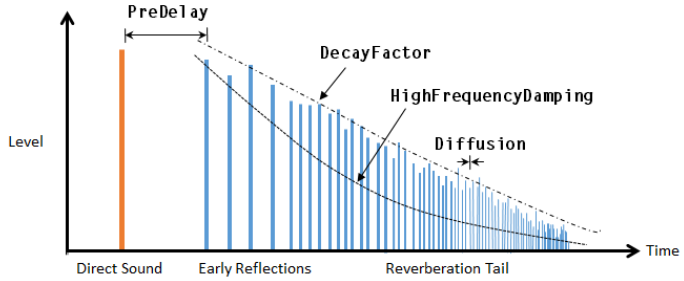
- 多频段参数 EQ
- 图形化 EQ
- Octave 滤波器组
- Crossover
- 动态范围控制
- 混响
- *And much more!*

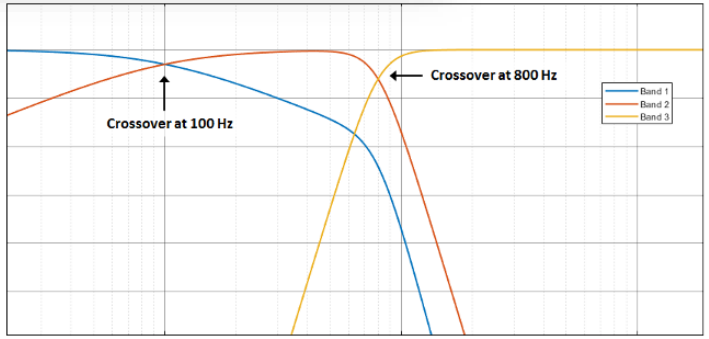
- MATLAB 和 Simulink 支持





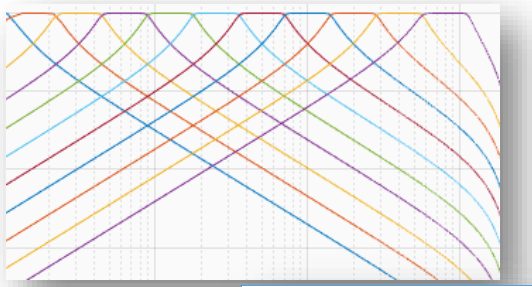


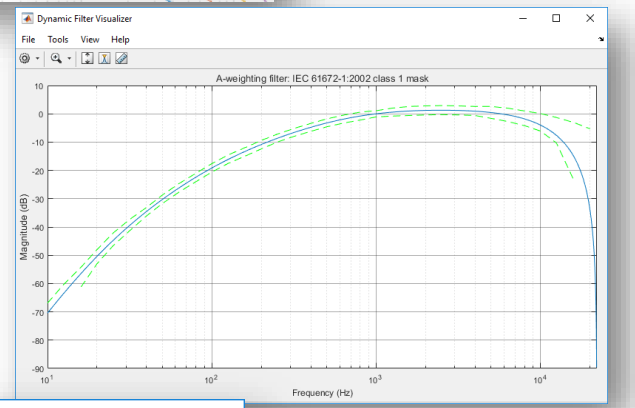


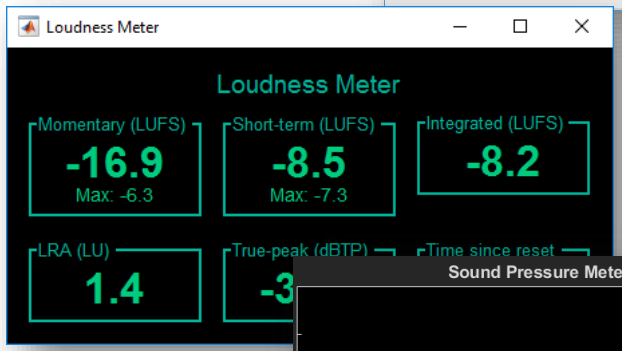


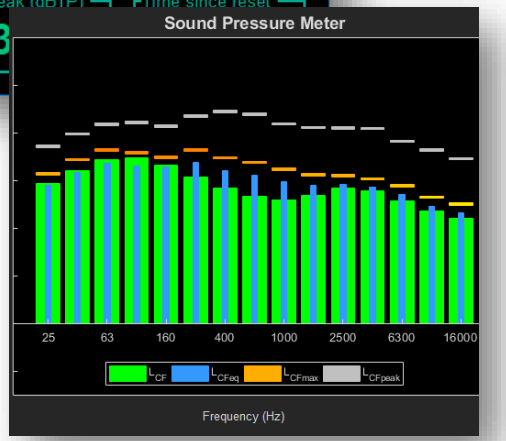
声学测量

- 倍频程滤波器 (整数和小数)
  - ANSI S1.11-2004 标准
  - 可变中心频率、整数和小数频带宽度 (最小到 1/48 倍频程)
- A-, C-, 和 K加权滤波器
  - ANSI S1.42-2001 (A,C) 或 ITU-R BS.1770-4 (K) 标准
  - 检查是否符合IEC 61672-1:2002
- 响度测量
  - 瞬时响度、短时响度、集成响度、响度范围、真实峰值
  - EBU R 128 和 ITU-R BS.1770-4 标准
- 声压（SPL）测量
  - 频率加权、快或慢时间加权、等效连续、峰值和最大声级
  - 倍频程频带的声压测量
- 心理声学测量
  - 尖锐度
  - 粗糙度
  - 起伏强度



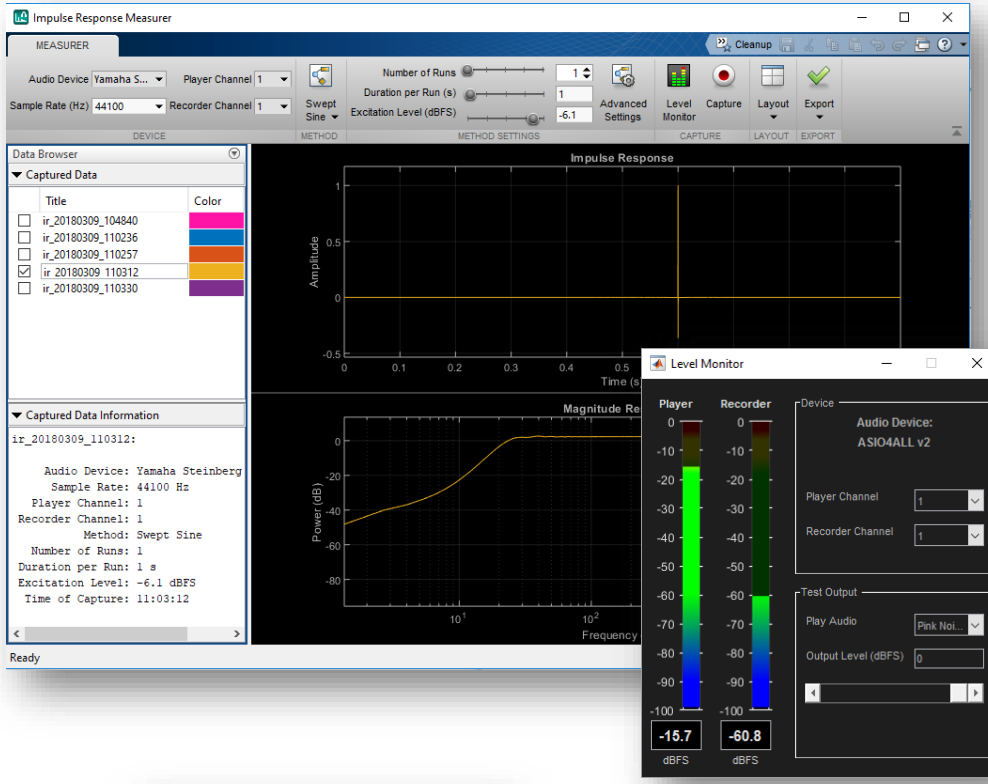


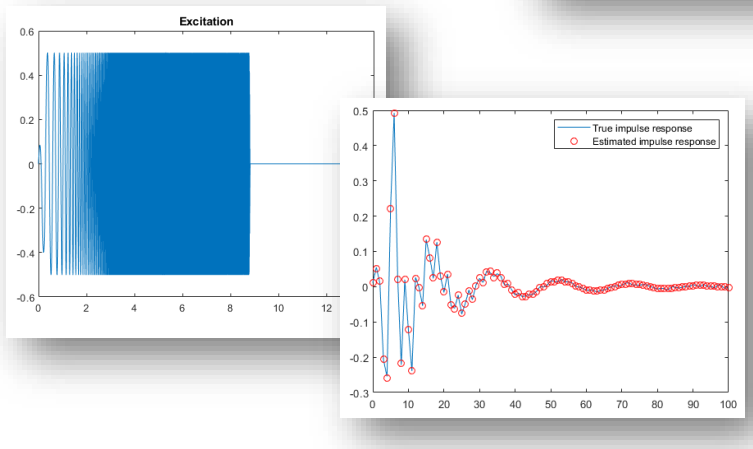




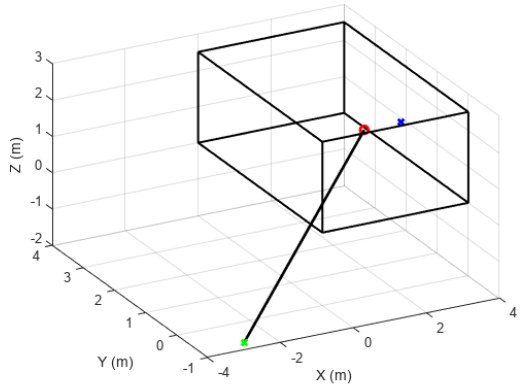
脉冲响应测量

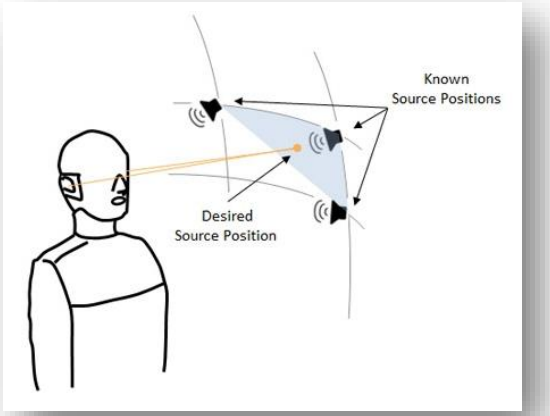
- 测量电子和声学系统的脉冲响应和频率响应：
  - Maximum-Length Sequences (MLS)
  - Exponentially Swept Sinusoids (ESS)
- 内置 Impulse Response Measurer App
  - 响应可视化和信号电平测量
  - 导出响应到工作空间，Signal Analyzer, Filter Visualizer
- 编程工作流
  - 激励序列生成 (`mls`, `sweeptone`)
  - 脉冲响应估计 (`impzest`)



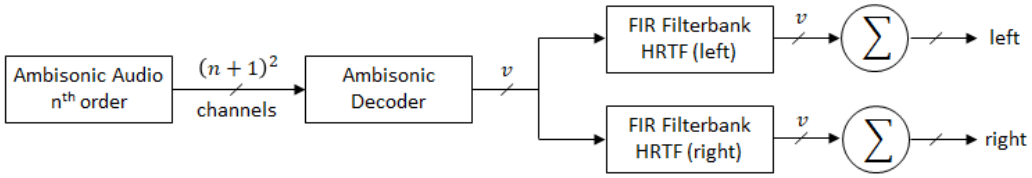


空间音频

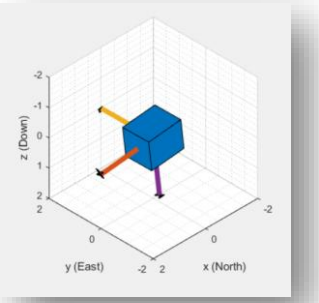




- SOFA
  - Read, Analyze and Process SOFA Files
- 房间脉冲响应
- 头传
- 示例:
  - Ambisonic Binaural Decoding
  - Binaural Audio Rendering Using Head Tracking
    - Arduino Uno + Invensense MPU-9250
  - Room Impulse Response Simulation with Stochastic Ray Tracing
  - Room Impulse Response Simulation with the Image-Source Method and HRTF Interpolation



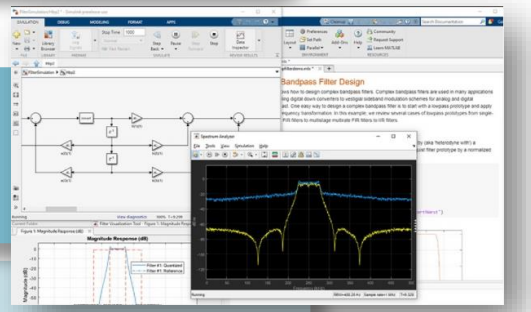
v: Number of virtual loudspeakers



信号处理工具生态

Audio Toolbox

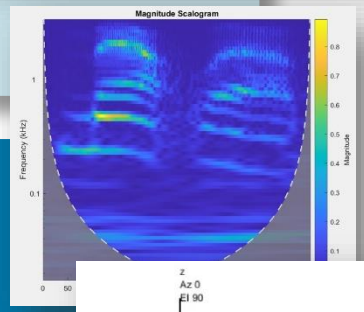
- 音频、声学和语音
- 特定应用的 AI



DSP System Toolbox

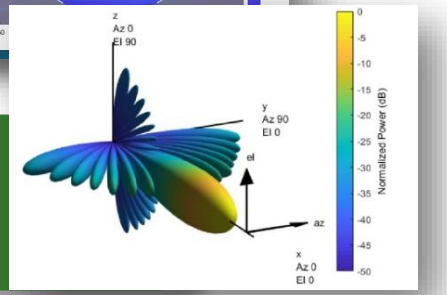
Signal Processing Toolbox

- 交叉应用信号处理
- 滤波器设计和频谱分析
- 信号分析与系统设计



Wavelet Toolbox

- 小波和高级时频分析



Phased Array System Toolbox

- 阵列信号处理

AI


Inside, small room
Breathing
Dog
Domestic animals, pets
Sound effect
Burst, pop
Gurgling
Pour
Cat
Water
Explosion
Snoring
Liquid
Stream
Bark
Silence
Machine gun
Artillery fire
Meow
Trickle, dribble
Animal
Gunshot, gunfire

用于音频、声学和语音的 AI

Access & Create Data

- Importing data
- Generating data from simulations
- Labeling data

数据集管理

自动标注

语音合成

语音增强

Preprocess & Explore Data

- Cleaning data
- Transforming data
- Extracting features

预处理

特征提取

时频变换

Develop Predictive Models

- Model design and tuning
- Hardware accelerated training
- Interoperability

特定于领域的模型和层

Python/MATLAB 的互操作性

Deploy & Share

- Embedded devices
- IT/OT integration
- Edge computing

C 代码生成

GPU 代码生成

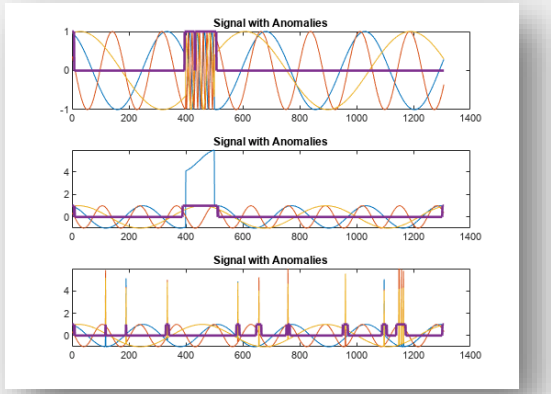
MATLAB 中用于音频信号处理的 AI 模型

面向任务的预置人工智能函数

- `speech2text`
- `text2speech`
- `classifySound`
- `pitchnn`
- `vggishFeatures`
- `openl3Features`
- `detectspeechnn`
- `speakerRecognition`
- `deepSignalAnomalyDetector`

```
transcriber = speechClient("wav2vec2.0",Segmentation="none");
txt = speech2text(transcriber,audioIn,fs)
```

```
txt = "every day we rely on a wide range of machines but the truth is that
```

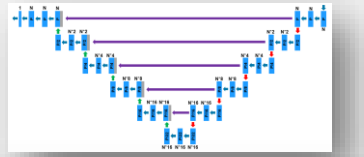


直接可用的 AI 模型

- Sound type classification (YAMNet)
- Speech recognition (wav2vec, DeepSpeech)
- Audio embeddings (VGGish, OpenL3)
- Musical pitch estimation (CREPE)
- Speaker verification (i-Vectors, x-Vectors)
- Signal Anomaly Detection (CNN, LSTM)

预先设计好的架构和示例

- Baseline examples with **standard layers**
  (1D CNN, 2D CNN, LSTM, GRU, Fully-connected,…)
- Signal-specific architectures:
  (SincNet,1D U-NET, …)
- Differentiable DSP Layers
  (STFT, CWT, DWT)



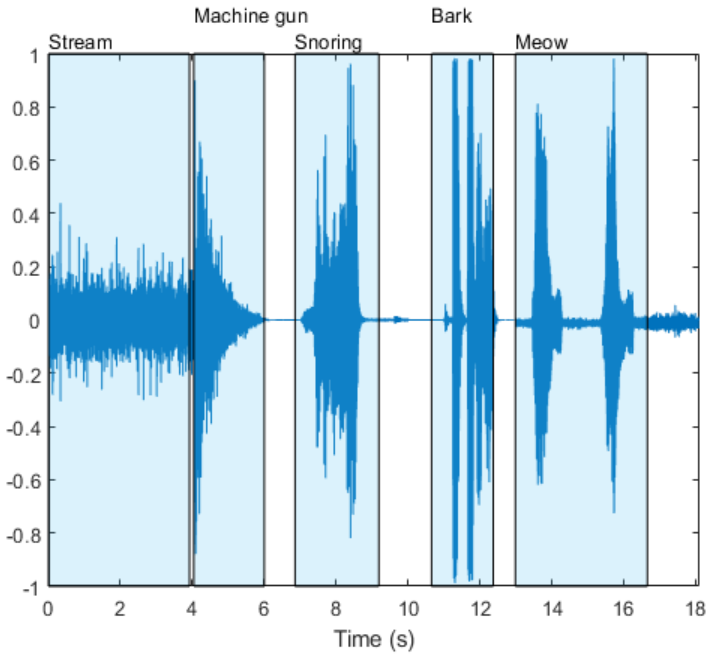
`stftLayer` (Signal Processing Toolbox)

`cwtLayer` (Wavelet Toolbox)

`modwtLayer` (Wavelet Toolbox)

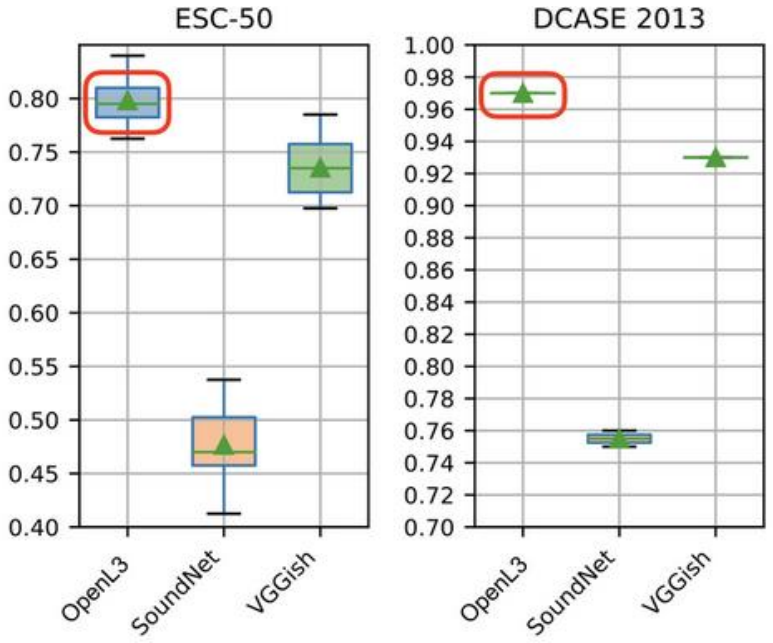
用于解决实际问题的 AI 模型

声音分类



classifySound
YAMNet

深度特征提取



vggish
openL3

语音转录与合成

speech2text

version 1.2.8 (204 KB) by MathWorks Audio Toolbox Team STAFF

Automatic speech-to-text conversion

10 Ratings
91 Downloads
Updated 16 Sep 2020
View Version History
View License

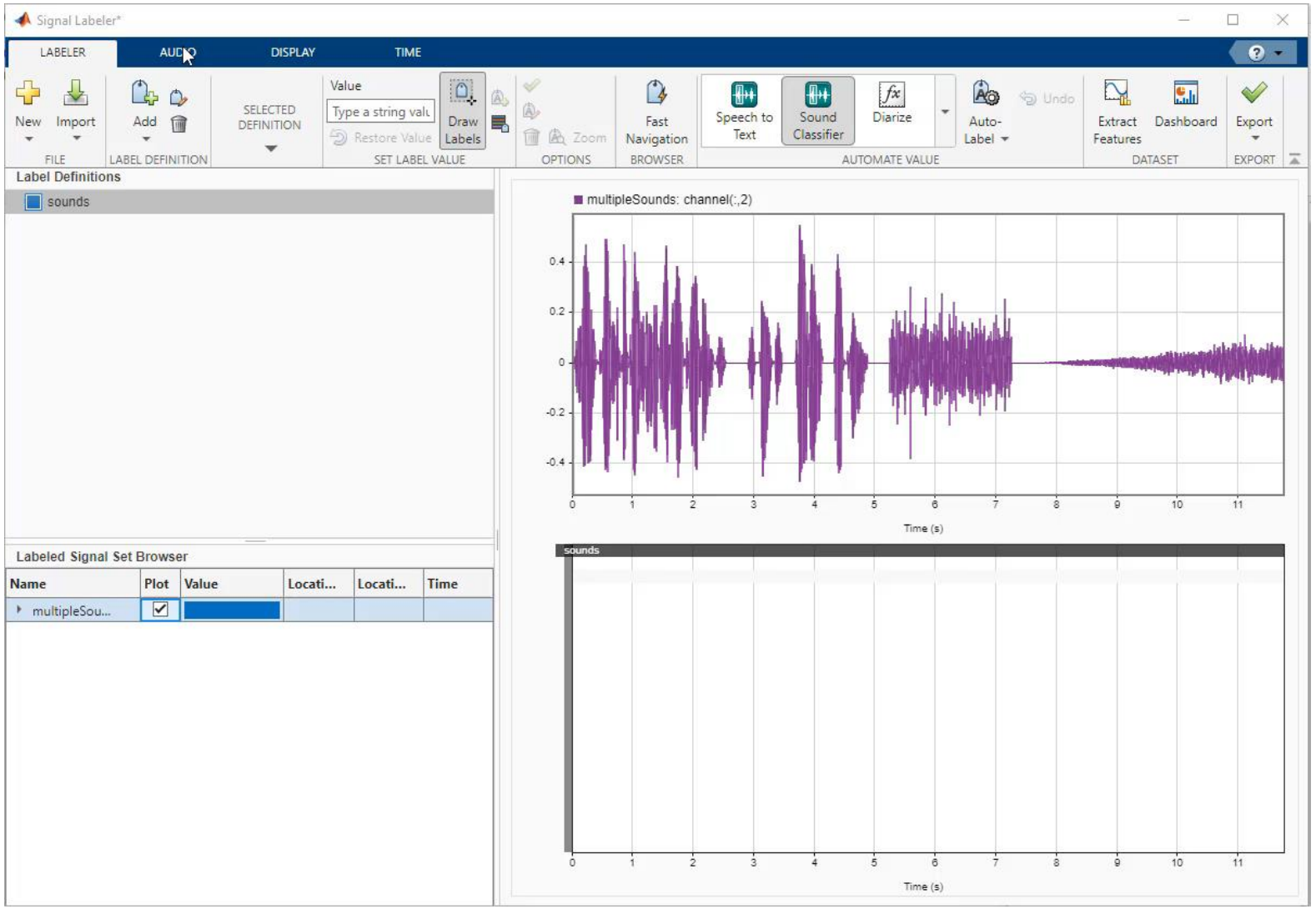
text2speech

version 1.0.1 (69.6 KB) by MathWorks Audio Toolbox Team STAFF

Automatic text-to-speech synthesis

3 Ratings
29 Downloads
Updated 27 Feb 2020
View Version History
View License

自动标注：声音分类


Signal Labeler*
LABELER
AUDIO
DISPLAY
TIME
New
Import
FILE
Add
LABEL DEFINITION
SELECTED DEFINITION
Value
Type a string valu
Restore Value
Draw Labels
SET LABEL VALUE
Zoom
OPTIONS
Fast Navigation
BROWSER
Speech to Text
Sound Classifier
Diarize
AUTOMATE VALUE
Auto-Label
Undo
Extract Features
Dashboard
DATASET
Export
EXPORT
Label Definitions
sounds
Labeled Signal Set Browser
Name
Plot
Value
Locati...
Locati...
Time
multipleSou...
multipleSounds: channel(:,2)
0.4
0.2
0
-0.2
-0.4
0
1
2
3
4
5
6
7
8
9
10
11
Time (s)
sounds
Time (s)

一行指令即得 AI：声源分离

```
[audioIn,fs] = audioread("ThreeSpeakers.flac");
play(audioIn,fs)
```

Separate the three speakers.

```
y = separateSpeakers(gpuArray(audioIn),fs,NumSpeakers=3);
```

Listen to the separated speakers.

```
play(y(:,1),fs)
```

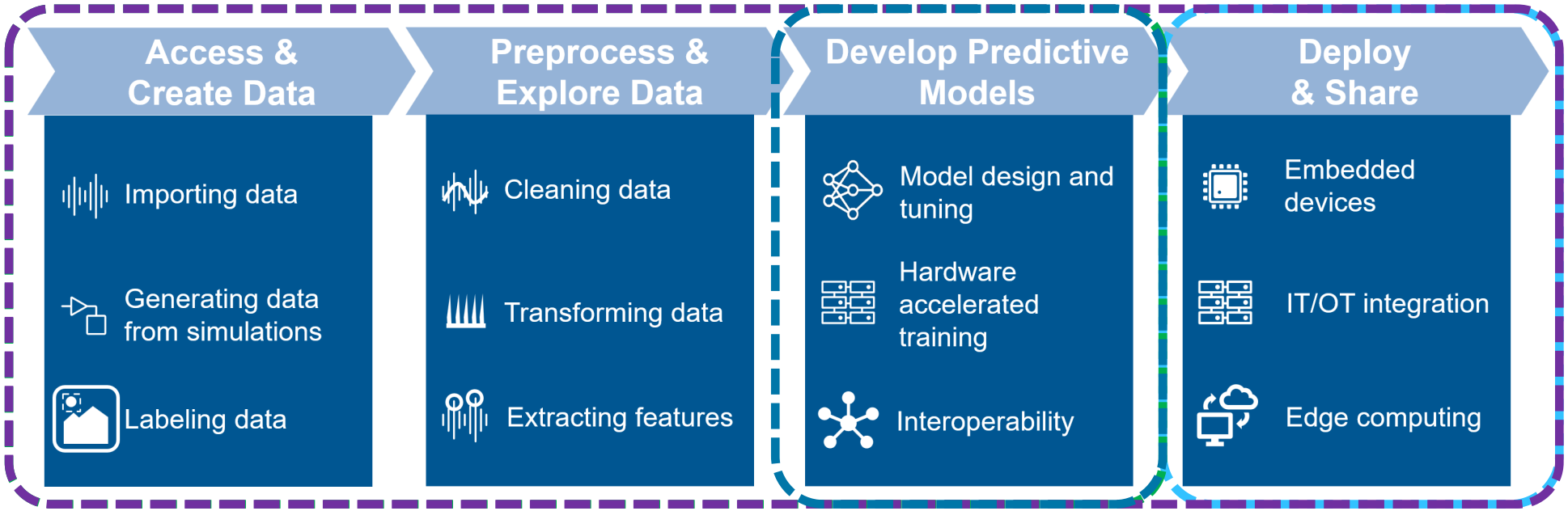
```
play(y(:,2),fs)
```

```
play(y(:,3),fs)
```

Call `separateSpeakers` again. Number of speakers is now unknown.

```
separateSpeakers(gpuArray(audioIn),fs);
```

AI 工具生态 – 机器学习和深度学习



Audio Toolbox

Deep Learning Toolbox

- 深度网络设计，训练和分析

GPU Coder

- CUDA 代码生成
- GPU 实时实现

Statistics and Machine Learning Toolbox

- 少量数据机器学习模型

Parallel Computing Toolbox

- 多核多节点分布式计算
- 基于 GPU 加速

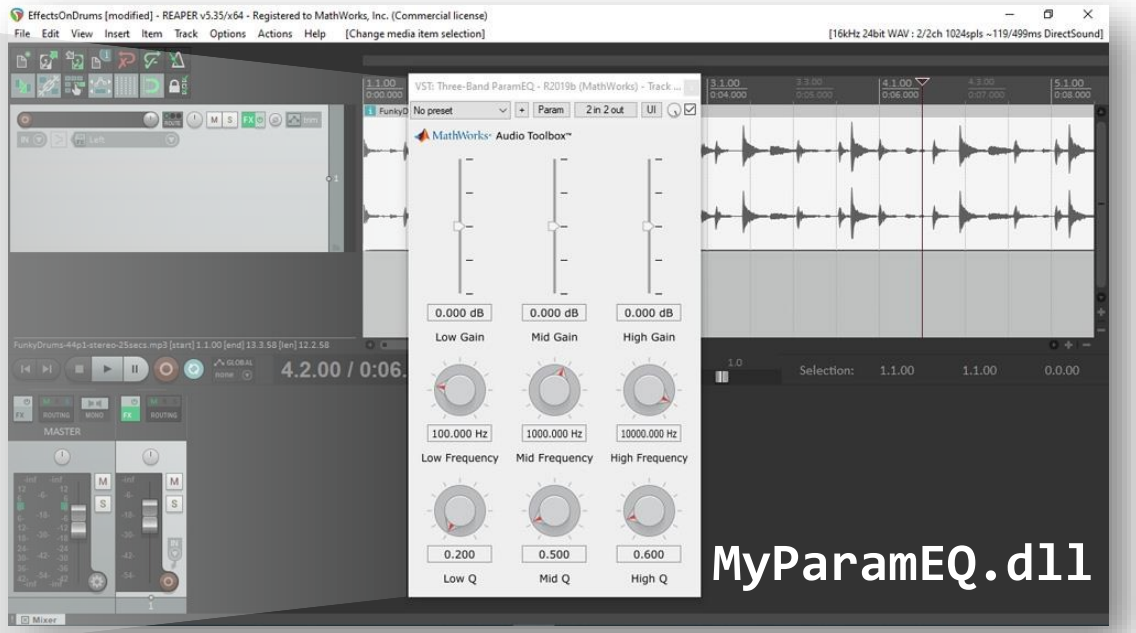
音频插件

在 MATLAB 中设计，分析和仿真

流畅地在您喜欢的 DAW 中运行

MyParamEQ.m

```
classdef MyParamEQ < matlab.System & audioPlugin
%MyParametricEqualizer 3 band parametric equalizer.
    %   This example shows a 3-band parametric equaliz
    %   band provides a center frequency in Hertz, a Q
    %   decibels.
    %
    %   This is an example of an audio plugin that is
    %

    %   Copyright 2015-2016 The MathWorks, Inc.

    %#codegen

    %--------------------------------------------------
    % Public properties
    %--------------------------------------------------
    properties
        % Center frequencies for each band
        CenterFrequency1 = 100
        CenterFrequency2 = 1000
        CenterFrequency3 = 10000

        % Q factors for each band
        QualityFactor1 = 0.2
        QualityFactor2 = 0.5
        QualityFactor3 = 0.6

        % dB gain for each band
        PeakGain1 = 0
        PeakGain2 = 0
        PeakGain3 = 0
```

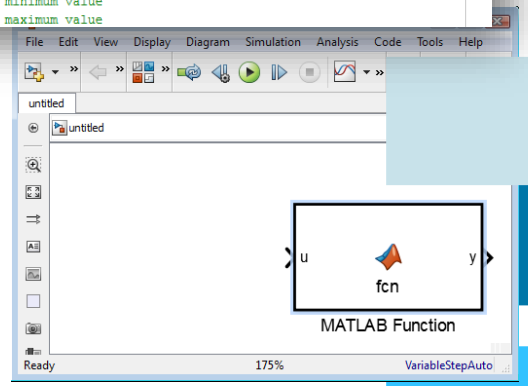



```
>> generateAudioPlugin MyParamEQ
```

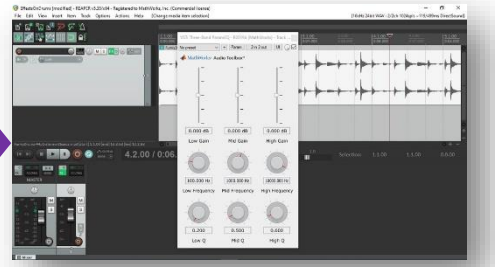
使用 Simulink 进行音频系统开发

基于模型设计工具生态 – 原型与产品

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

        % Appearance and display of plugin parameters is specified by
        % constant properties having the 'Display' suffix.  These specify:
        %     name displayed on the dialog
        %     units displayed on the dialog
        %     minimum value
        %     maximum value
```



Audio Toolbox



音频插件
(VST, AU,…)

MATLAB Coder

Simulink Coder

C/C++

用于手动集成的可移植源代码

Simulink Real-Time

实时原型机

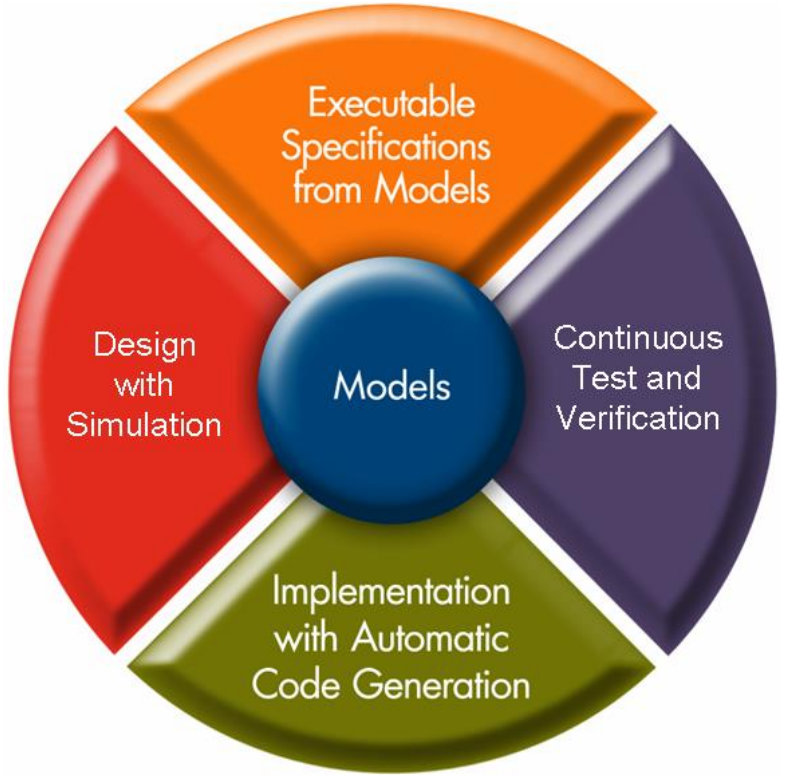
Embedded Coder

优化的嵌入式代码

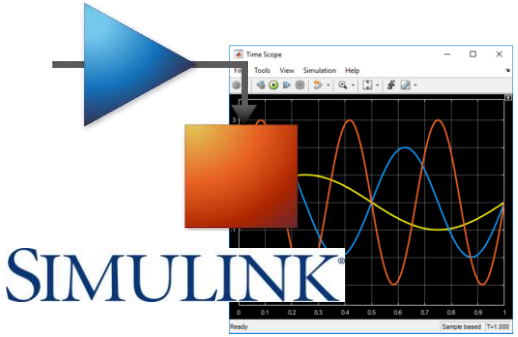
什么是基于模型的设计？

一个集中的开发环境，大型项目的所有方面都与之相连或从中派生

- 将开发重点聚焦于模型
- 随着时间的推移增加设计的复杂性
- 模拟系统性能
- 利用工具的基础功能实现自动化
- 及早发现错误
- 减少工作的重复



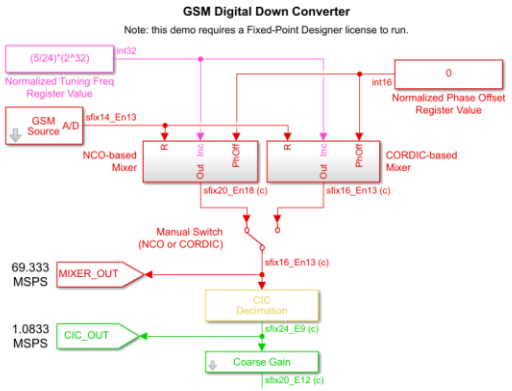
为什么使用 Simulink 进行音频系统设计?



- 跨多个团队和工作流的统一平台
- Simulink 有时间的概念
  - 多采样率，帧和基于采样的信号，同步，自适应系统
- 实时音频 I/O 流与交互式调优和可视化
- 处理具有多个工程领域的复杂系统
  - 模拟、离散、物理系统、人工智能（机器学习/深度学习）
- 原型和产品的代码生成
  - C/C++/HDL代码生成
  - 硬件上的实时调优和性能分析

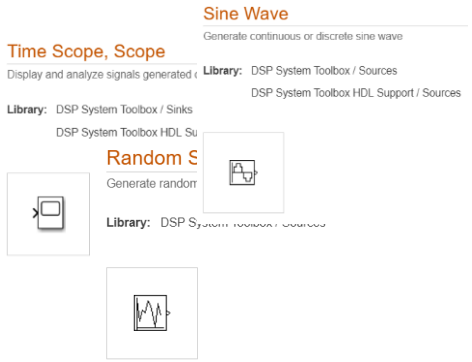
Simulink 专为流式信号处理而设计

Signal Management



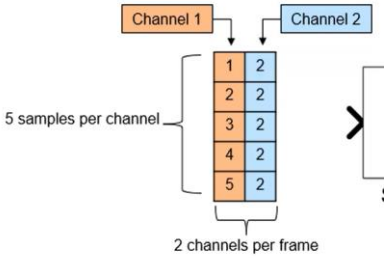
Signal dimensions, datatypes and sample rates

Optimized DSP blocks



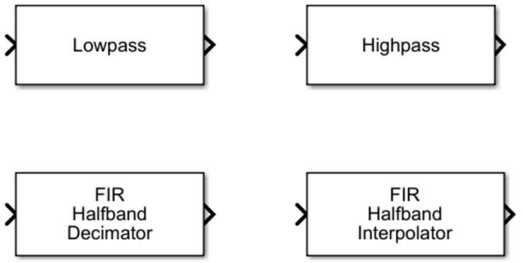
Signal from Workspace, Sine Wave, Delay, Time Scopes

Working with Frames



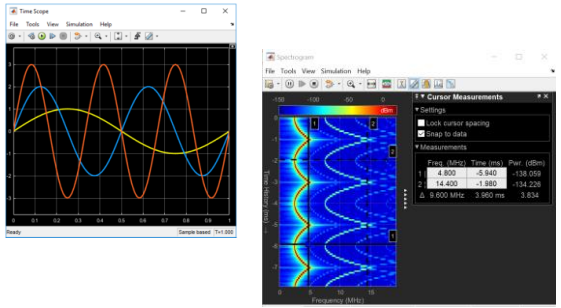
Blocks interpret signals as samples or frames

Filter Blocks



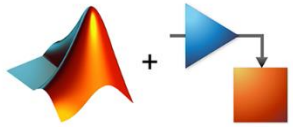
Pre-defined and configured filter blocks
Support for fixed-point and code gen
Parameter tuning

Signal Visualization



Scopes and Spectrum Analyzer in the DSP System Toolbox

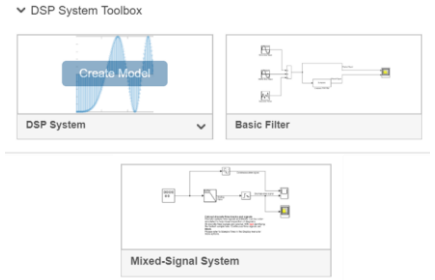
MATLAB and Simulink



Incorporate MATLAB code into Simulink for system level integration

Run Simulink simulations through batching scripting in MATLAB

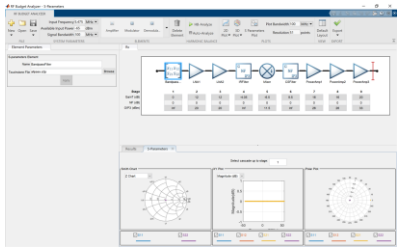
Model Templates and Solvers



Configure Simulink Environment for Signal Processing models

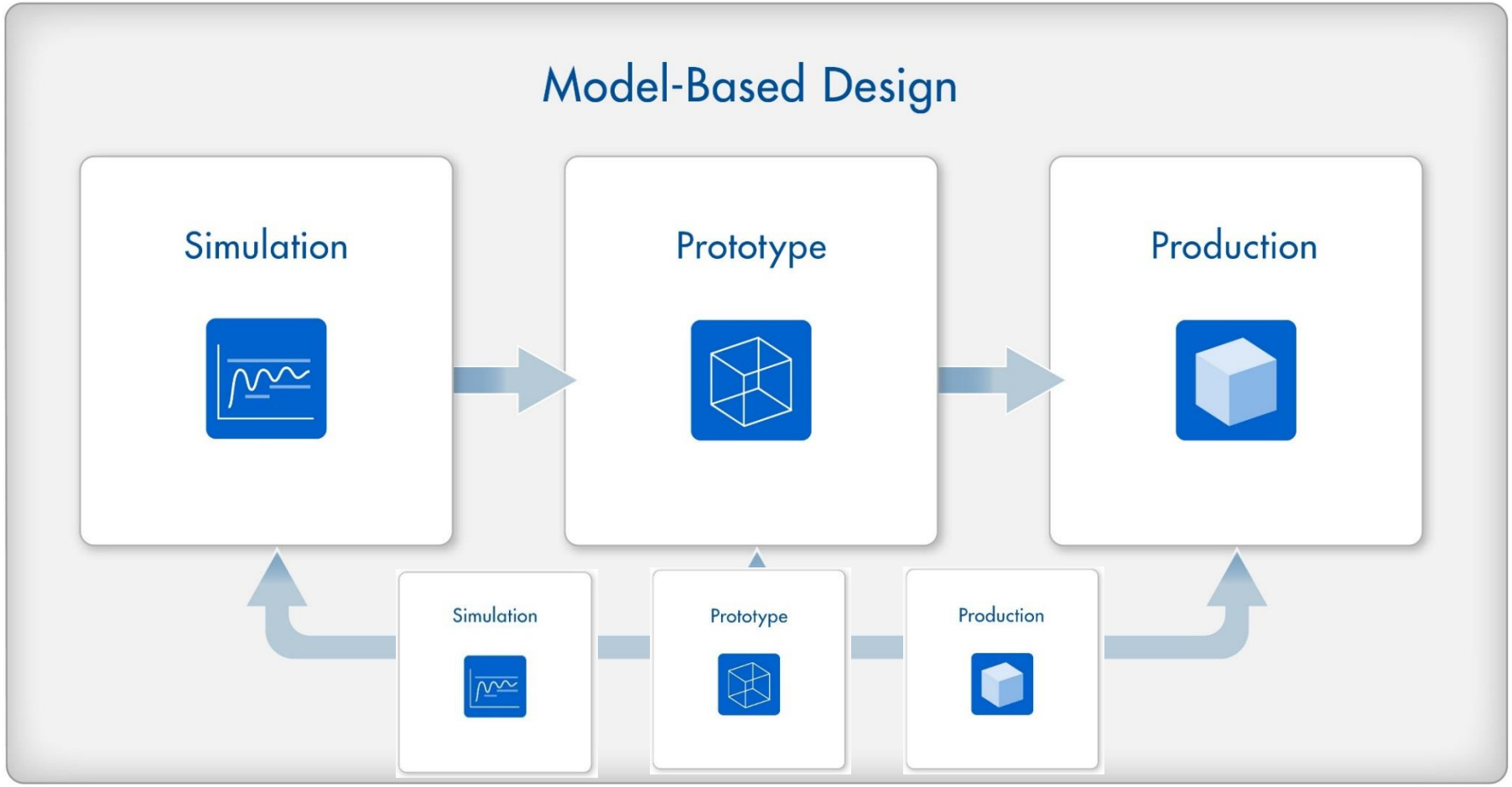
Fixed step solver

Apps to Models



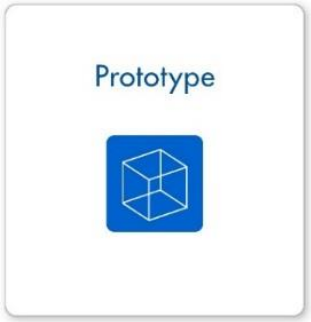
RF Budget Analyzer, Filter Designer, Radar Waveform Analyzer

基于模型的音频设计概述



基于模型的音频设计概述

利用记录的信号迭代开发算法



在 Simulink 或实时硬件中验证信号流

设计和部署产品级软件

音频开发流程 — 在真实的音频基础上开发

Simulation

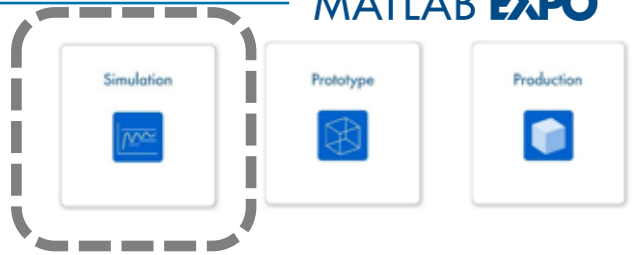
Prototype

Production

声卡，USB音频

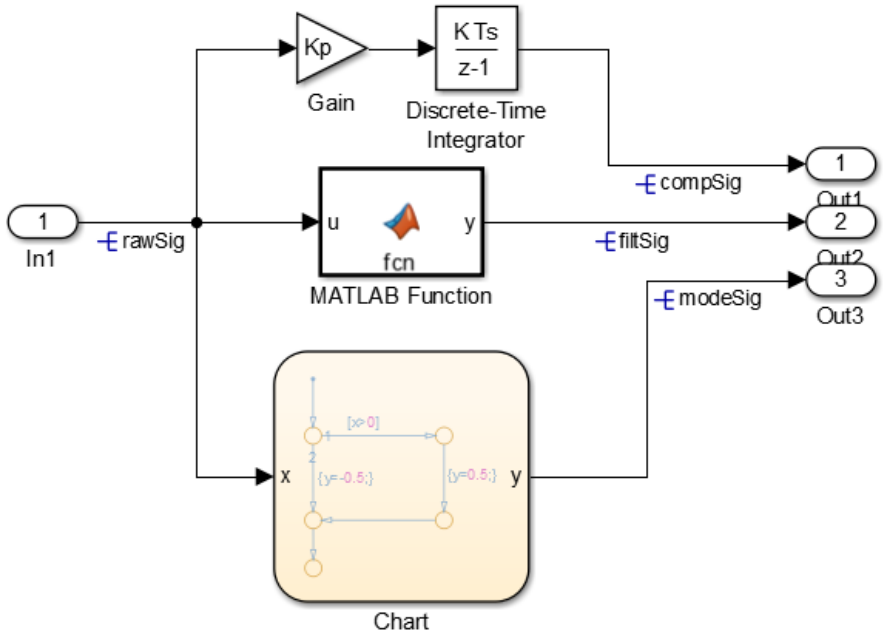
实时 Speedgoat, Raspberry Pi, …

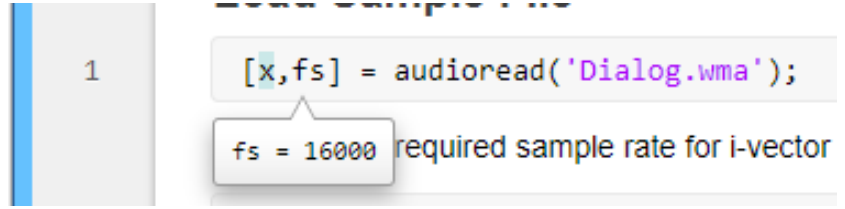
产品级 DSP



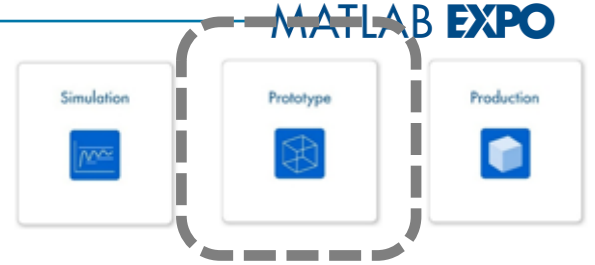
仿真：研究与算法探索

- MATLAB 算法和 Simulink 模型
- 参考函数和模块，用于
  - 信号与音频处理
  - 机器学习和深度学习
- 内置应用程序，可视化，自定义UI
- 高级文件I/O
- 全局文档和调试

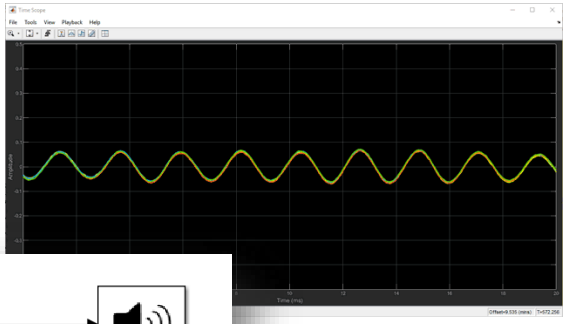


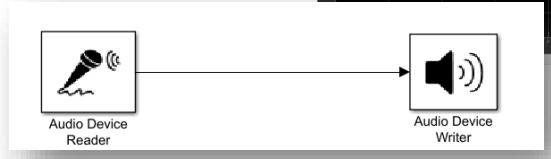


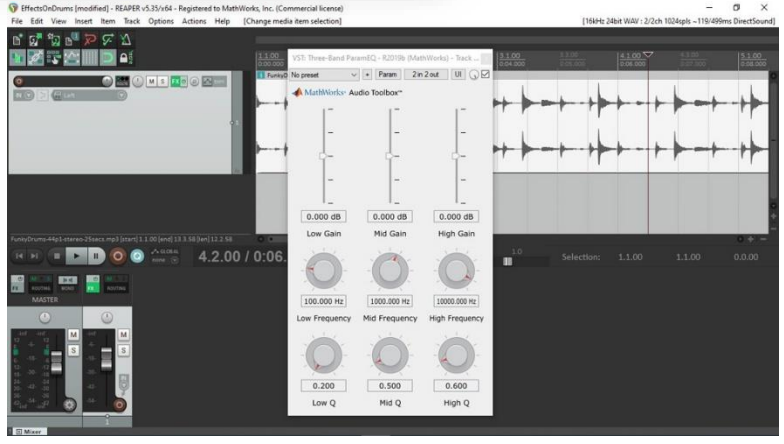
原型：早期基于 PC 的算法验证



- 流式多通道实时音频 I/O
- 连接声卡（ASIO, Core Audio）和主板 I/O
- 低至几毫秒的往返延迟
- 通过 UI 或 MIDI 交互调整参数
- 使用软件示波器进行动态可视化
- 将 MATLAB 算法转换为 VST 和 AU 插件

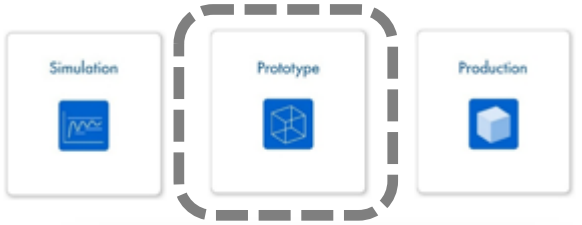


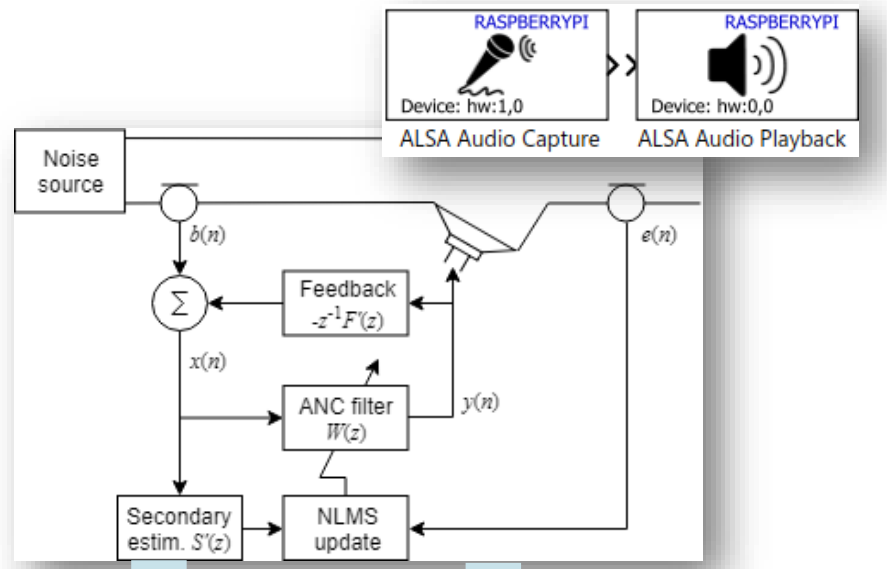


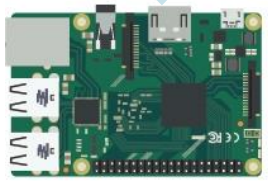


原型：实时硬件上的系统模型

- 在目标硬件上运行系统模型
- 来自 Simulink 的一键自动化
- 基于 C/C++ 的代码生成 + 音频驱动程序
- Speedgoat 音频机的亚毫秒延迟
- 用于自动模型线程的数据流
- 不受约束的可执行文件或“外部模式”

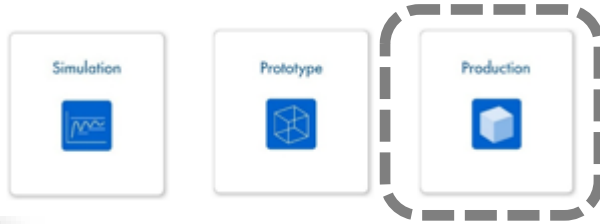


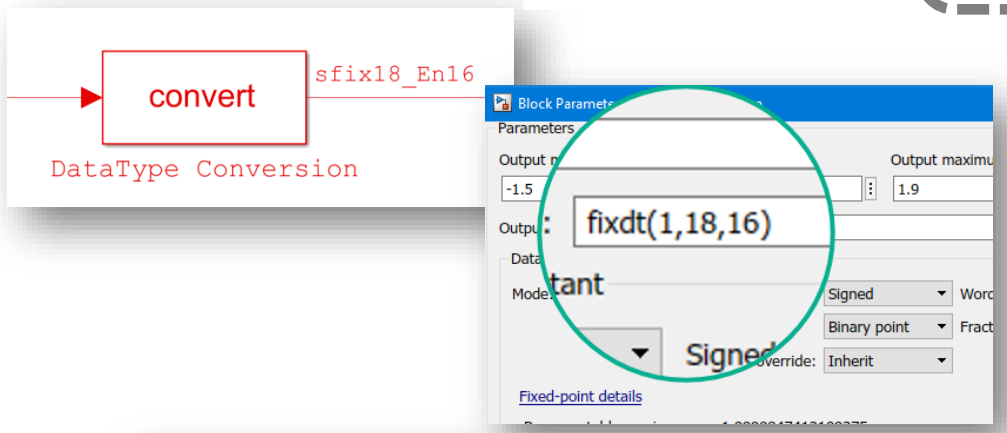


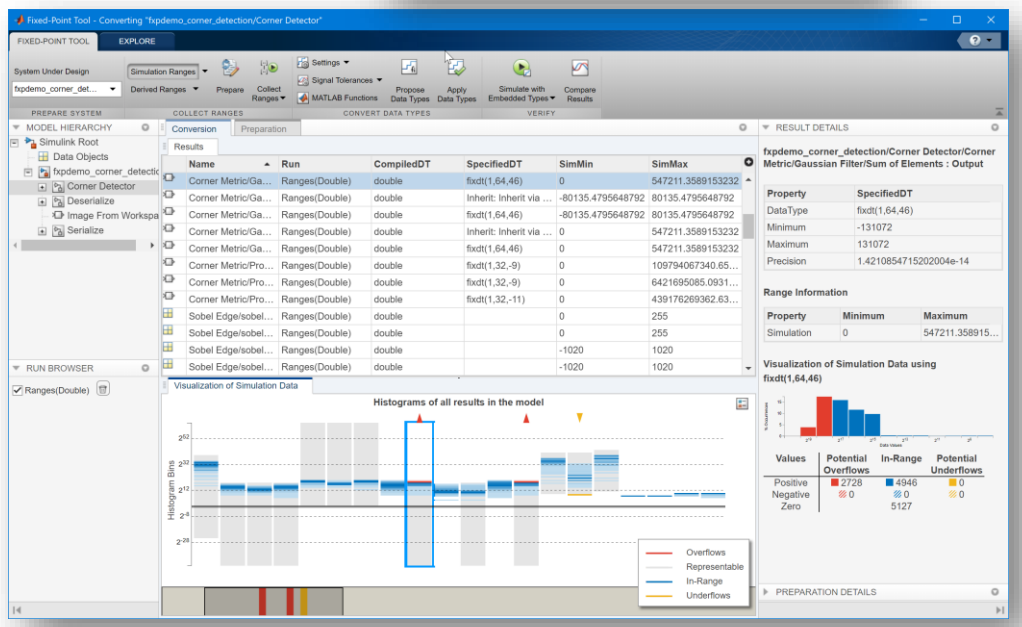


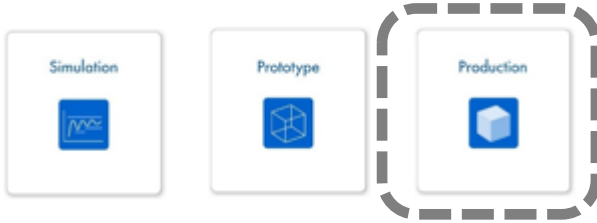
产品：嵌入式算法设计

- 聚焦系统组件
- 定义和使用定点数据类型
- 模拟到比特级的精确响应
- 与“黄金标准”对比验证
- 在早期原型中集成
- 分析溢出及位宽覆盖









产品：嵌入式算法实现

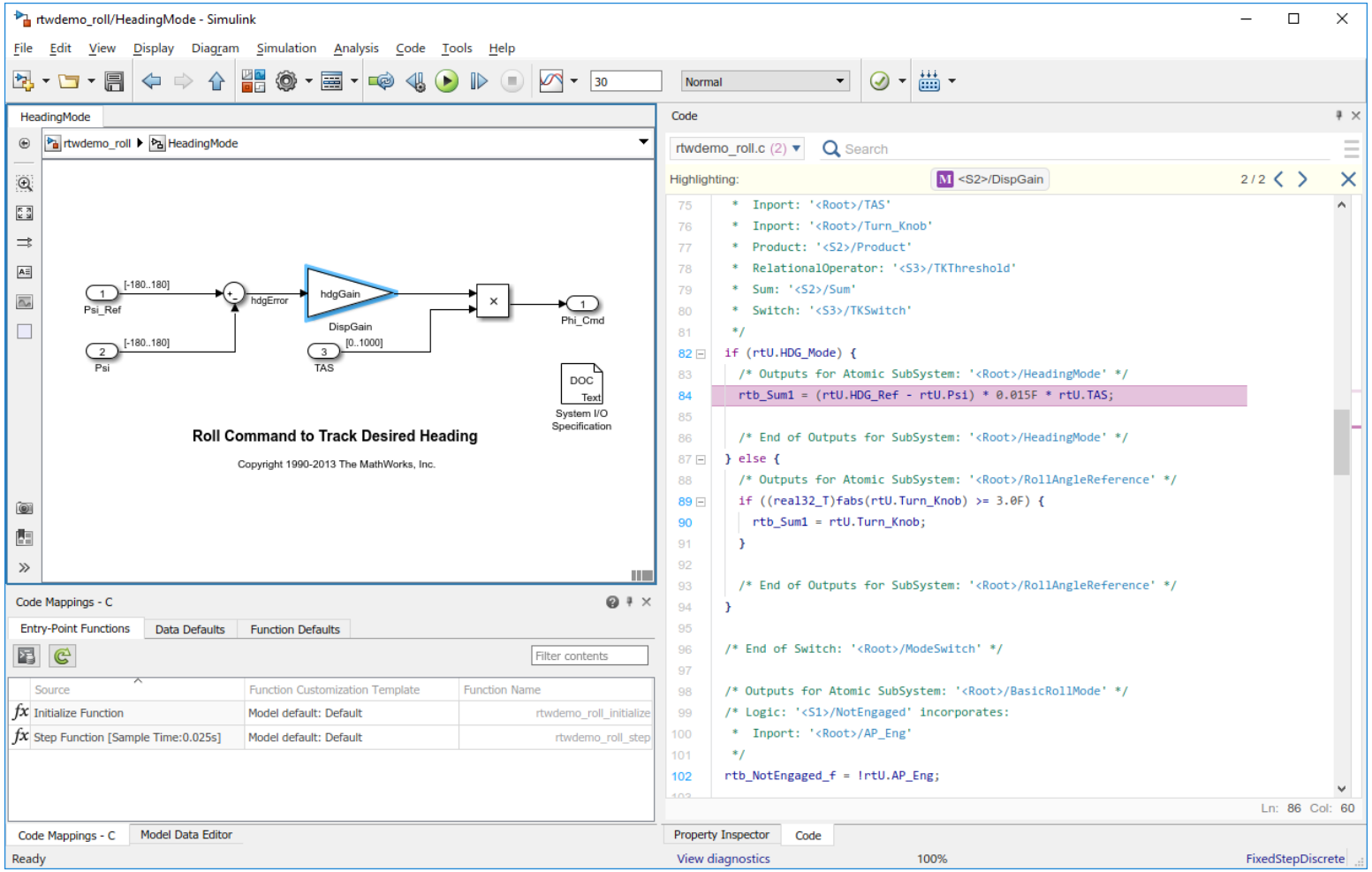
- 聚焦系统组件
- 生成产品 C/C++（或 CUDA, VHDL, Verilog）
- 使用处理器在环（PIL）进行验证
- 通过 PIL 配置 MIPS
- 迭代地优化性能
- 用优化的库和内在函数突破瓶颈
- 自定义代码接口，以便集成到产品中


```
void arm_EC_ops(const float u1[2], const float u2[2],
                float y3[2])
{
  ne10_add_float_neon(&b_y1[0], u1, u2, 2U);
  ne10_sub_float_neon(&y2[0], u1, u2, 2U);
  ne10_mul_float_neon(&y3[0], u1, u2, 2U);
}
```

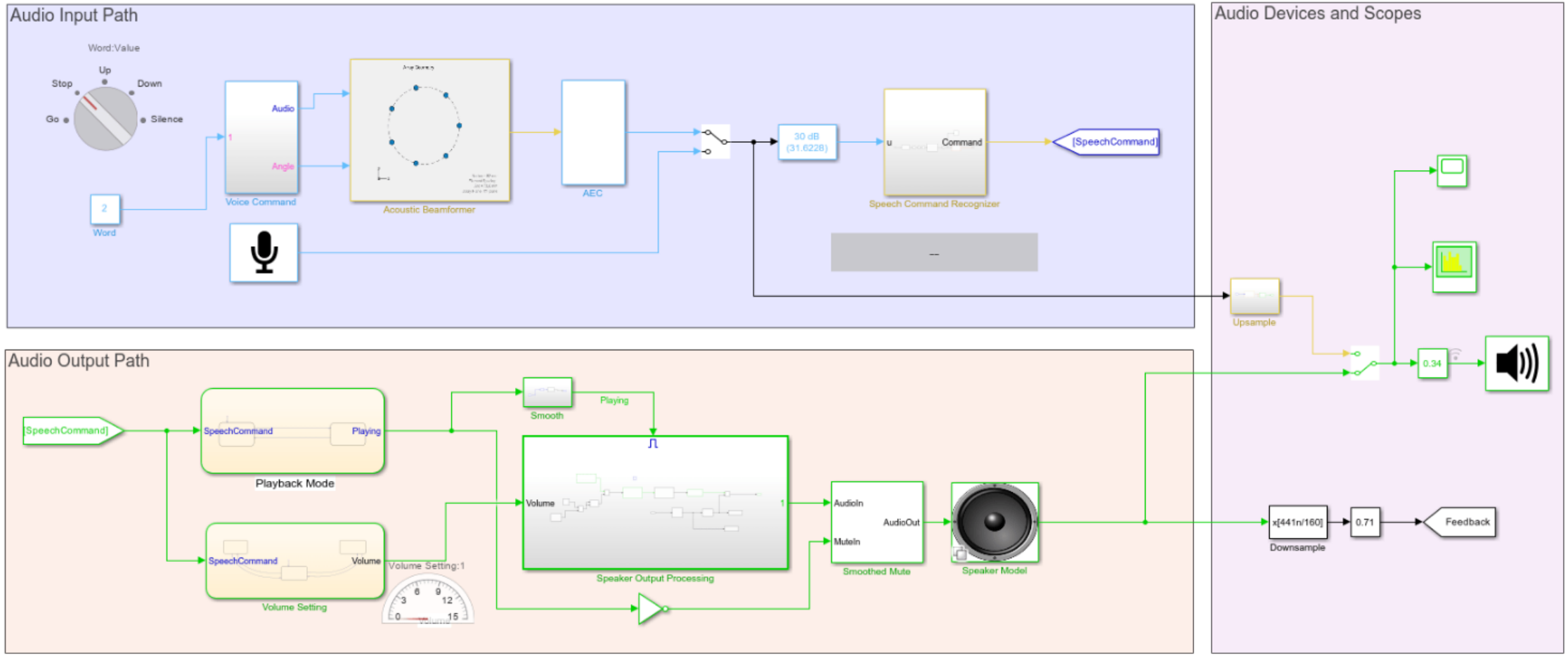
我们如何从模型到代码？

Embedded Coder — 生成为嵌入式系统优化的 C/C++ 代码

- **提高**设计迭代的效率
- 在需求、模型、代码和测试之间进行**跟踪**
- 使用可定制的代码接口和外观**简化**代码集成
- **利用**硬件优化代码，包括ARM处理器的DSP库



智能扬声器 Simulink 示例



[Link to Demo: Model Smart Speaker in Simulink]

智能扬声器 Simulink 示例

- 大型多速率信号处理系统仿真
  - 波束成形，降噪，语音识别，动态处理，滤波等
- 实时调试及可视化
- 基于深度学习的性能优化及 MKL-DNN 库代码生成
- 模块化的系统设计
  - 算法作为 Simulink 模块和 MATLAB 代码执行
  - 支持协作式开发过程和源代码控制

Sonova 利用基于模型的设计缩短了助听器和植入物产品开发周期

挑战

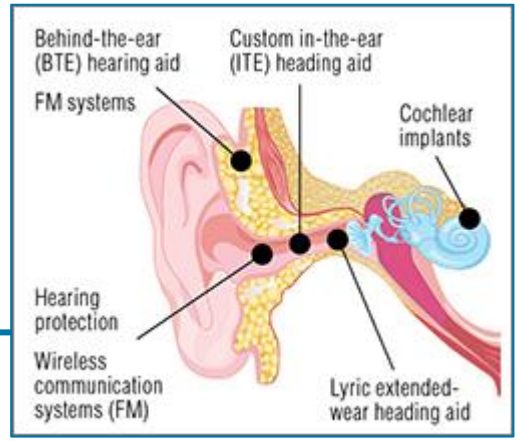
缩短新助听器和植入物的开发时间

解决方案

采用基于模型的设计方法，结合 MATLAB 和 Simulink 开发了一个可重用的数字信号处理组件库，并使用 Simulink Real-Time 对设计思想进行了快速的实时原型设计和测试

结果

- 实时原型在几分钟内更新，而不是几天
- 80%的软件库在平台上被重用
- 提高了软件质量



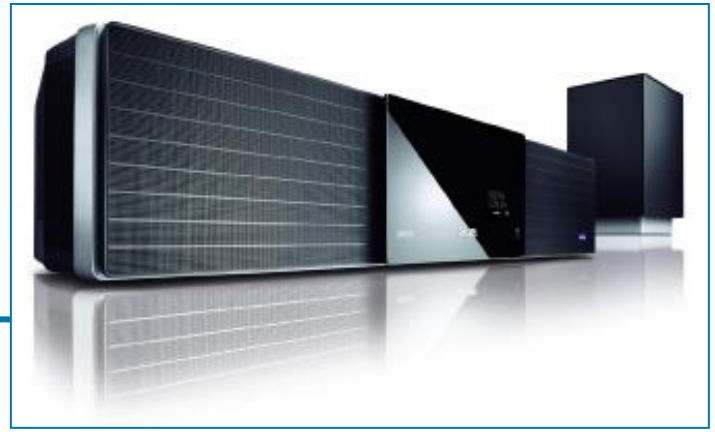
Sonova 的助听器和人工耳蜗解决方案。

“在Sonova，来自不同背景的工程师使用MATLAB和Simulink作为设计创新信号处理系统的共享语言。基于模型的设计和快速的实时原型设计使我们能够保持我们业务需求的产品开发速度。”

Raoul Glatt
Sonova

Link to User Story

飞利浦利用 MathWorks 工具开发一体式环绕立体声系统



集成 Ambisound Soundbar 的家庭影院系统。

挑战

开发一个集成在单一组件高质量的环绕声系统

解决方案

使用 MathWorks 工具开发和测试声学算法，运行实时模拟，并微调参数

结果

- 新算法验证只需几天，而不是几周
- 可执行的演示在一天内准备好
- 80%的设计在未来的项目中被重用

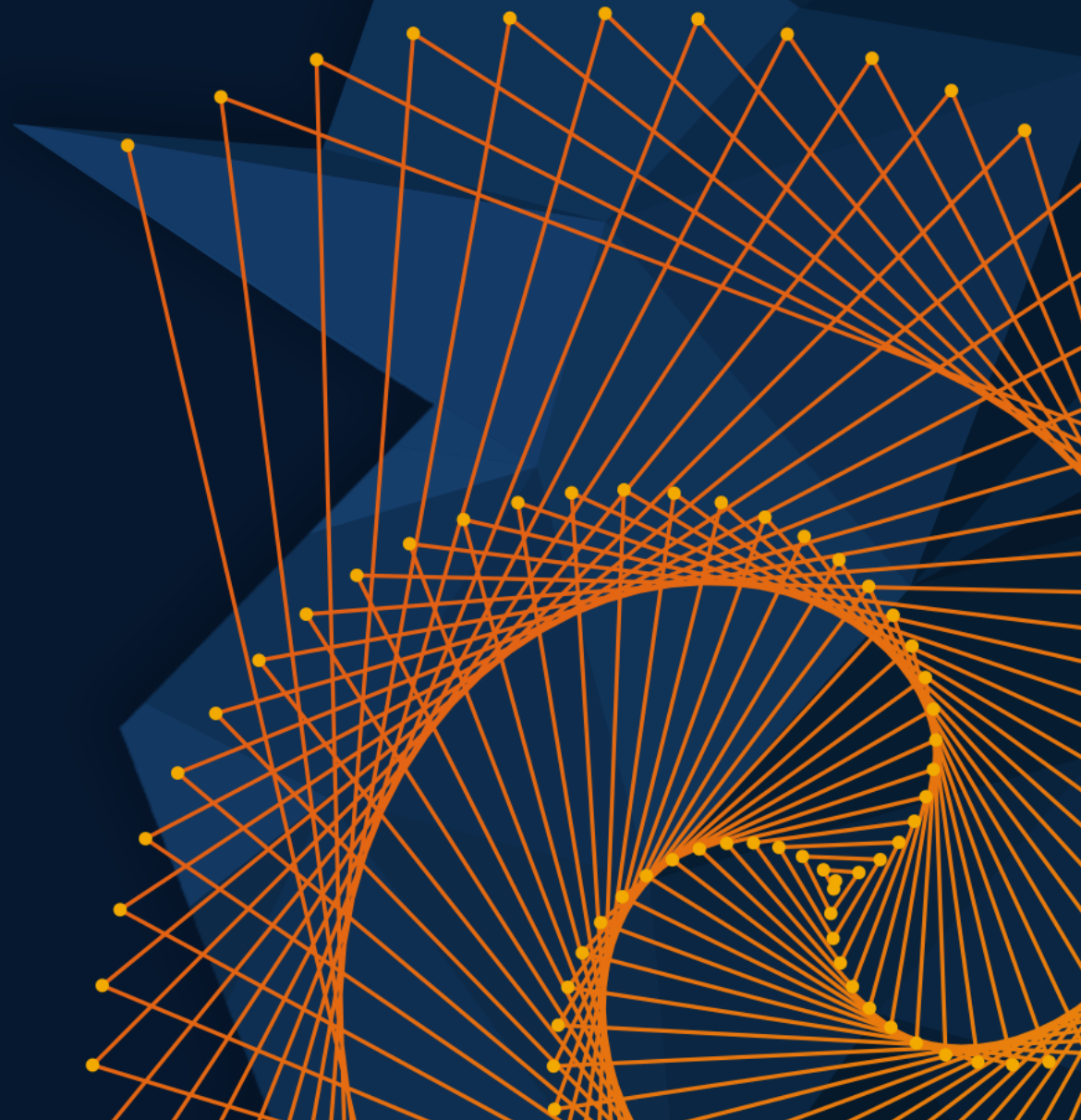
“我们的声学工程师不是专业的程序员。使用Simulink，他们可以快速开发算法并测试他们的想法，而无需编写任何低级DSP代码。只有在确定最佳解决方案之后，才会在DSP上实现合理的算法并将其提交给硬件。”

Georges Aerts
Philips Consumer Lifestyle

Link to User Story

MATLAB EXPO

谢谢



MathWorks®