

【信号処理技術で約40年の実績】 MATLABで信号処理 ～机上シミュレーションから実装までを実例でご紹介～

Mathworks Japan

アプリケーションエンジニアリング部

全体のAgenda

1. MATLABで信号処理

- なぜ、信号処理が重要なのか？
- 信号処理ツールとしてMATLABが選ばれてきた理由
- 信号処理の各種例題のご紹介

2. MATLABで簡単Raspberry Pi実装

- サポートパッケージで簡単セットアップ + 外部ハードとの接続
- まずはLチカ！ + MATLABおよびSimulinkならではのプラスアルファ
- エッジAIとして利用する際の留意点と構成例

3. MATLABで楽々FPGA/ASIC実装

- MATLABおよびSimulinkを用いたレーダーシステムのシミュレーション
- サポートパッケージで楽々環境構築！
- シミュレーションから簡単にHDLを生成、実装、検証

Session1 : MATLABで信号処理

Mathworks Japan

アプリケーションエンジニアリング部

竹本 佳充

Agenda

1. なぜ、信号処理が重要なのか？
2. 信号処理ツールとしてMATLABが選ばれてきた理由
3. 信号処理の各種例題のご紹介

Agenda

1. なぜ、信号処理が重要なのか？
2. 信号処理ツールとしてMATLABが選ばれてきた理由
3. 信号処理の各種例題のご紹介

データセントリックAI

DeepLearning.AI FourthBrain
AIFUND LANDING AI

EXPERT PANEL

Data-centric AI: Real World Approaches

Wed, AUG 11
10 to 11:30am PT
RSVP: <https://data-centric-ai.eventbrite.com>

ANDREW NG
Founder, DeepLearning.AI

ALEXANDER RATNER
Co-founder & CEO, Snorkel AI

JOAQUIN VANSCHOREN
Assistant Professor of Machine Learning, Eindhoven University of Technology

SALWA NUR MUHAMMAD
CEO, FourthBrain

CARLOS ALZATE
General Manager, AIFund Colombia

DILLON LAIRD
Engineering Manager, Landing AI



AI Modeling Reinvented: It's Time to Shift to Better Data, Rather Than Just Building Better Models

October 8, 2021 by Todd R. Weiss



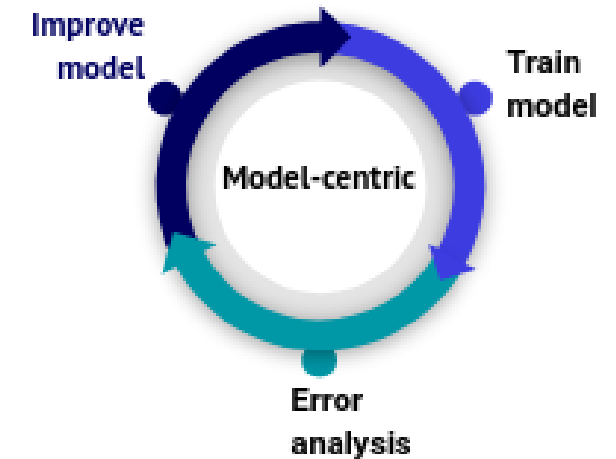
Andrew Ng: Farewell, Big Data

Google Brain co-founder and Landing AI founder Andrew Ng has become an evangelist for what he calls the data-centric AI movement. "Collecting more data often helps," he says. "But if you try to collect more data for

spectrum.ieee.org

"In many industries where giant data sets simply don't exist, I think the focus has to shift from big data to good data. Having 50 thoughtfully engineered examples can be sufficient to explain to the neural network what you want it to learn."

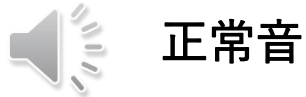
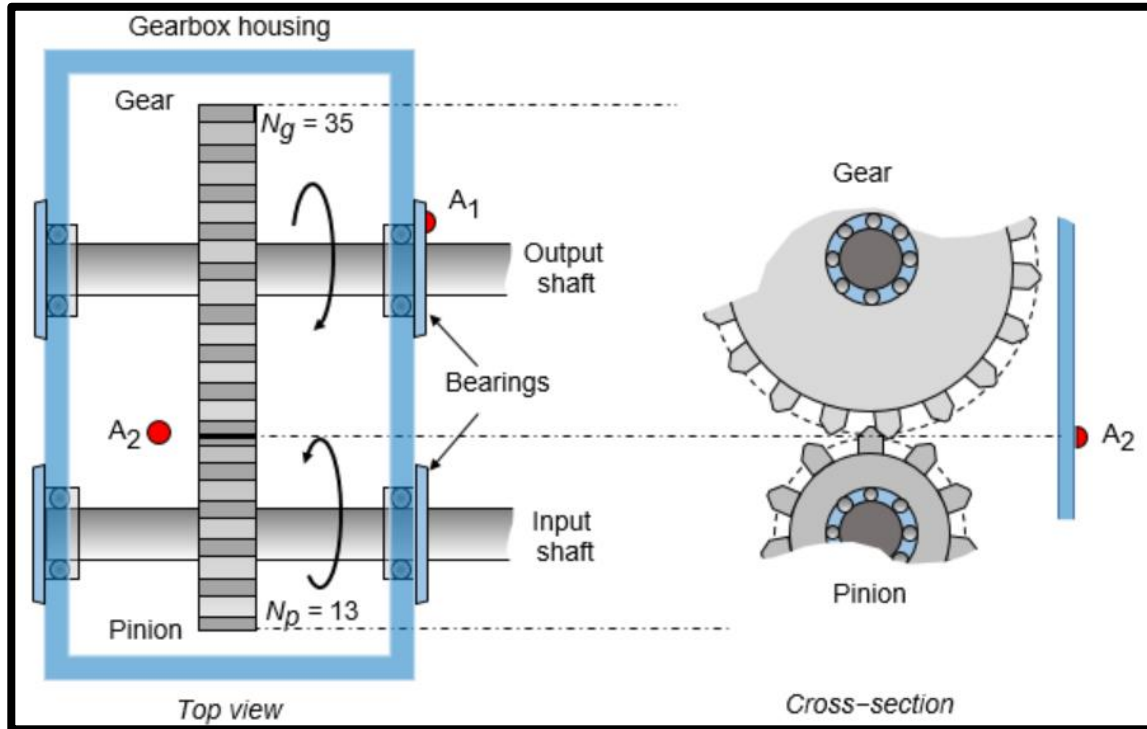
Fixed data, evolving model



Fixed model, evolving data

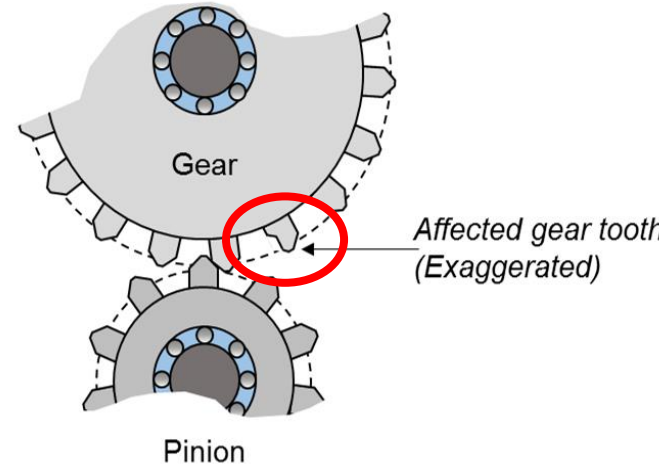


ギアの異常判別例：官能評価から自動判定へ

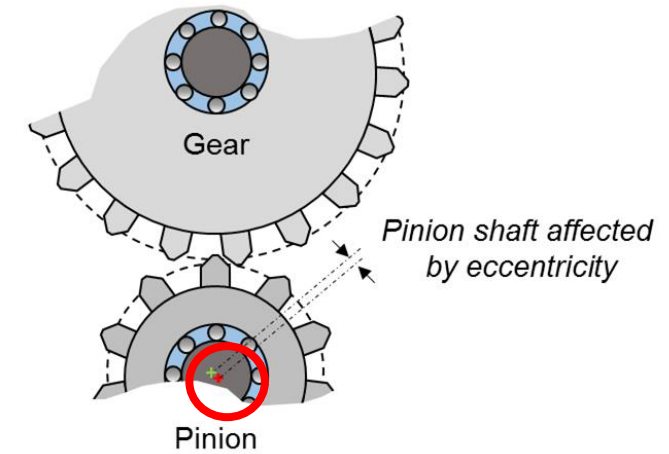


官能評価から自動判定へ
⇒ 波形を眺めてみる

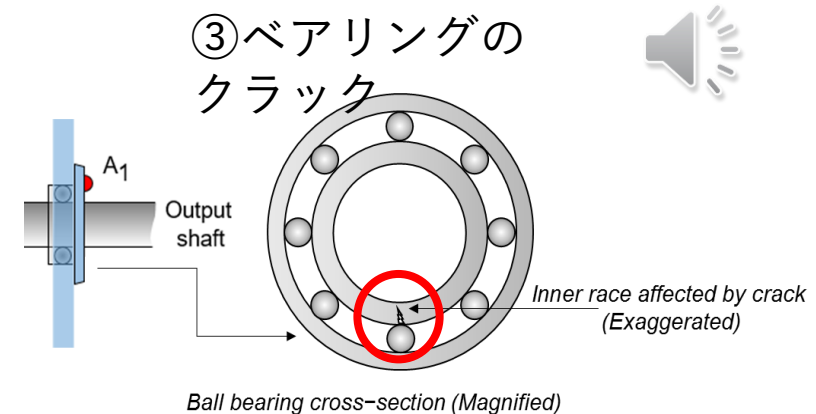
① 大歯の欠け



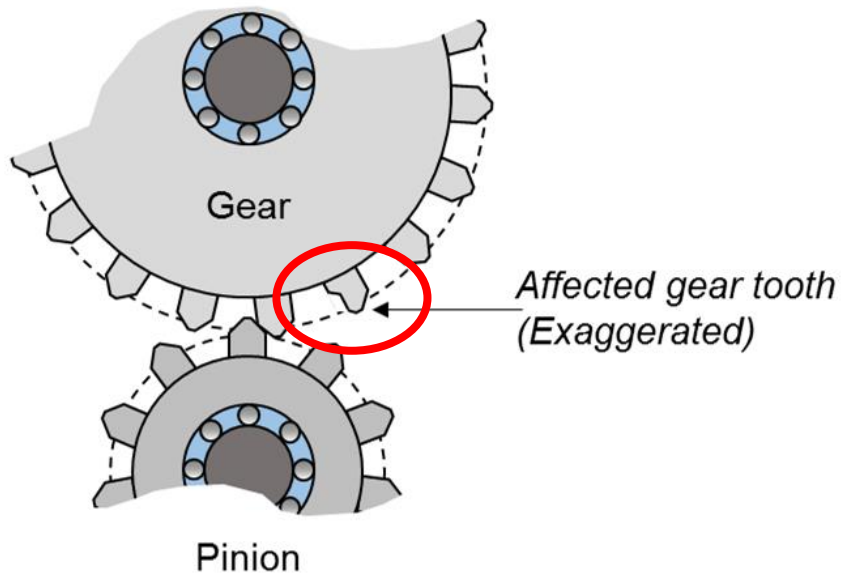
② 小歯の偏芯



③ ベアリングのクラック

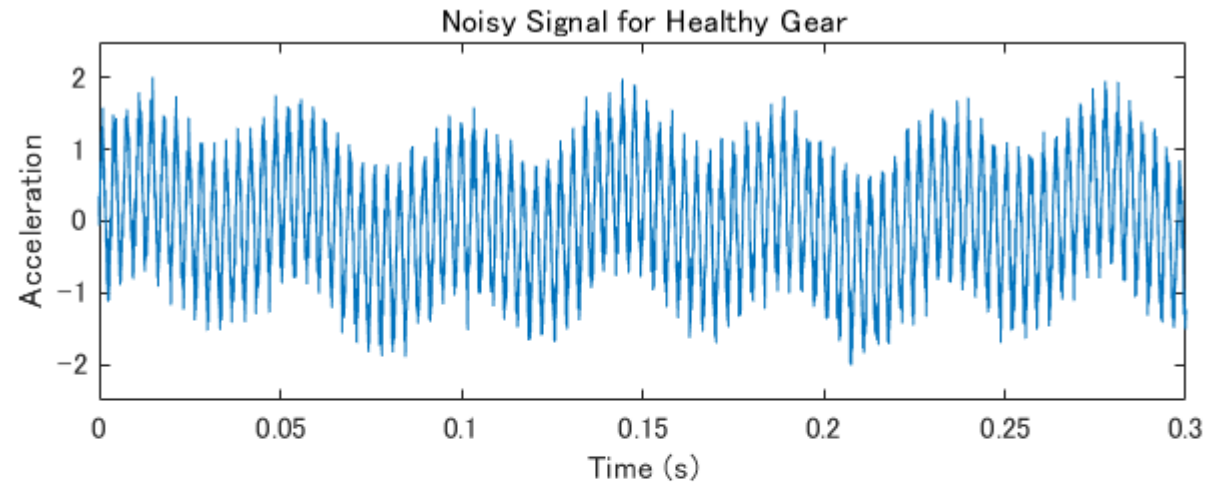


まずは時間波形で波形を観測

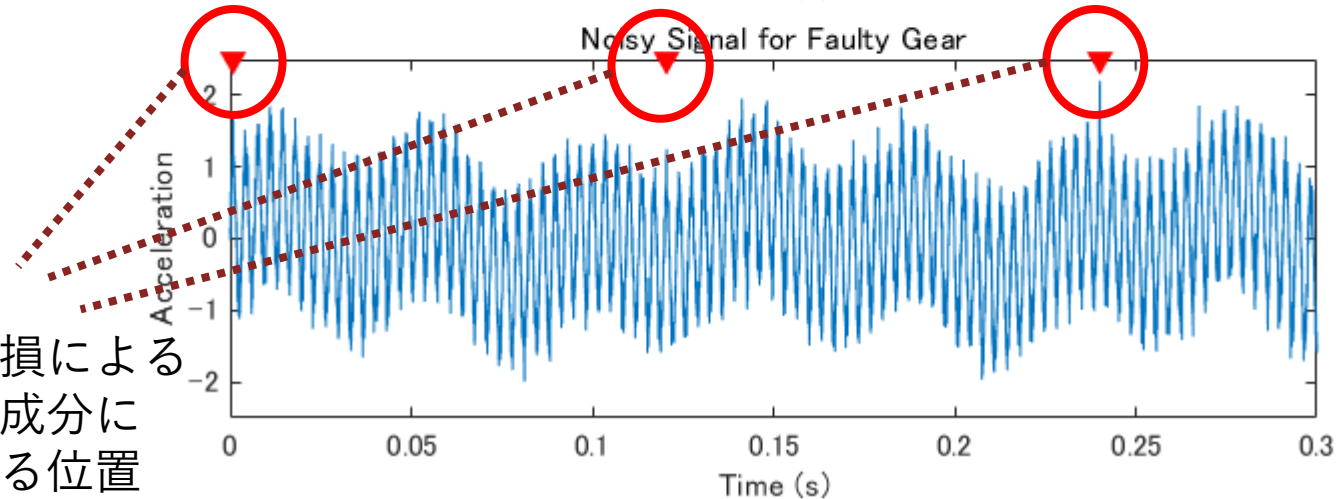


1/fMesh (0.25 ミリ秒) の 8% の時間で
2 kHz の振動信号が発生すると仮定

時間軸では判別が困難
⇒ 周波数軸で眺めてみる



破損なし



破損あり

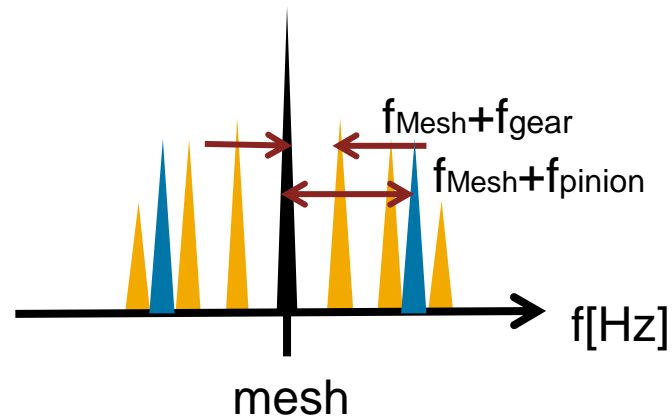
歯の破損による
高周波成分に
相当する位置

信号処理の活用：周波数領域から眺める

破損に起因する周波数成分

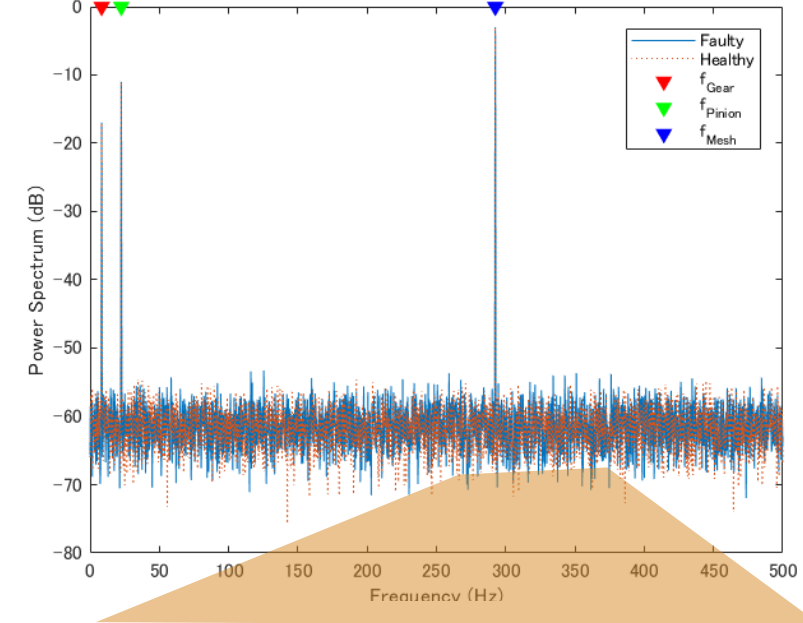
$$f_{\text{sideband,Pinion}} = f_{\text{Mesh}} \pm m \times f_{\text{Pinion}} \quad \forall m \in \{1, 2, 3, \dots\}$$

$$f_{\text{sideband,Gear}} = f_{\text{Mesh}} \pm m \times f_{\text{Gear}} \quad \forall m \in \{1, 2, 3, \dots\}$$

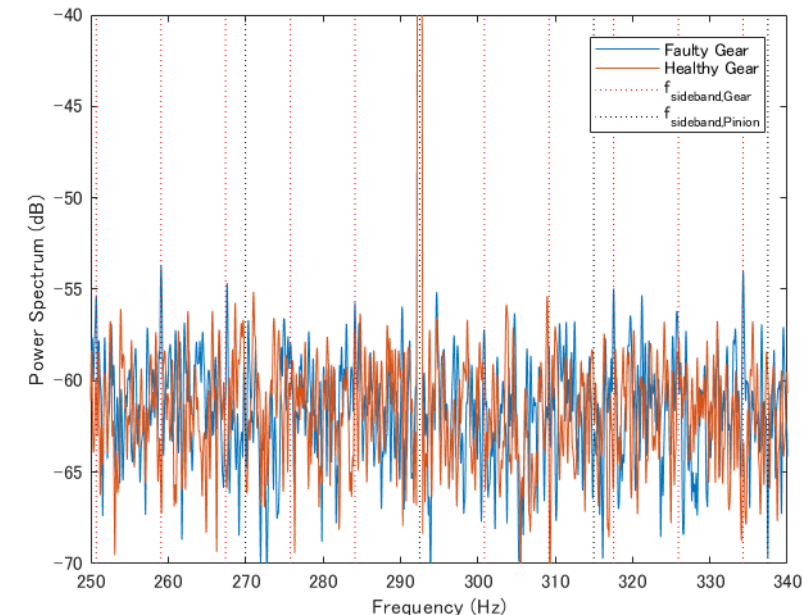


```
[Spect, f] = pspectrum(...  
[vFaultNoisy' vNoFaultNoisy'], ...  
Fs, 'FrequencyResolution', 0.2, ...  
'FrequencyLimits', [0 500]);
```

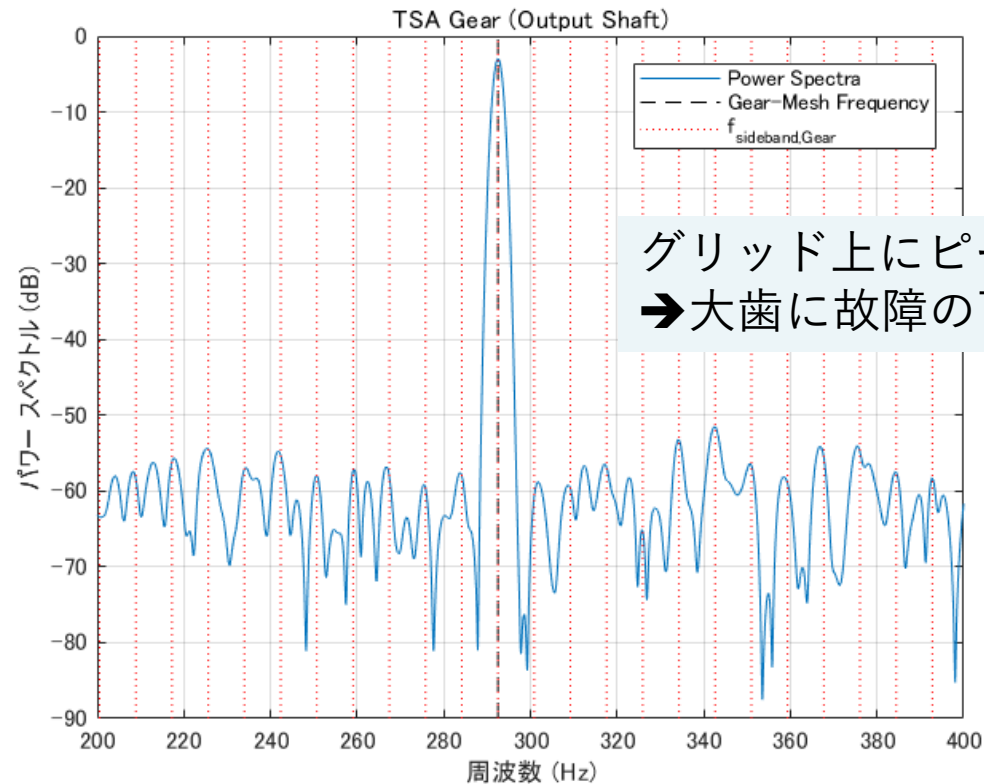
周波数軸でも、雑音に埋もれて観測は困難
⇒ TSA(時間同期平均) による前処理



拡大



効果的な前処理により特徴が明らかに



- 信号処理技術の使いどころ
 - 様々なドメインから眺める
 - 効果的な前処理で信号の本質を知る
 - 特徴的な成分を見極める
-
- 信号処理技術の効果
 - AIまかせからの脱却
 - 現場の知見を活かせる

信号処理・AIシステム開発のワークフロー

1. データセットの作成

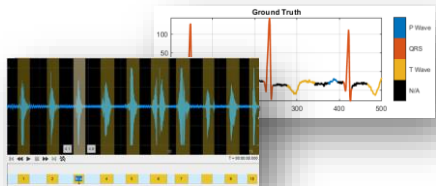
■ データソース



■ シミュレーション ■ オグメンテーション

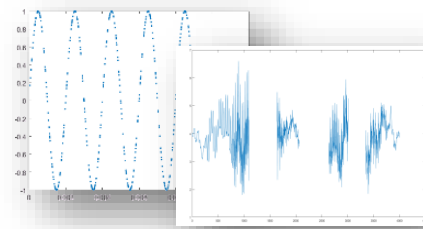


■ データラベリング

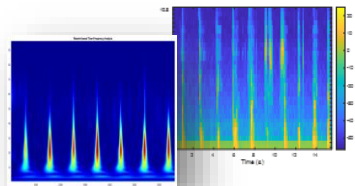


2. 前処理と変換

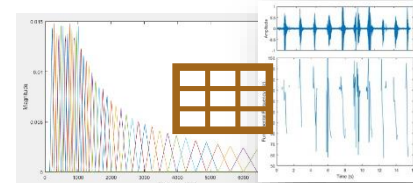
■ 前処理



■ 変換



■ 特徴抽出

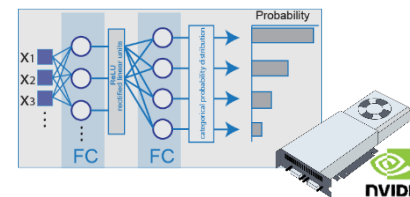


3. 予測モデルの開発

■ リファレンスモデルの流用 ■ オリジナルモデルの作成



■ GPUによる高速な学習



■ ハイパーパラメータの 解析とチューニング



4. 実システムへの展開

■ デスクトップアプリ



■ エンタープライズシステム

Java
MATLAB
C/C++
Python

■ エッジ（組み込み）デバイス



Agenda

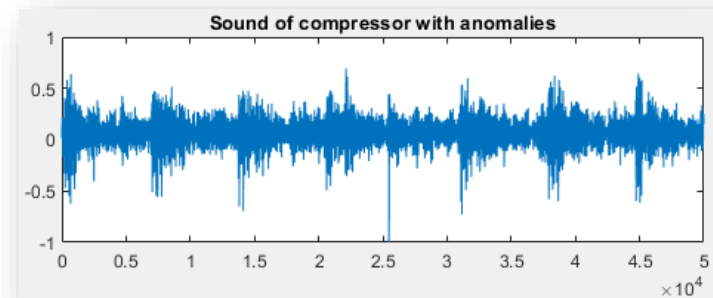
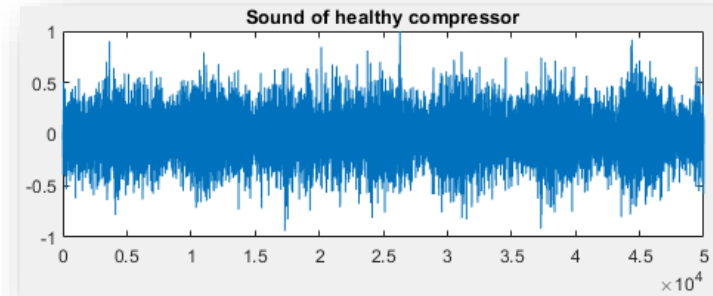
1. なぜ、信号処理が重要なのか？
2. 信号処理ツールとしてMATLABが選ばれてきた理由
3. 信号処理の各種例題のご紹介

MATLABが選ばれる理由

1. データセットの置き換えですぐ試せる
2. 具体的かつ実践的な例題
3. 便利なアプリの活用

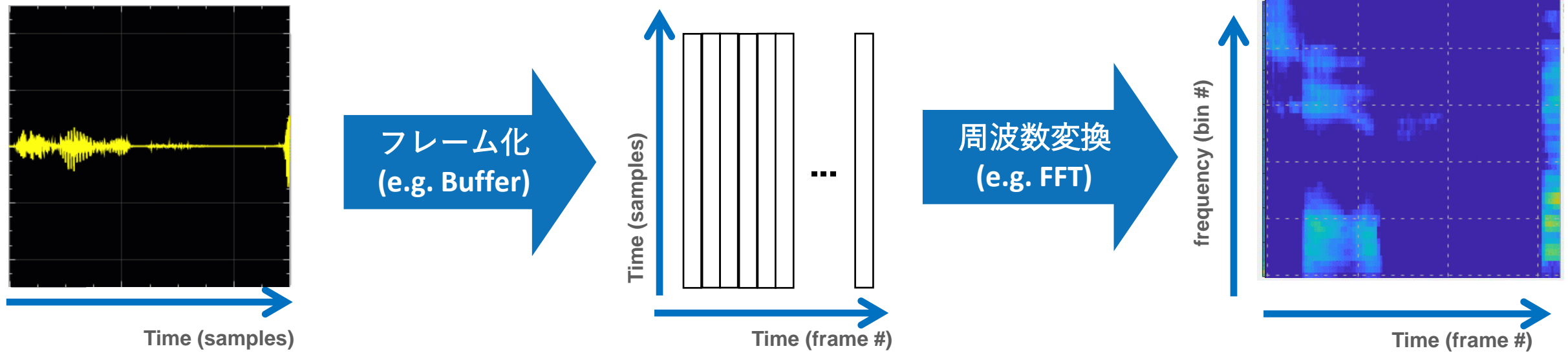
エアコンプレッサーの異音判別例

- ラベル付き録音データセット
- 1種類の正常データ
- 7種類の異音データ
 - ベアリング異常
 - バルブ異常
 - ピストン異常
 - その他4種
- 1800点の .wavファイル
- 各データにつき225ファイル(3秒程度)



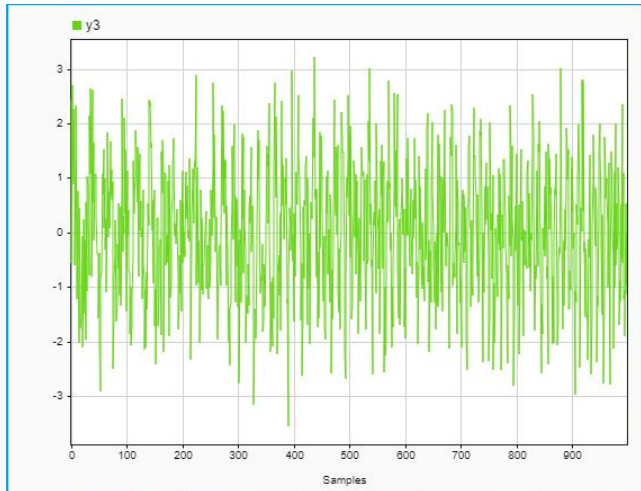
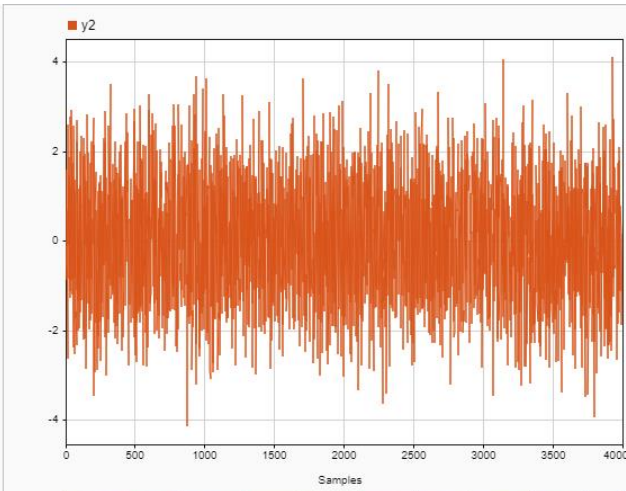
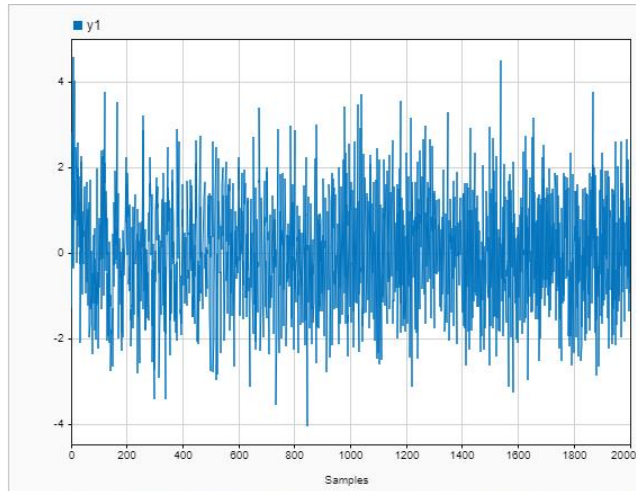
[Example: Transfer Learning with Pretrained Audio Networks in Deep Network Designer](#)

時間-周波数変換



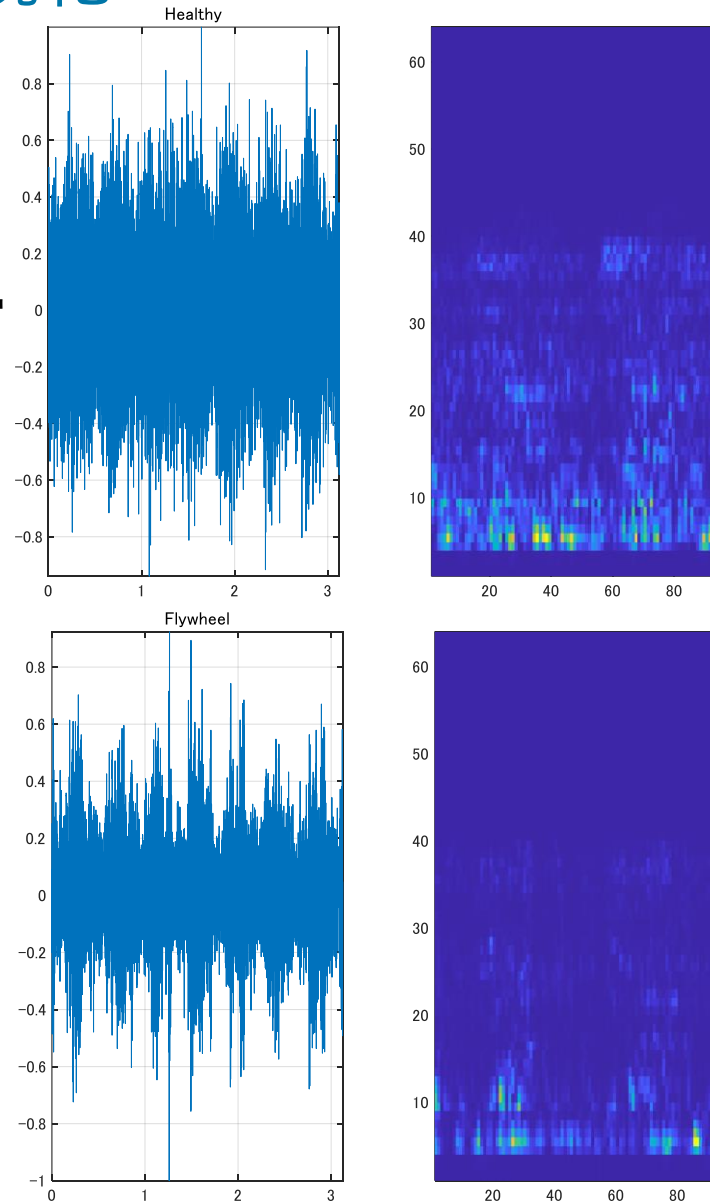
時間-周波数変換は、信号の特性を際立たせる

時間-周波数変換

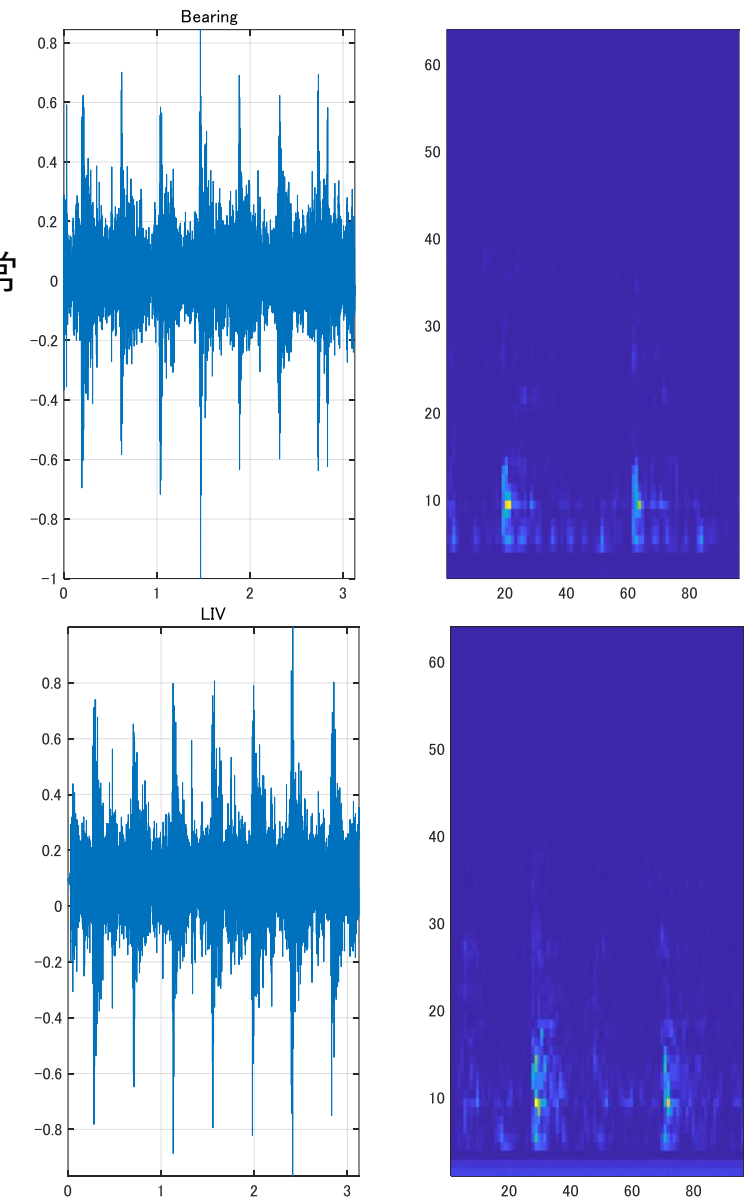


各波形の可視化

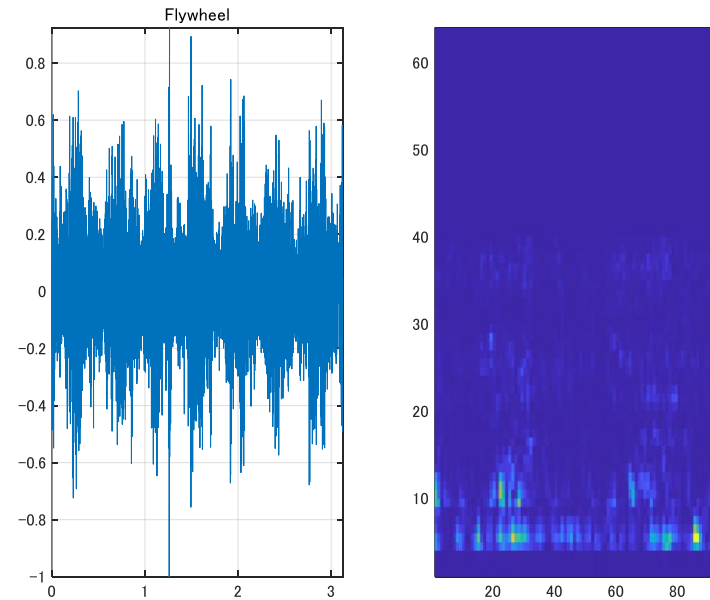
□ 正常



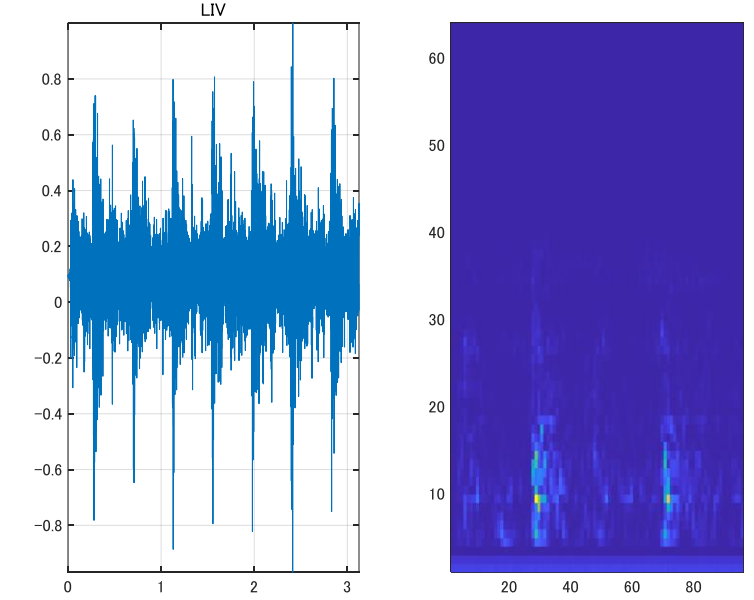
□ ベアリング異常



□ フライホイール異常

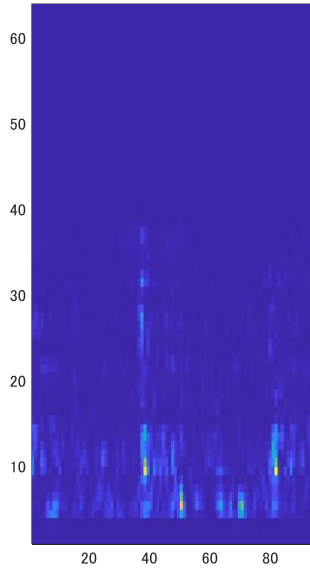
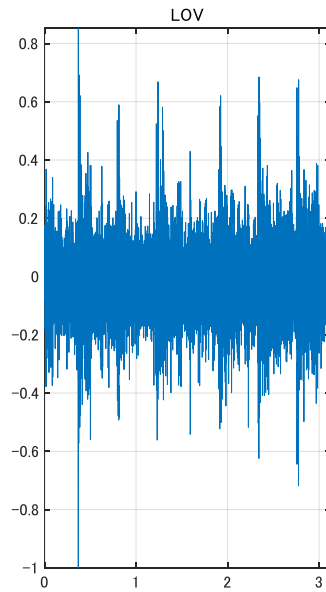


□ インレット
バルブ漏れ異常

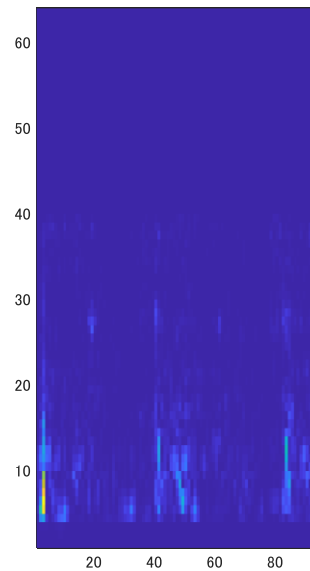
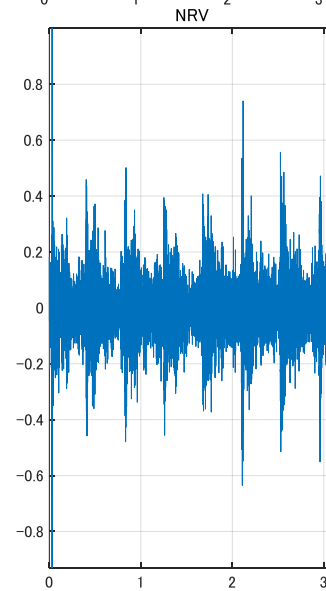


各波形の可視化 (cont'd)

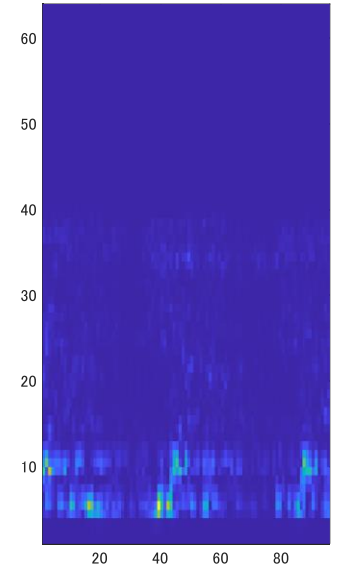
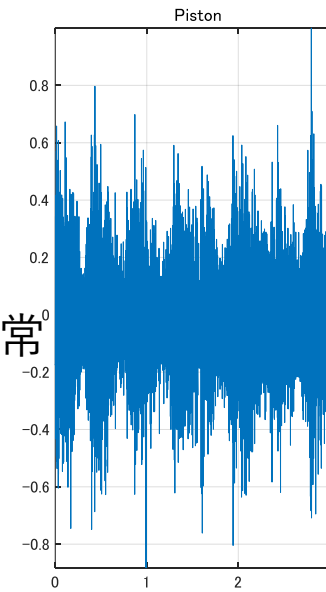
□ アウトレット
バルブ漏れ異常



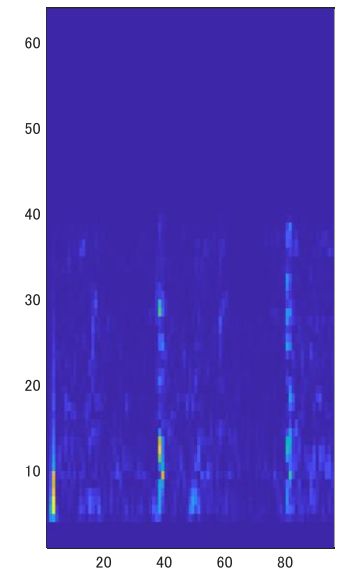
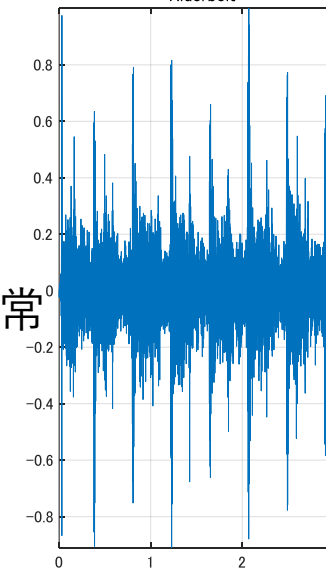
□ ノンリターン
バルブ異常



□ ピストンリング異常



□ ライダーベルト異常



MATLAB R2022a

HOME

PLOTS

APPS

LIVE EDITOR

INSERT

VIEW

Level A Update

Cleanup

Search Documentation

Gabriele

New

Open

Save

Compare

Print

Export

Go To

Find

Bookmark

Text

Normal

B

I

U

M

Code

Control

Task

Refactor

Run Section

Run and Advance

Run to End

Run

Step

Stop

C:\Users\gbunkhei\OneDrive - MathWorks\Documents\MATLAB\Examples\R2022a\deeplearning_shared\TransferLearningWithAudioNetworkInDeepNetworkDesignerExample

Current Folder

Command History

Live Script

Transf...425 KB 07/04/2022 ...

Details

Workspace

Name	Value
ads	1x1 audioData
adsTest	1x1 audioData
adsTrain	1x1 audioData
adsValidation	1x1 audioData
datasetLocation	'C:\Users\gbun
downloadFolder	'C:\Users\gbun
tdsTrain	1x1 Transform
tdsValidation	1x1 Transform
url	'https://www.n

Transfer Learning with Pretrained Audio Networks in Deep Network Designer

This example shows how to interactively fine-tune a pretrained network to classify new audio signals using Deep Network Designer.

Transfer learning is commonly used in deep learning applications. You can take a pretrained network and use it as a starting point to learn a new task. Fine-tuning a network with transfer learning is usually much faster and easier than training a network with randomly initialized weights from scratch. You can quickly transfer learned features to a new task using a smaller number of training signals.

This example retrains YAMNet, a pretrained convolutional neural network, to classify a new set of audio signals. This example requires Audio Toolbox™ and Deep Learning Toolbox™.

Load Data

Download and unzip the air compressor data set [1]. This data set consists of recordings from air compressors in a healthy state or one of 7 faulty states.

```
1 url = 'https://www.mathworks.com/supportfiles/audio/AirCompressorDataset/AirCompressorDataset.zip';
2 downloadFolder = fullfile(tempdir,'aircompressordataset');
3 datasetLocation = tempdir;
4
5 if ~exist(fullfile(tempdir,'AirCompressorDataSet'),'dir')
6     loc = websave(downloadFolder,url);
7     unzip(loc,fullfile(tempdir,'AirCompressorDataSet'))
8 end
```

Create an audioDatastore object to manage the data and split it into training, validation, and test sets.

Command Window

```
fx >>
```

Zoom: 100%

UTF-8

LF

script

混同行列による検証

学習済み
ネットワーク

クラスが未知の
特徴量

% 学習済みネットワークによる分類

```
ValPred = classify(net, single(validationFeatures));
```

% 混同行列の表示

```
confusionchart(validationLabels, ValPred);
```

真のクラス

予測クラス

ご自身のデータに置き換えて
是非お試しください

混同行列

真のクラス	Bearing	212	5					8		94.2%	5.8%
	Flywheel		216	1		3	3	2		96.0%	4.0%
	Healthy			225						100.0%	
	LIV				225					100.0%	
	LOV				4	219	1	1		97.3%	2.7%
	NRV				15		210			93.3%	6.7%
	Piston		3					222		98.7%	1.3%
	Riderbelt				9	1			215	95.6%	4.4%
		100.0%	96.4%	99.6%	88.9%	98.2%	98.1%	95.3%	100.0%		
			3.6%	0.4%	11.1%	1.8%	1.9%	4.7%			
		Bearing	Flywheel	Healthy	LIV	LOV	NRV	Piston	Riderbelt	予測されたクラス	

異常を正常と判断

正常を異常と判断

MATLABが選ばれる理由

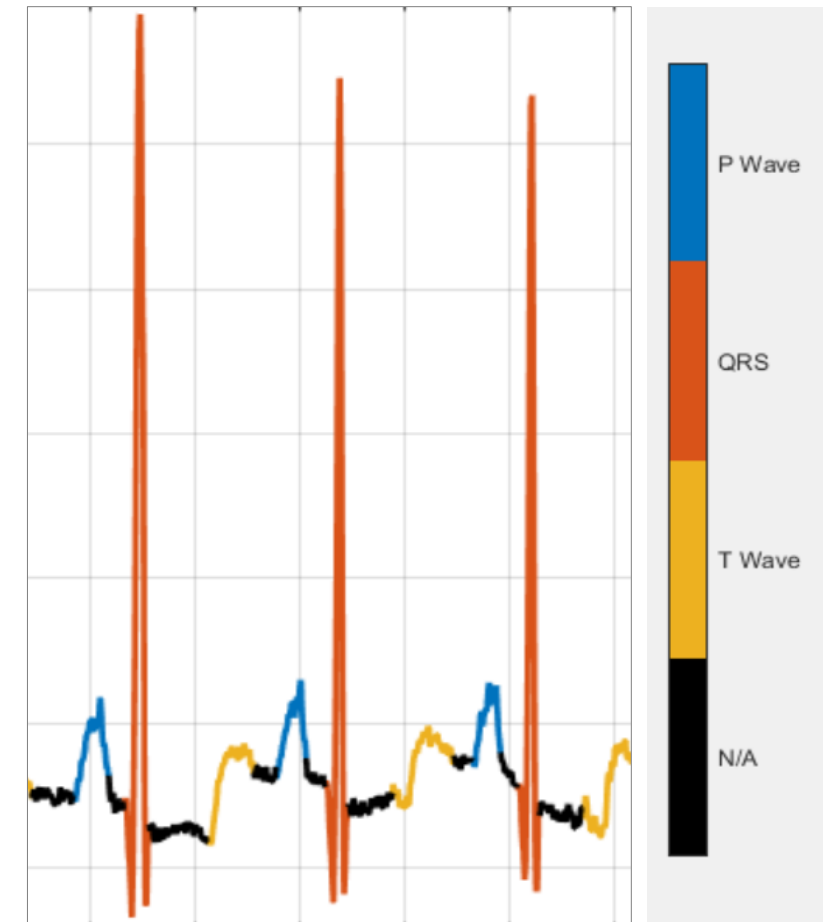
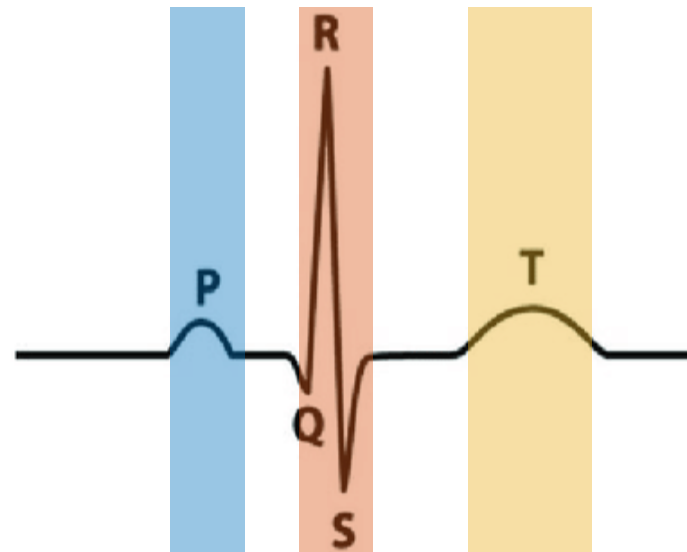
1. データセットの置き換えですぐ試せる

2. 具体的かつ実践的な例題

3. 便利なアプリの活用

信号の領域分割の自動化例 ～性能改善の手順～

- 専門医がラベル付けしたデータセット
- 3種類の波形領域
- 210のECGデータ(～15 分)



[Example: Waveform Segmentation Using Deep Learning](#)

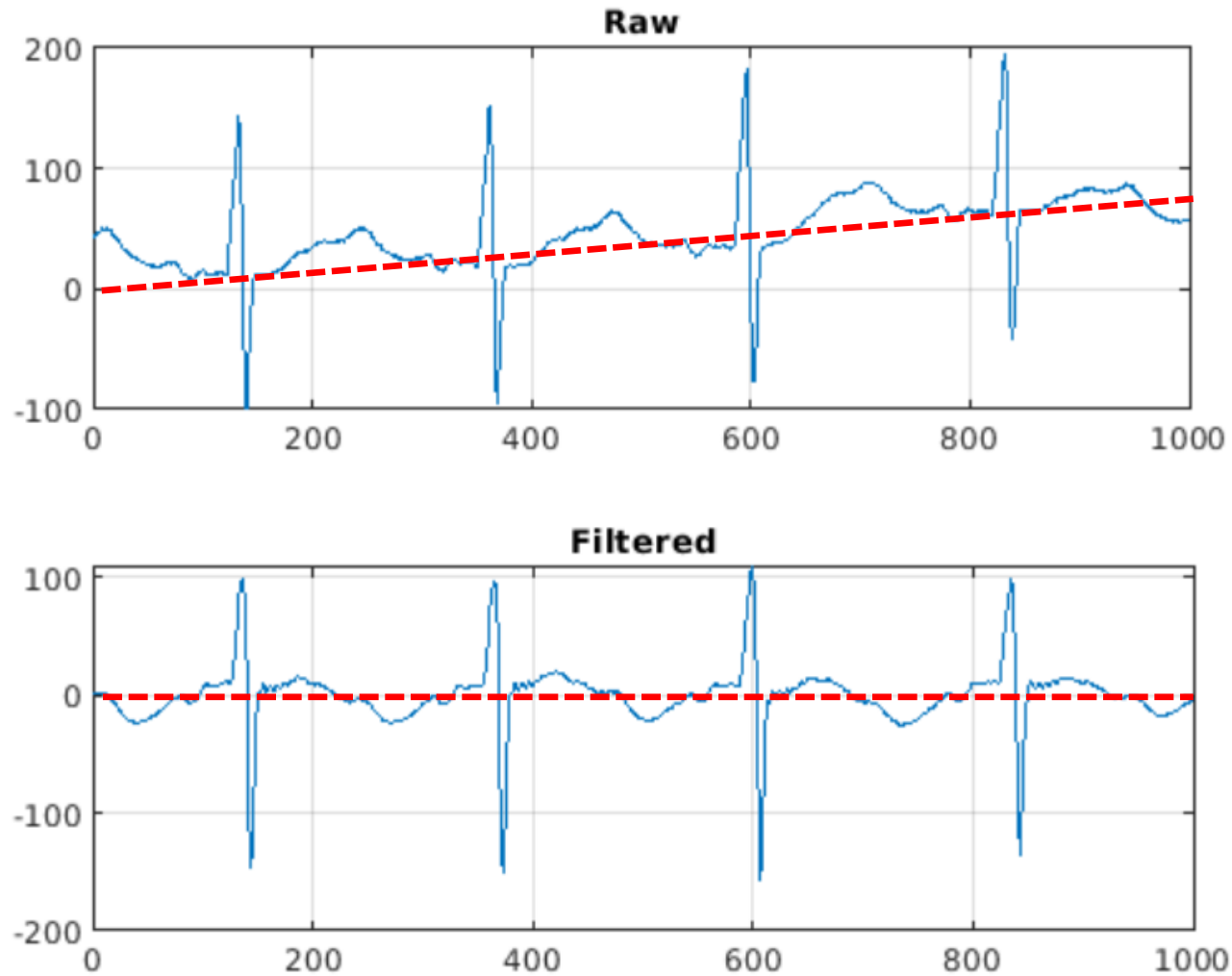
生データを用いても満足する性能は得られない



```
layers = [ ...  
    sequenceInputLayer(1)  
    lstmLayer(200, 'OutputMode', 'sequence')  
    fullyConnectedLayer(4)  
    softmaxLayer  
    classificationLayer];
```

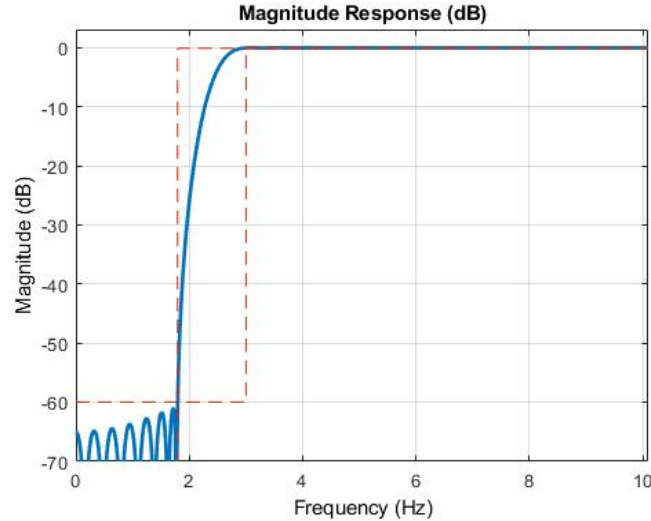
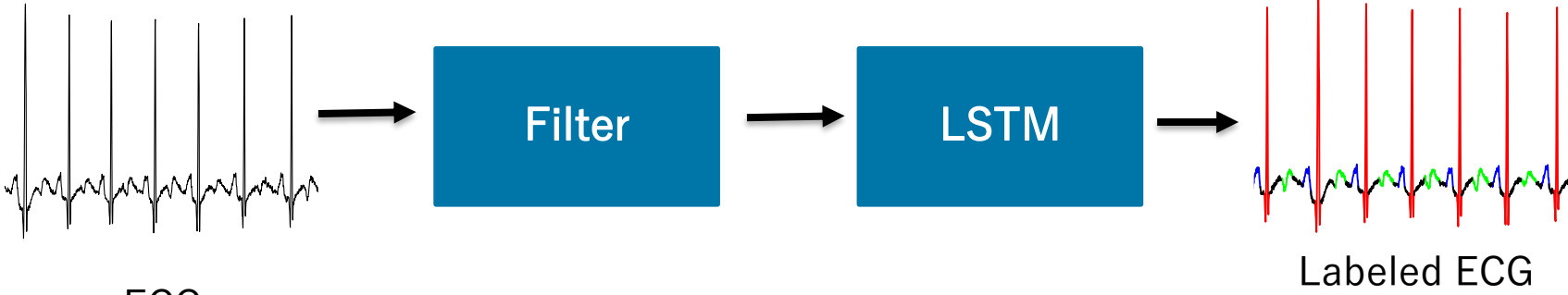
True Class	P	QRS	T	n/a
P	37.4%	2.3%	1.1%	2.1%
QRS	4.1%	61.4%	0.6%	4.3%
T	2.5%	1.4%	58.7%	7.3%
n/a	56.0%	34.8%	39.6%	86.2%
Predicted Class				

信号の”質”を改善



必要な情報だけを
ネットワークに学習させる！

信号の”質”を改善



Filter baseline wander

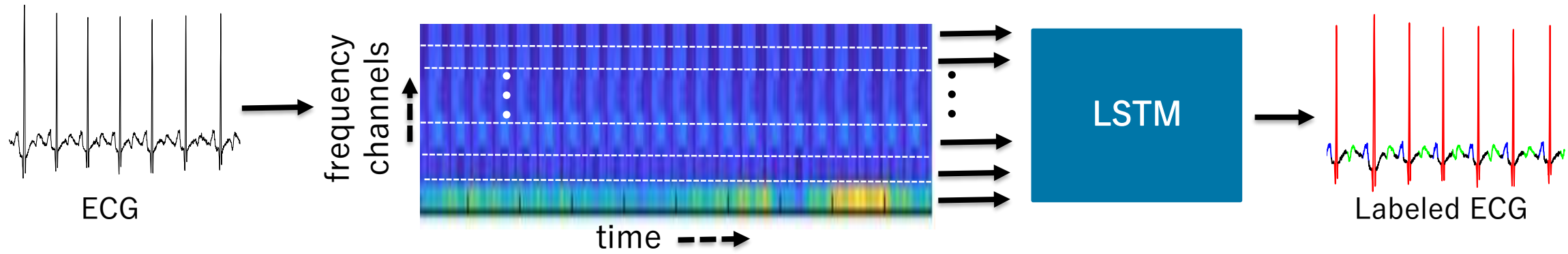
Raw signals

True Class \ Predicted Class	P	QRS	T	n/a
P	37.4%	2.3%	1.1%	2.1%
QRS	4.1%	61.4%	0.6%	4.3%
T	2.5%	1.4%	58.7%	7.3%
n/a	56.0%	34.8%	39.6%	86.2%

Filtered signals

True Class \ Predicted Class	P	QRS	T	n/a
P	45.7%	3.5%	0.3%	5.2%
QRS	3.5%	73.0%	0.9%	6.0%
T	2.6%	0.7%	74.6%	9.0%
n/a	48.3%	22.8%	24.2%	79.8%

信号の特徴が、より際立つように変換処理を加える



信号の特徴が、より際立つように変換処理を加える

源信号（未処理）

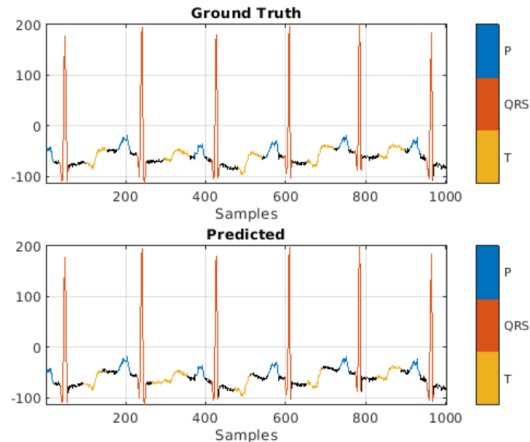
True Class	P	QRS	T	n/a
P	37.4%	2.3%	1.1%	2.1%
QRS	4.1%	61.4%	0.6%	4.3%
T	2.5%	1.4%	58.7%	7.3%
n/a	56.0%	34.8%	39.6%	86.2%
Predicted Class				

フィルタ後の信号

True Class	P	QRS	T	n/a
P	45.7%	3.5%	0.3%	5.2%
QRS	3.5%	73.0%	0.9%	6.0%
T	2.6%	0.7%	74.6%	9.0%
n/a	48.3%	22.8%	24.2%	79.8%
Predicted Class				

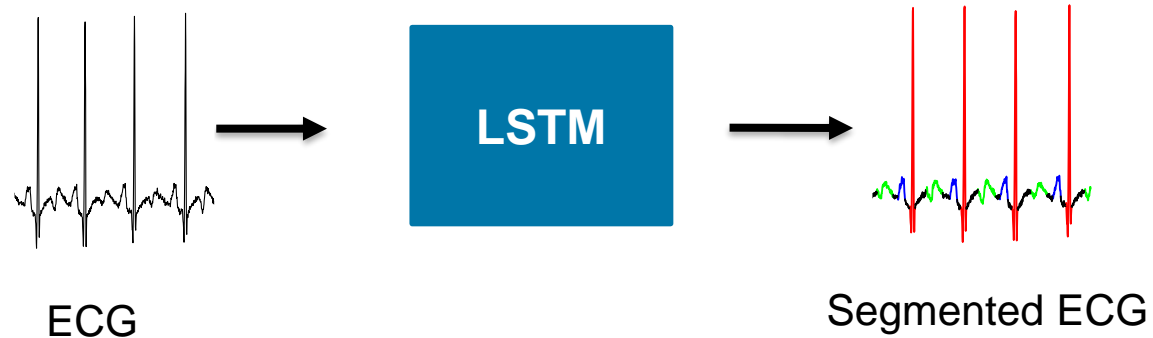
FSST

True Class	P	QRS	T	n/a
P	80.5%	0.4%	0.3%	3.2%
QRS	0.7%	90.7%	0.3%	2.1%
T	1.0%	0.3%	82.2%	7.7%
n/a	17.8%	8.7%	17.2%	87.1%
Predicted Class				

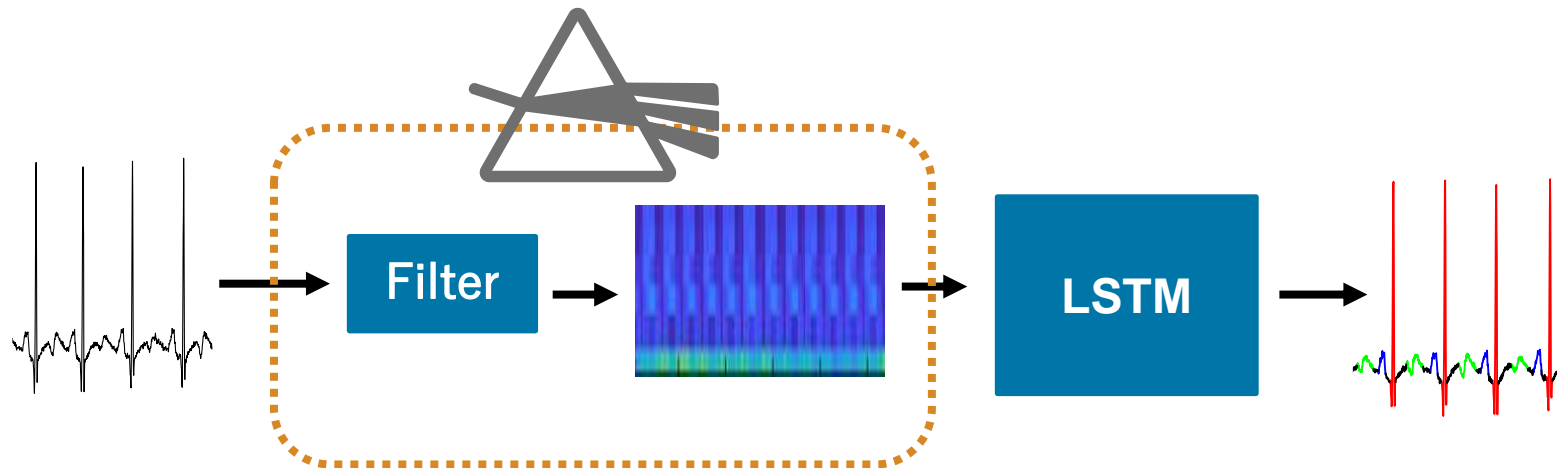


Region	Raw Signals Accuracy (%)	FSST Accuracy (%)
P	37.4	80.5
QRS	61.4	90.7
T	58.7	82.2

特徴抽出により、シンプルなAIモデルでも精度は上がる

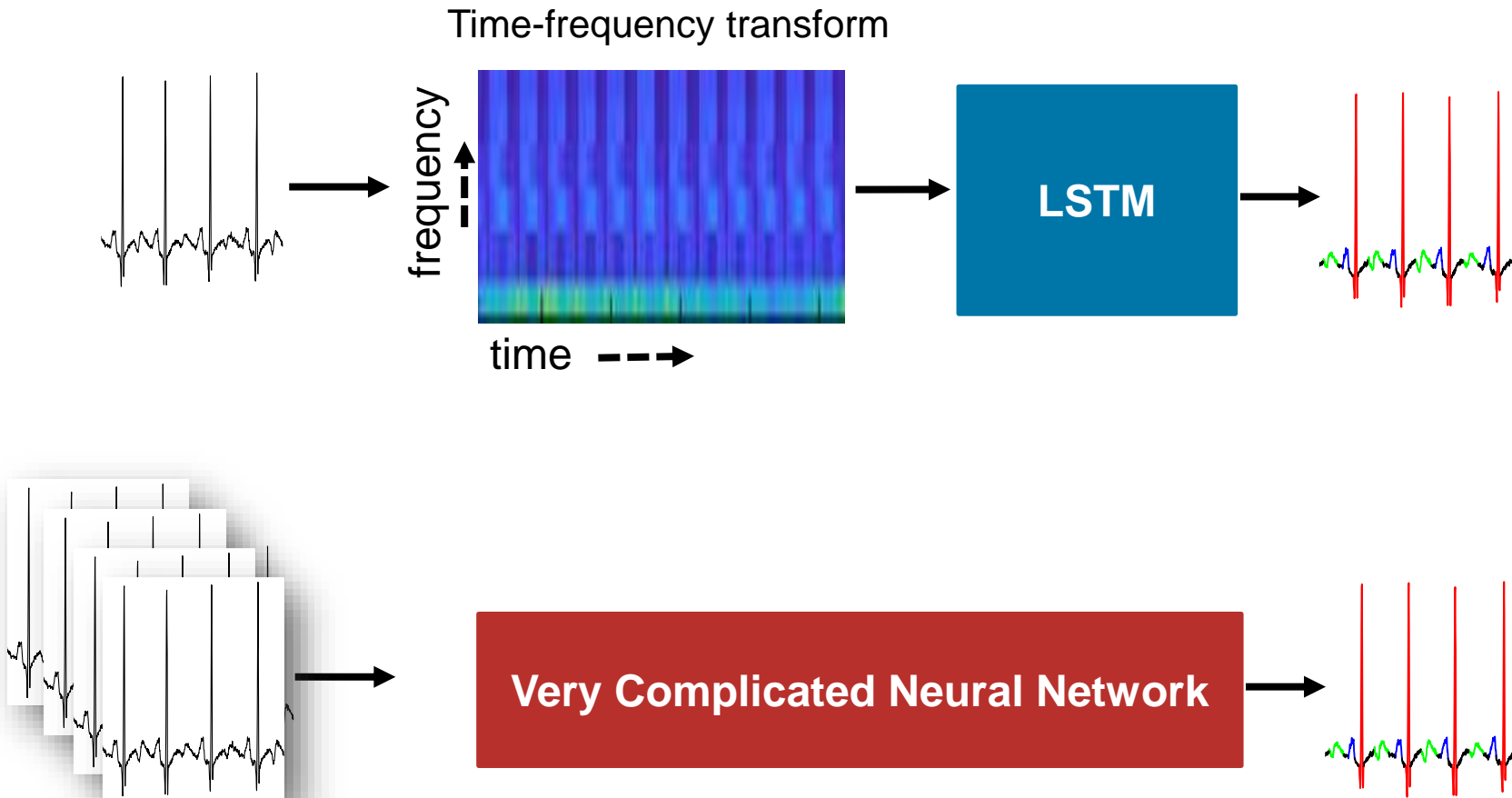


True Class	P	QRS	T	n/a
P	37.4%	2.3%	1.1%	2.1%
QRS	4.1%	61.4%	0.6%	4.3%
T	2.5%	1.4%	58.7%	7.3%
n/a	56.0%	34.8%	39.6%	86.2%
Predicted Class				



True Class	P	QRS	T	n/a
P	80.5%	0.4%	0.3%	3.2%
QRS	0.7%	90.7%	0.3%	2.1%
T	1.0%	0.3%	82.2%	7.7%
n/a	17.8%	8.7%	17.2%	87.1%
Predicted Class				

(逆に) 特徴抽出により、モデルの複雑さを軽減



True Class	P	QRS	T	n/a
QRS	80.5%	0.4%	0.3%	3.2%
T	0.7%	90.7%	0.3%	2.1%
n/a	1.0%	0.3%	82.2%	7.7%
Predicted Class				

推論ロジックがコンパクトになれば…
⇒実装コストの低減

MATLABが選ばれる理由

1. データセットの置き換えですぐ試せる
2. 具体的かつ実践的な例題
3. 便利なアプリの活用

信号アナライザー（センサーデータの可視化、解析）

MATLAB R2022a

ホーム プロット アプリ

ドキュメンテーションの検索 Takemotoさん

アプリの設計 さらにアプリを取得 アプリのインストール アプリのパッケージ化

ファイル

曲線フィッター 信号アナライザー Audio Test Bench フィルターデザイナー 信号ラベラー 信号多重解像度アナライザー 診断特徴デザイナー Sensor Array Analyzer 分類学習器 Raspberry Pi リソース モニター ディープ ネットワーク デザイナー

アプリ

E:\mwork2\mydemo_app\PdM\RasPiFan\modified22av1

現在のフォルダー

名前	更新...	サイズ	タ...
slprj	2022...		フォル...
step1_daq_ert_rtw	2022...		フォル...
step2_simplemodel_ert_r.2022...	2022...		フォル...
step3_MLmodel_ert_rtw	2022...		フォル...
feature_for_simulink_test..2022...	2022...	2 KB	エディ...
step1_daq.elf	2022...	48 KB	ELF ...
step2_simplemodel.elf	2022...	406 ...	ELF ...
step3_MLmodel.elf	2022...	498 ...	ELF ...
feature_for_simulink_test..2022...	2022...	3 KB	関数
feature_for_simulink_test..2022...	2022...	2 KB	関数
feature_for_simulink_test..2022...	2022...	2 KB	関数
raspberrypi_pdm_extract..2022...	2022...	4 KB	関数
ClassificationLearnerSess..2022...	2022...	1.66 ...	MAT ...
DFD_App_SessionData....2022...	2022...	281 ...	MAT ...
sigana_ONOFF.mat	2022...	11.1...	MAT ...
sigana_ONOFFNG.mat	2022...	11.6...	MAT ...
step1_rawdata.mat	2022...	11 KB	MAT ...
step2_rawdata.mat	2022...	17 KB	MAT ...
step2_rawdata_old.mat	2022...	18 KB	MAT ...

step1_rawdata.mat (MAT ファイル)

ワークスペース

名前	値
raw_sensor_data_off	1001x3 double
raw_sensor_data_on	1001x3 double

コマンド ウィンドウ

```
fx >>
```

信号ラベラー（ラベリング作業の効率化）

MathWorks Signal Labeler interface showing the workflow for labeling signals.

Top Panel: LABEL, DISPLAY, TIME tabs. Includes buttons for Add Definition, ROI, Description, Value (LFM), Label, AUTOMATE VALUE, Save Labels, and Cancel.

Label Definitions: SignalSource, WaveformType (selected).

Labeled Signal Set:

Name	Plot	Value	Location (Min)	Location (Max)
▼ X				
X(:,1)	<input checked="" type="checkbox"/>			
X(:,2)	<input type="checkbox"/>			
SignalSource		Receiver1		
WaveformT...				

Signal Plots:

Top plot: X(:,1) vs Samples (0 to 6000). The signal shows a burst of high-frequency noise between 1000 and 2000 samples, and another burst between 3000 and 4000 samples.

Bottom plot: WaveformType vs Samples (0 to 6000). The plot shows a horizontal line at 0, indicating no signal is currently labeled.

Legend:

SignalSource
Receiver1

診断特徴デザイナー/分類学習器（特徴抽出、ランキング、学習）

MATLAB R2022a

ホーム プロット アプリ

アプリの設計 さらにアプリを取得 アプリのインストール アプリのパッケージ化

ファイル

アプリ

曲線フィッター 信号アナライザー Audio Test Bench フィルターデザイナー 信号ラベラー 信号多重解像度アナライザー 診断特徴デザイナー Sensor Array Analyzer 分類学習器 Raspberry Pi リソース モニター ディープ ネットワーク デザイナー

ドキュメンテーションの検索 Takemotoさん

E:\mwork2\mydemo_app\PdM\RasPiFan\modified22av2

現在のフォルダー

名前	更新日	サイズ	タイプ
slprj	2022/05/12 13...		フォルダー
step1_daq_ert_rtw	2022/05/16 13...		フォルダー
step2_simplemodel_e...	2022/05/16 13...		フォルダー
step3_MLmodel_ert_rtv	2022/05/12 14...		フォルダー
untitled_ert_rtw	2022/05/12 13...		フォルダー
crestfactor_shapafact...	2022/05/12 18...	1 KB	スクリプト
sense_hat_test.m	2022/05/16 10...	1 KB	スクリプト
step2_rawdata.mat	2022/05/12 14...	28 KB	MAT ファイル
trainedModel_svm.mat	2022/05/12 14...	6 KB	MAT ファイル
data_prep.mlx	2022/05/16 17...	4 KB	ライブ スクリプト
step3_MLmodel.slx.ori.	2022/05/11 9...	51 KB	ORIGINAL ファイル

data_prep.mlx (ライブ スクリプト)

ワークスペース

名前	値
raw_sensor_data_ng	1001x3 double
raw_sensor_data_off	1001x3 double
raw_sensor_data_on	1001x3 double
raw_sensor_data_on1	1001x3 double
raw_sensor_data_on2	1001x3 double
tt	3x4 table
ttx	3x1 table
tty	3x1 table
ttz	3x1 table

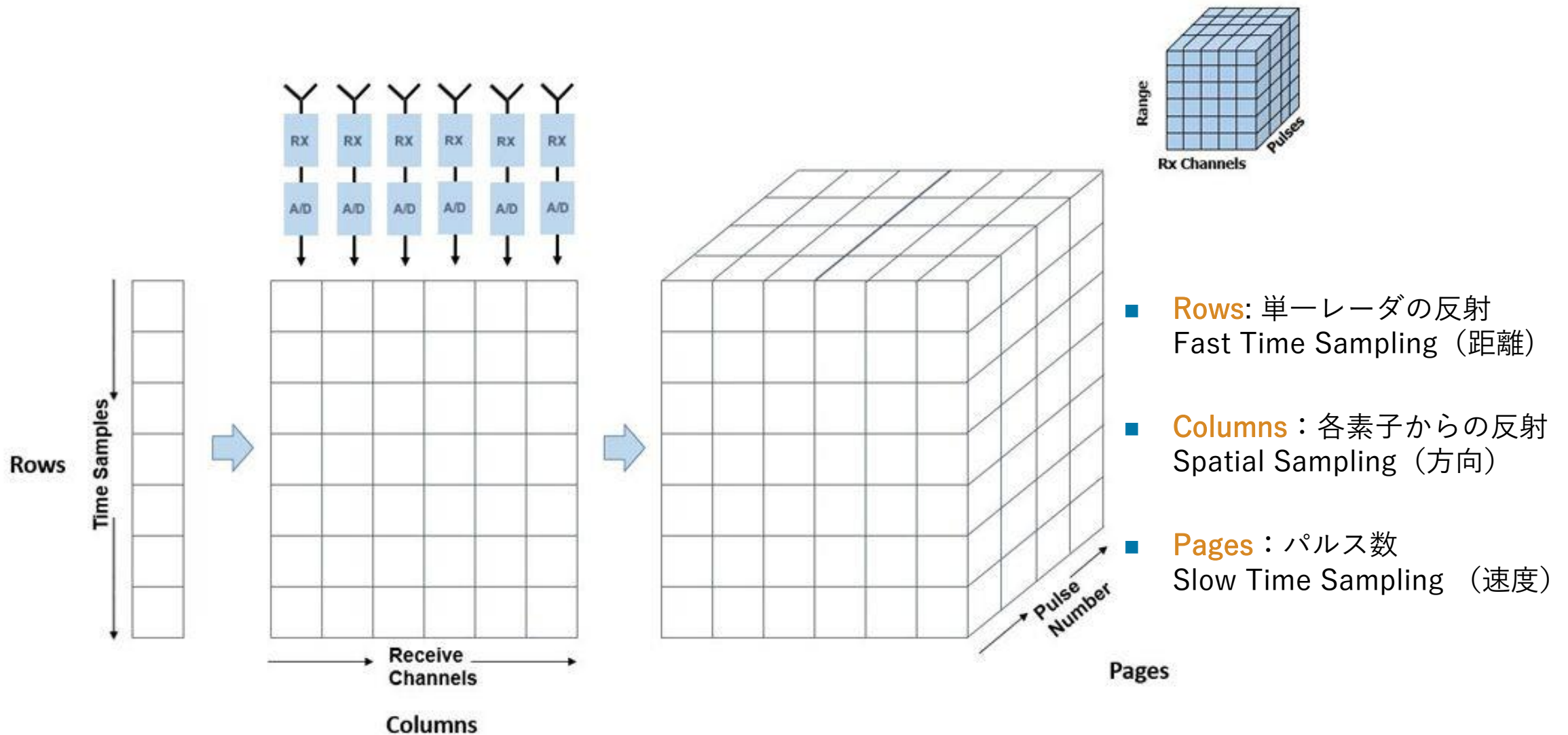
コマンド ウィンドウ

fx >>

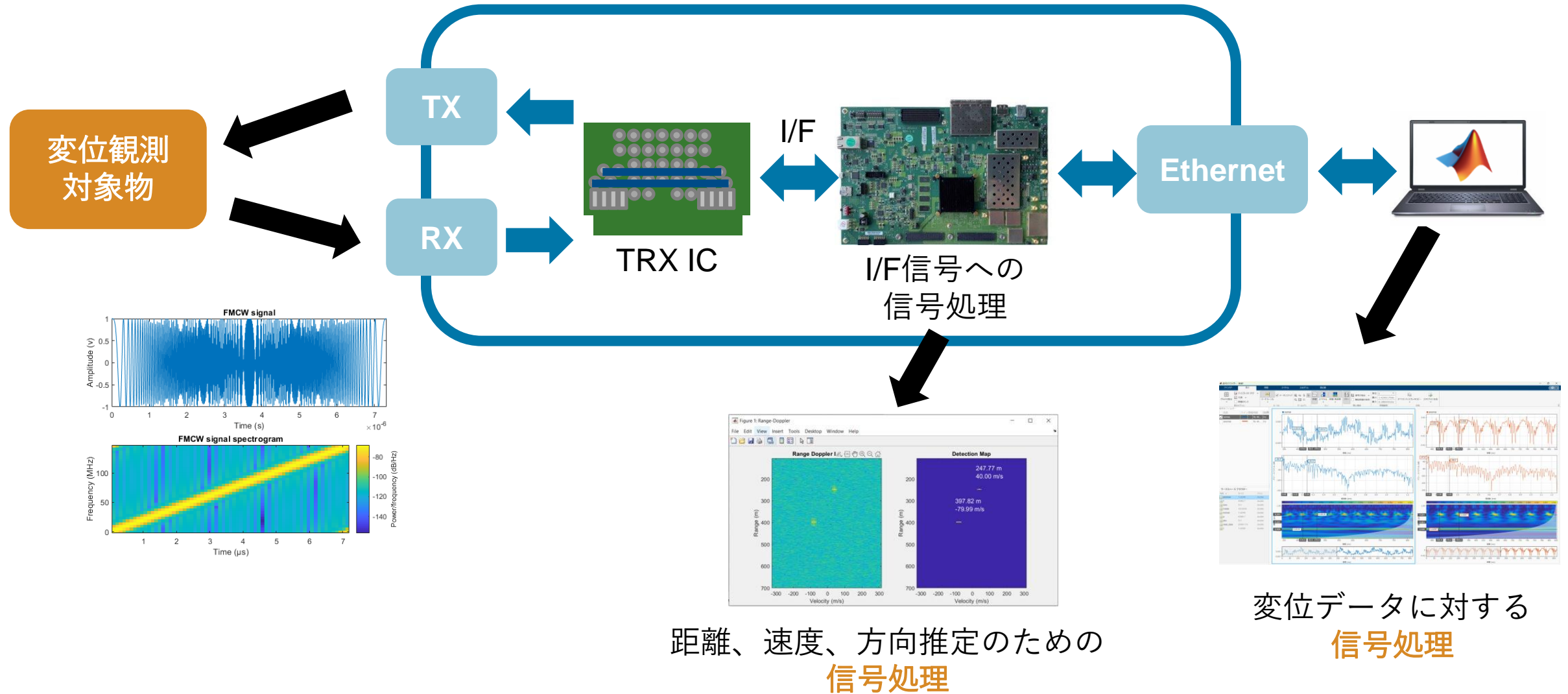
Agenda

1. なぜ、信号処理が重要なのか？
2. 信号処理ツールとしてMATLABが選ばれてきた理由
3. 信号処理の各種例題のご紹介

レーダー信号の概要



レーダーシステムにおける信号処理（受信処理、後処理）

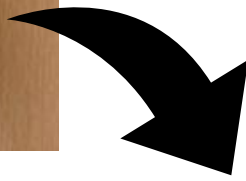


変位データによる異常判別（実験概要）

- モーター仕様
 - メーカー: オリエンタルモーター社
 - AC コントローラ: US206-001
 - 回転速度: 20rpm
- レーダー仕様
 - 79-81GHz FMCW レーダー (ADAR6902)
 - FMCW ramp-to-ramp period: 600 μ s (1667Hz)
 - Burst ramps: 2048
- モーター偏心 (次ページ)
 - Imbalance lv0: Normal
 - Imbalance lv1: 1 screw
 - Imbalance lv2: 1 screw + 1 nut
 - Imbalance lv3: 1 screw + 2 nuts
 - Imbalance lv4: 1 screw + 3 nuts
 - Imbalance lv5: 1 screw + 4nuts

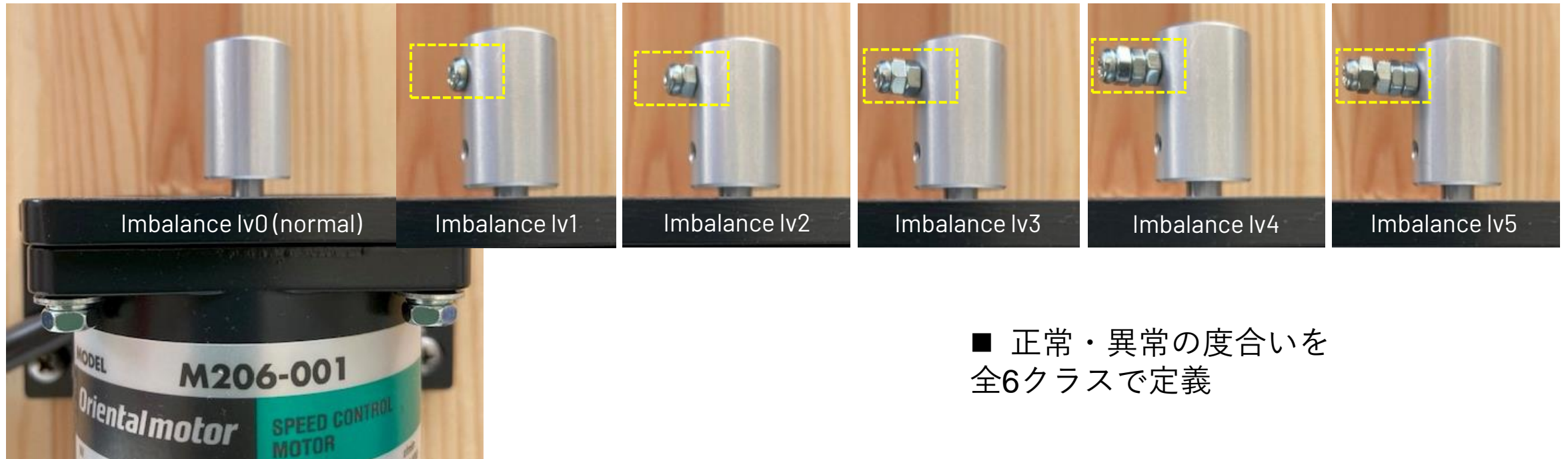


正常・異常（偏心）のレベル



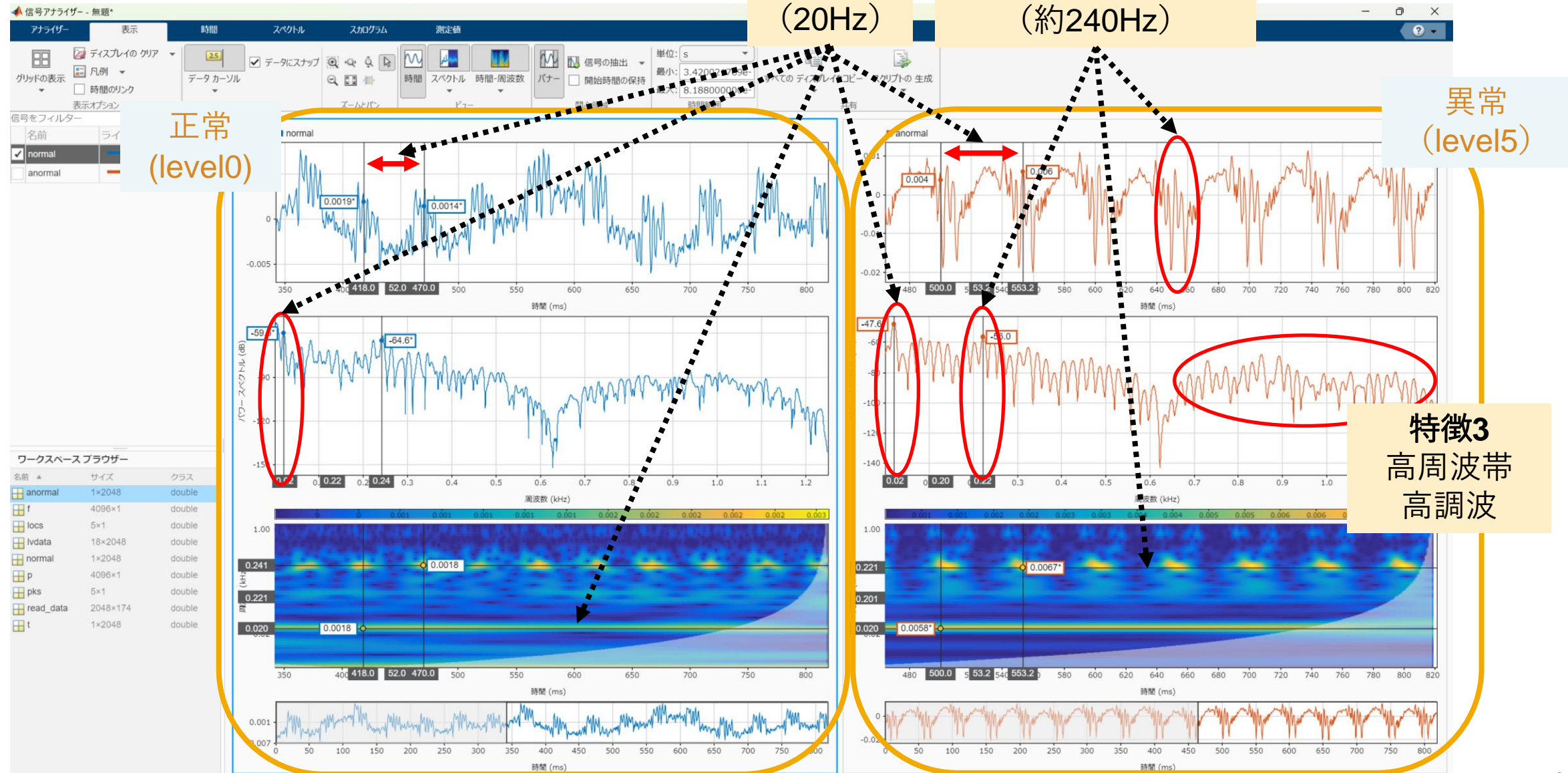
normal

anormal



■ 正常・異常の度合いを
全6クラスで定義

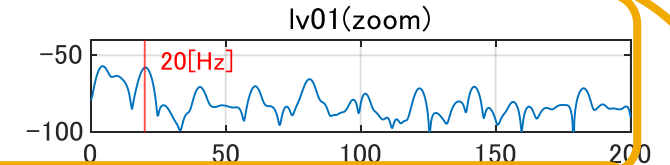
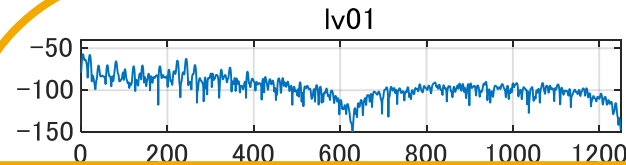
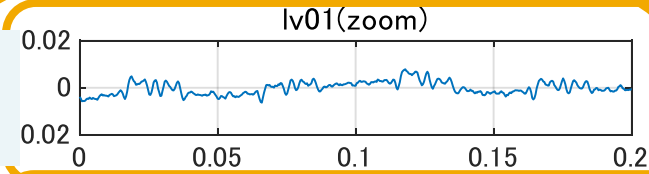
信号アナライザによる特徴の吟味



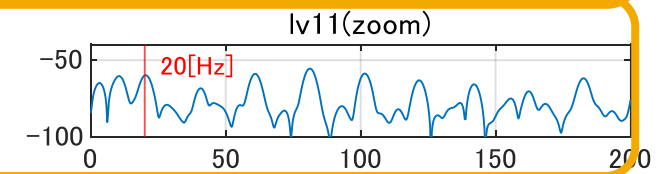
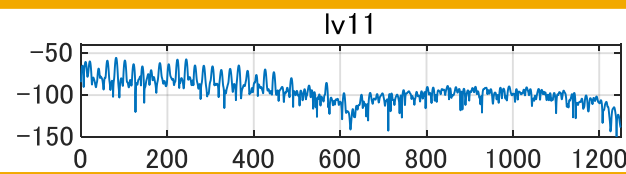
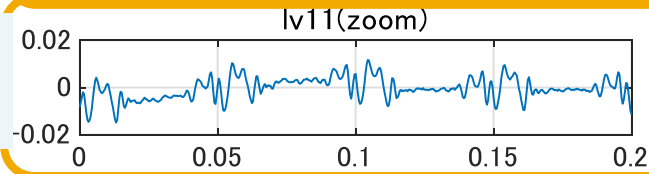
6クラスの波形比較（時間軸、周波数軸）

低異常度

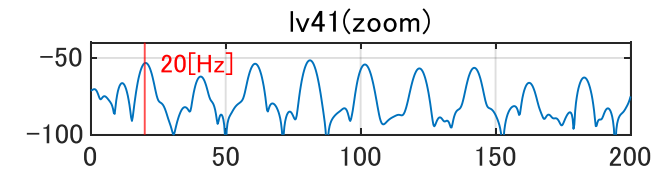
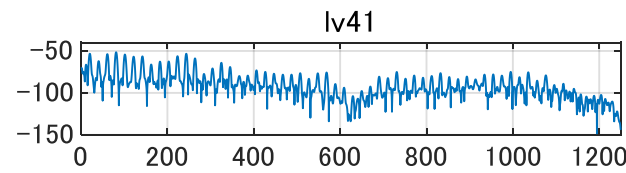
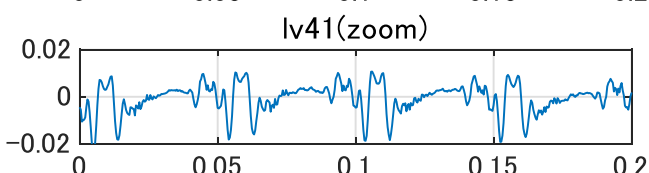
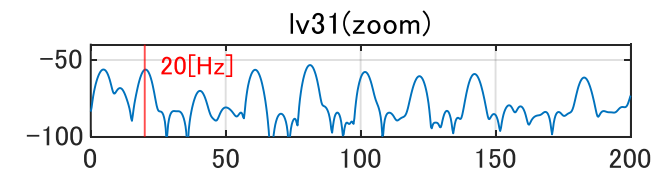
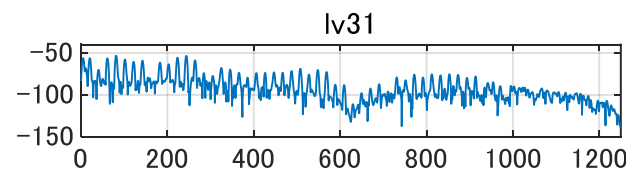
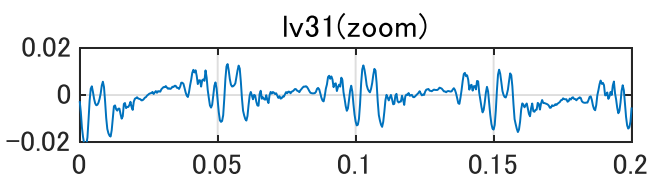
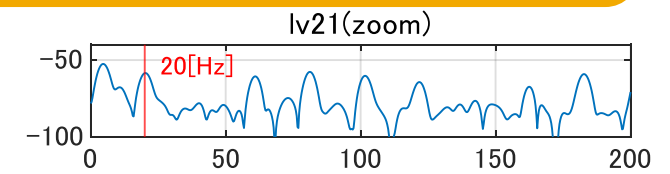
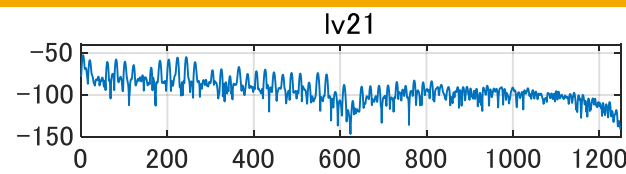
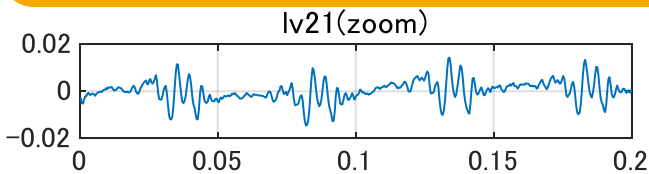
偏心無し
level0



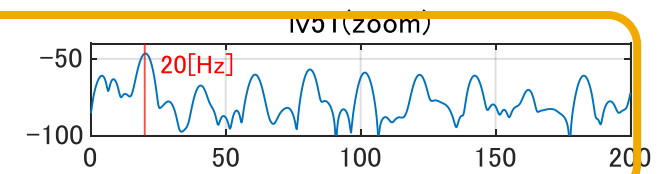
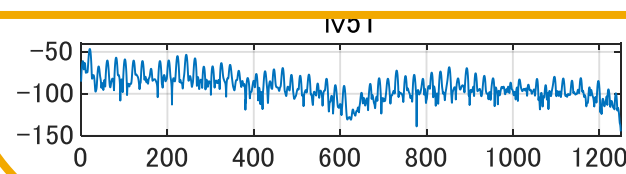
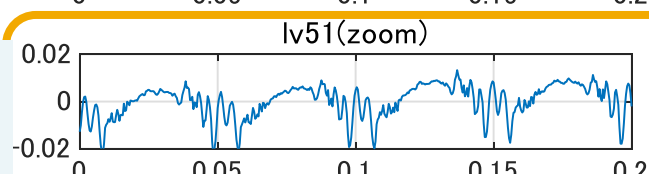
偏心
level1



⋮



偏心
level5

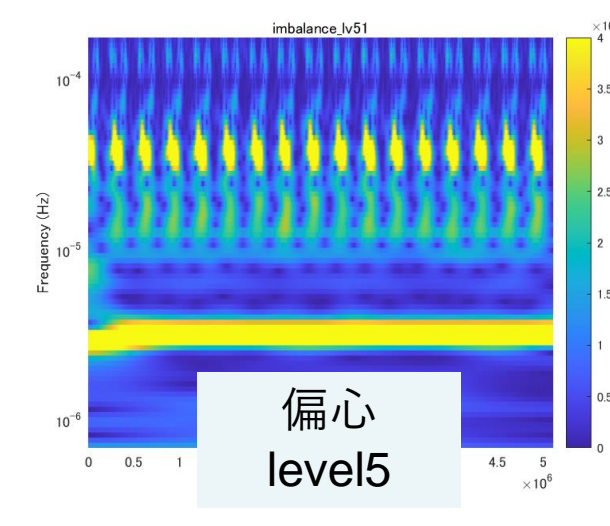
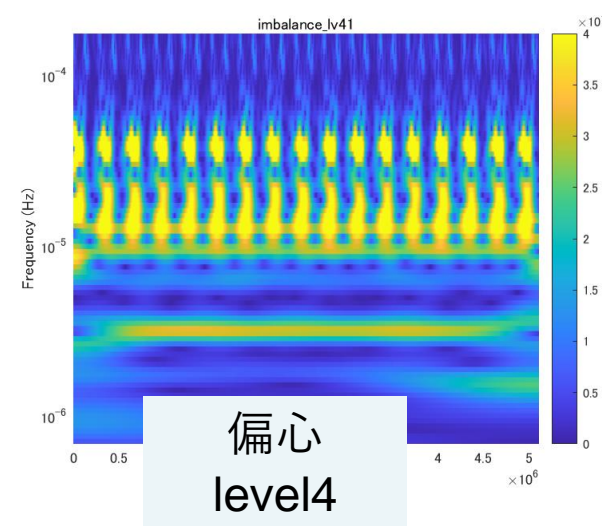
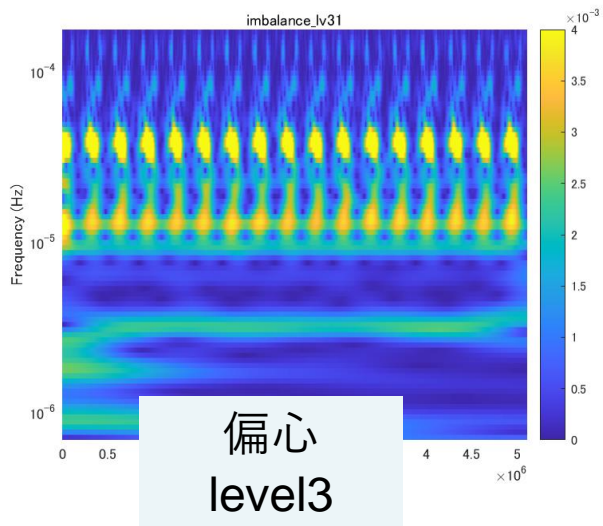
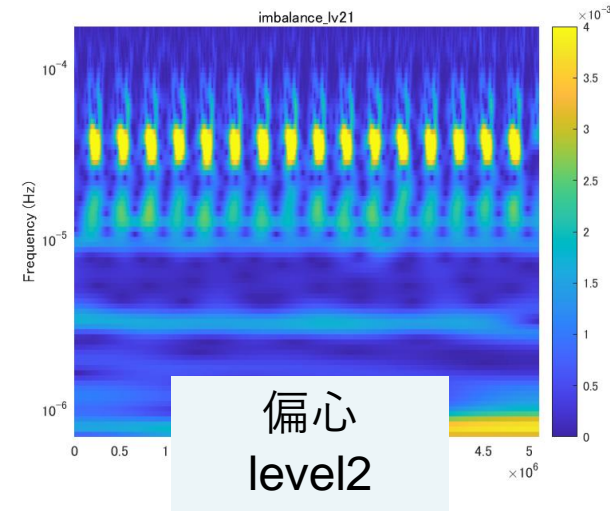
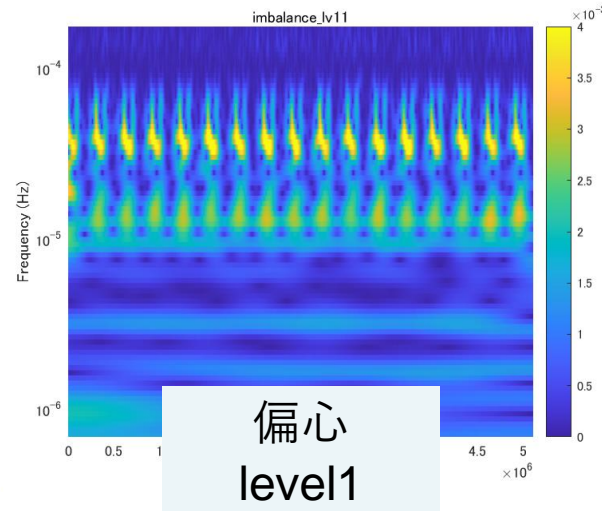
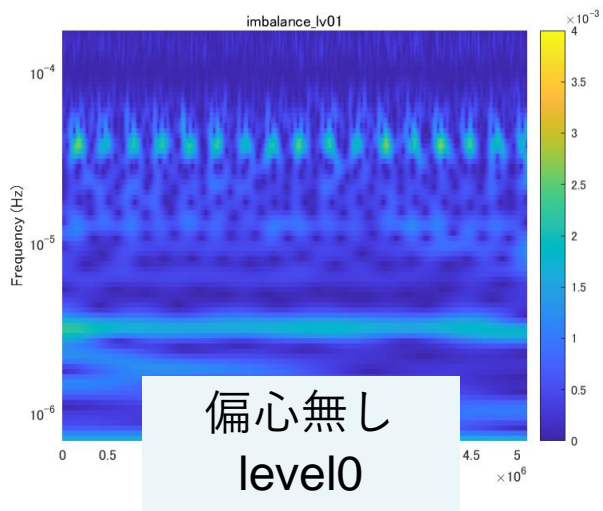


時間軸

周波数軸

高異常度

時間-周波数特性（ウェーブレット）



各レベルで
特徴に違いあり



異常度の増加に
伴う単調増加
とは限らない



単純な閾値判定
では判別は困難



ディープラーニン
グの得意分野

学習用データセットの用意

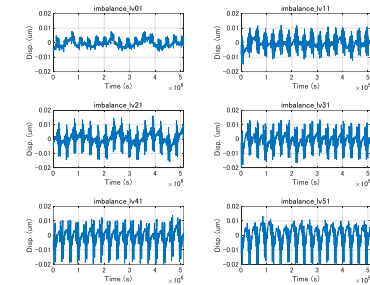
CSVファイルの読み込み

readmatrix

	FI	FJ	FK	FL	FM	FN	FO	FP	FQ	FR	FS
2342											
2343	-0.00507	-0.00239	-0.00394	0.001004	0.001955	0.001602	-0.003	-0.00173	-0.0036	0.000087	
2344	-0.00562	-0.00307	-0.00491	0.000876	0.001992	0.000628	-0.00365	-0.00228	-0.00412	-0.0006	
2345	-0.00699	-0.00526	-0.00619	-5.7E-05	0.000743	-0.00105	-0.00542	-0.00394	-0.00529	-0.00109	
2346	-0.0068	-0.00443	-0.00406	0.001695	0.000767	-0.00096	-0.00512	-0.00358	-0.0045	-0.00045	
2347	-0.00673	-0.00417	-0.00167	0.002174	-0.00014	-0.00147	-0.00508	-0.00404	-0.00464	-0.00048	
2348	-0.00561	-0.00416	0.000621	0.002667	-0.00154	-0.00052	-0.00459	-0.00416	-0.00482	0.000742	

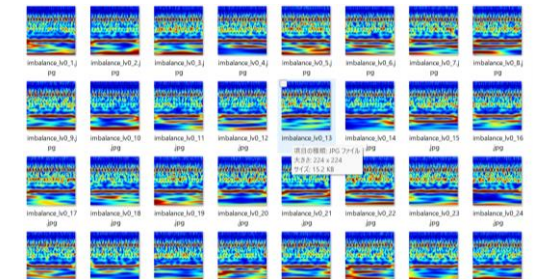
変位データ（時系列データ）の切り出し

- サンプリング周波数：400μs
- データ点数：1024点
- データの種類：6種類（正常:1、異常:5）
- ファイル数：3132



全データについて
ウェーブレット変換後
.jpg形式でクラス別に保存

- jpegフォーマットとして保存
- ディレクトリ名: imbalance_lv* (*はレベル0~5)
- ファイル数は各クラス同数 (3132/6 = 522)



imageDatastore定義後、
学習用、検証用、テスト用に分割

- 学習用：2190 (3132の約70%)
- 検証用：630 (3132の約20%)
- テスト用：312 (3132の約10%)

```

imageDatastore('imbalance_lv*_cwt',
    'imgsTest', 1x1 ImageDatastore
    'imgsTrain', 1x1 ImageDatastore
    'imgsValidation', 1x1 ImageDatastore
)

```


学習パラメータの設定、学習

```
net = googlenet;
```

```
lgraph = layerGraph(net);
numberOfLayers = numel(lgraph.Layers);
```

```
newDropoutLayer = dropoutLayer(0.6,'Name','new_Dropout'); % 過適合対策
lgraph = replaceLayer(lgraph,'pool5-drop_7x7_s1',newDropoutLayer);
```

```
newConnectedLayer = fullyConnectedLayer(6,'Name','new_fc',... % 分類数で置き換え
'WeightLearnRateFactor',5,'BiasLearnRateFactor',5);
lgraph = replaceLayer(lgraph,'loss3-classifier',newConnectedLayer);
```

```
newClassLayer = classificationLayer('Name','new_classoutput');
lgraph = replaceLayer(lgraph,'output',newClassLayer);
```

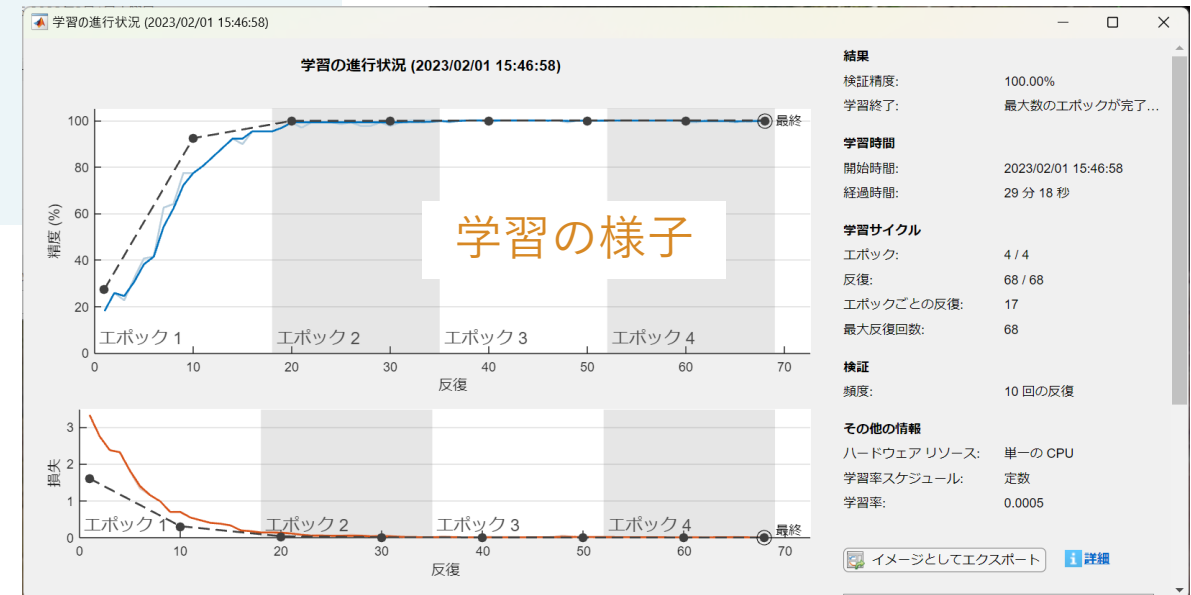
GoogLeNetの編集

```
options = trainingOptions('sgdm',...
'MiniBatchSize',128,...
'MaxEpochs',4,...
'InitialLearnRate',5e-4,...
'ValidationData',imgsValidation,...
'ValidationFrequency',10,...
'Verbose',1,...
'ExecutionEnvironment','cpu',...
'Plots','training-progress');
```

学習オプション

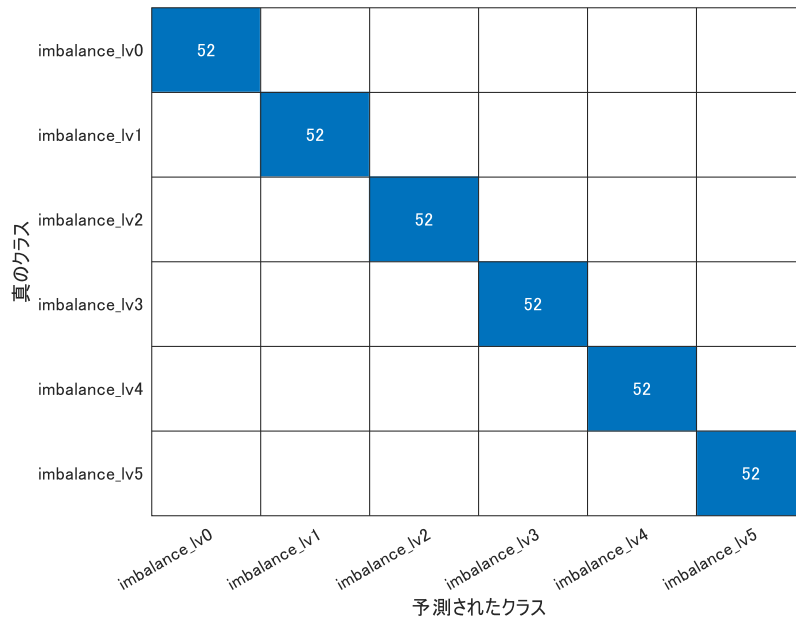
GoogLeNetの再学習

```
>>trainedGN = trainNetwork(imgsTrain,lgraph,options);
```

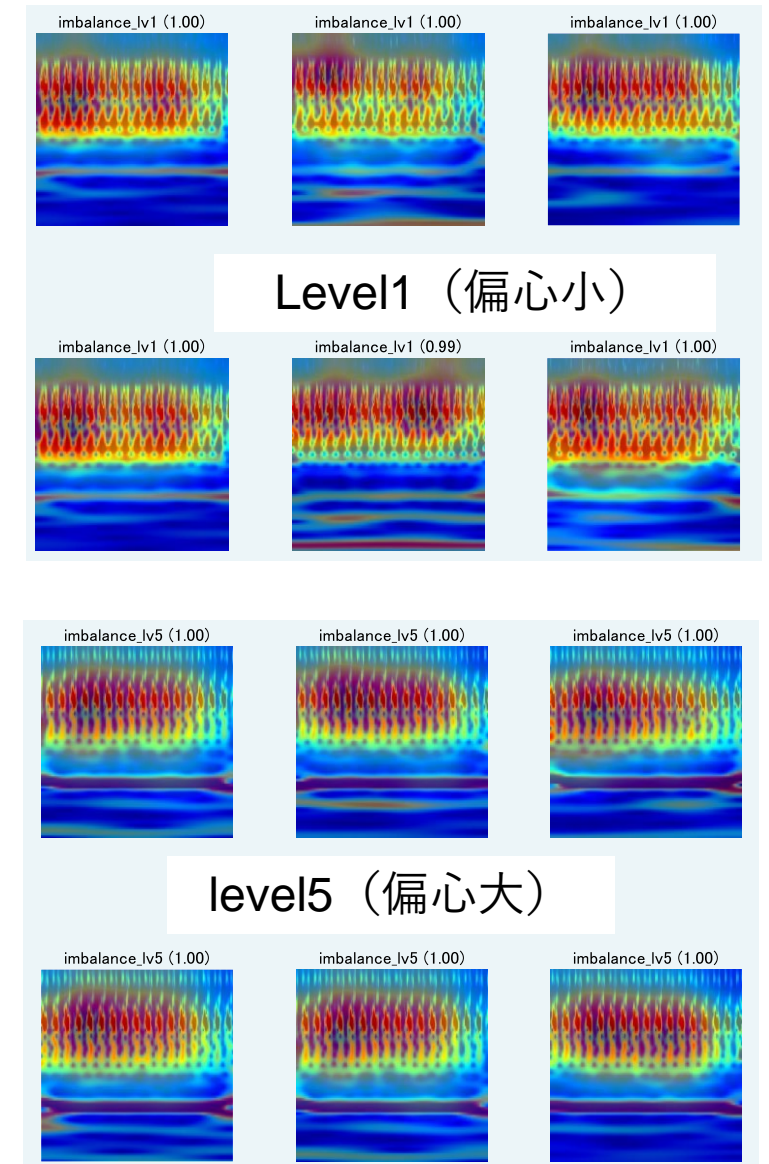
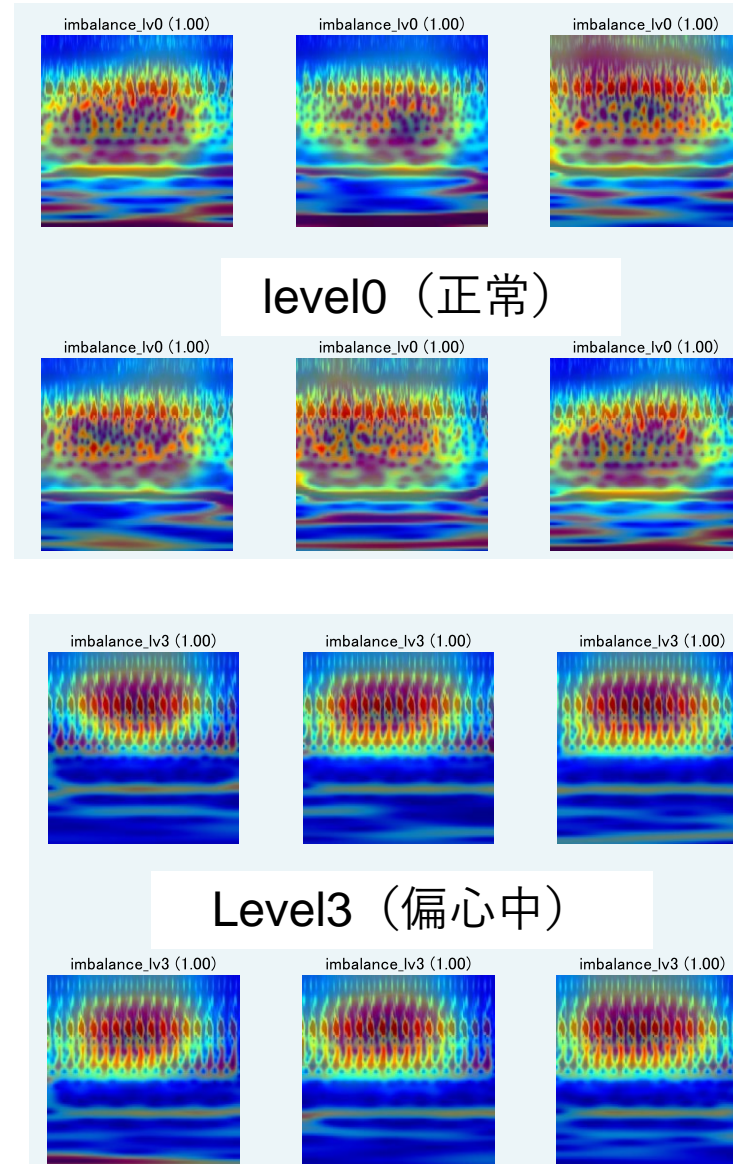


混同行列、Grad-CAMによる検証

混同行列



Grad-CAM



AI 関連例題

51 Vertical Application Examples

アプリケーション	Biosignals	2	2	1	12	8	2	4
	Radar/Comms	2	1	3	1	2	1	1
	Vibrations	4		4	3	3	2	
	Power Systems	3		3	1	1		
	Speech & Audio	2	2		5	3	1	
	Predictive Maintenance*			1	1	1		
	Earth & Ocean	1	1					
	Auto				1	1		
	Finance	1						
* excl. Predictive Maintenance Tbx examples		Analyze	Label	Synthesize	Engineer Features	Classify (AI)	Regress (AI)	Deploy

- Analyze : データアクセス、データ管理、可視化、データクレンジング
- Label : 信号のラベリング (マニュアルおよび自動ラベリング)
- Synthesize : 信号生成、データの水増し、GANによる信号合成
- Deploy : CPUおよびGPU向け自動コード生成

まとめ：信号処理・AIシステム開発のワークフロー（再掲）

1. データセットの作成

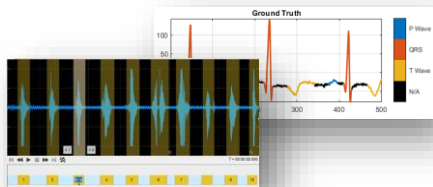
■ データソース



■ シミュレーション ■ オグメンテーション

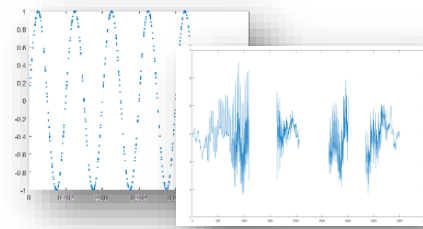


■ データラベリング

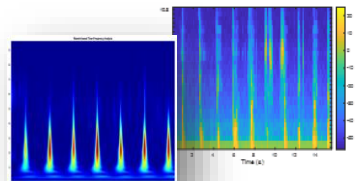


2. 前処理と変換

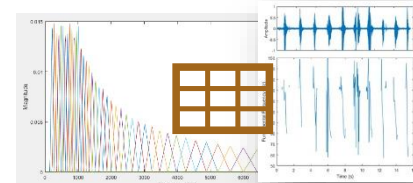
■ 前処理



■ 変換



■ 特徴抽出

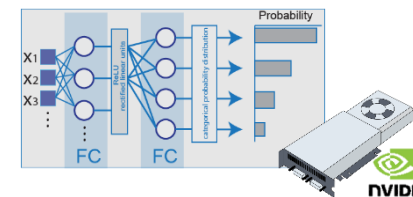


3. 予測モデルの開発

■ リファレンスモデルの流用 ■ オリジナルモデルの作成



■ GPUによる高速な学習

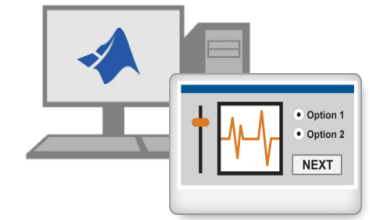


■ ハイパーパラメータの 解析とチューニング



4. 実システムへの展開

■ デスクトップアプリ



■ エンタープライズシステム

Java
MATLAB
C/C++
Python

■ エッジ（組み込み）デバイス

