plotTDKCharDataset

Explore TDK TAS2141 characterization dataset and plot its content.

Syntax

`plotTDKCharDataset()`

Description

plotTDKCharDataset() explores the dataset and plot its content in three docked figure windows. Loads dataset location from config.mat.

Examples

`plotTDKCharDataset();`

Input Arguments

None

Output Arguments

None

Requirements

- Other m-files: none
- Subfunctions: none
- MAT-files required: data/TDK_TAS2141_Characterization_2020-10-22_18-12-16-827.mat, data/config.mat

See Also

- plot
- imagesc
- polarplot

Created on October 24. 2020 by Tobias Wulf. Copyright Tobias Wulf 2020.

```
function plotTDKCharDataset()
    try
        % load dataset path and dataset content into function workspace
        load('config.mat', 'PathVariables');
        load(PathVariables.tdkDatasetPath, 'Data', 'Info');
%         close all;
    catch ME
        rethrow(ME)
    end

    % figure save path for different formats
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    fig1Filename = 'tdk_magnetic_stimulus';
    fig1Path = fullfile(PathVariables.saveImagesPath, fig1Filename);

    fig2Filename = 'tdk_bridge_charistic';
    fig2Path = fullfile(PathVariables.saveImagesPath, fig2Filename);
```

```
% load needed data from dataset in to local variables for better handling
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% check if modulation fits to following reconstructioning
if ~strcmp("triang", Info.MagneticField.Modulation)
    error("Modulation function is not triang.");
end
if ~(strcmp("cos", Info.MagneticField.CarrierHx) && ...
        strcmp("sin", Info.MagneticField.CarrierHy))
    error("Carrier functions are not cos or sin.");
end

% modulation frequency
fm = Info.MagneticField.ModulationFrequency;
% carrier frequency
fc = Info.MagneticField.CarrierFrequency;
% max and min amplitude
Hmax = Info.MagneticField.MaxAmplitude;
Hmin = Info.MagneticField.MinAmplitude;
% step range or window size for output picking
Hsteps = Info.MagneticField.Steps;
% resoulution of H steps
Hres = Info.MagneticField.Resolution;
% get unit strings from
kApm = Info.Units.MagneticFieldStrength;
Hz = Info.Units.Frequency;
mV = Info.Units.SensorOutputVoltage;

% get dataset infos and format strings to place in figures
% subtitle string for all figures
infoStr = join([Info.SensorManufacturer, Info.Sensor, ...
    Info.SensorTechnology, ...
    Info.SensorType, "Sensor Characterization Dataset."]);
dateStr = join(["Created on", Info.Created, "by", 'Thorben Sch\"uthe', ...
    "and updated on", Info.Edited, "by", Info.Editor + "."]);

% load characterization data
Vcos = Data.SensorOutput.CosinusBridge;
Vsin = Data.SensorOutput.SinusBridge;
gain = Info.SensorOutput.BridgeGain;

% clear dataset all loaded
clear Data Info;
disp('Info:');
disp([infoStr; dateStr]);

% reconstruct magnetic stimulus and reduce the view for example plot by 10
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% number of periods reduced by factor 10
reduced = 10;
nPeriods = fc / fm / reduced;
% number of samples for good looking 40 times nPeriods
nSamples = nPeriods * 400;
% half number of samples
nHalf = round(nSamples / 2);
% generate angle base
phi = linspace(0, nPeriods * 2 * pi, nSamples);
% calculate modulated amplitude, triang returns a column vector, transpose
Hmag = Hmax * triang(nSamples)';
% calculate Hx and Hy stimulus
Hx = Hmag .* cos(phi);
```

```
Hy = Hmag .* sin(phi);
% index for rising and falling stimulus
idxR = 1:nHalf;
idxF = nHalf:nSamples;
% find absolute min and max values in bridge outputs for uniform colormap
A = cat(3, Vcos.Rise, Vcos.Fall, Vcos.All, Vcos.Diff, Vsin.Rise, ...
    Vsin.Fall, Vsin.All, Vsin.Diff);
Vmax = max(A, [], 'all');
Vmin = min(A, [], 'all');
clear A;

% figure 1 magnetic stimulus
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
fig1 = figure('Name', 'Magnetic Stimulus');
tiledlayout(fig1, 2, 2);

% title and description
disp("Title: Magnetic Stimulus Reconstructed H_x-/ H_y-Stimulus" + ...
     "in Reduced View");
disp("Description: Stimulus for characterization in H_x and H_y in " + ...
     "reduced period view by factor 10");
disp(["a) Triangle modulated cosine carrier for H_x stimulus."; ...
      "b) Triangle modulated sine carrier for H_x stimulus."; ...
      "c) Modulation trajectory for rising stimulus"; ...
      "d) Modulation trajectory for falling stimulus"]);

% Hx stimulus
nexttile;
p = plot(phi, Hmag, phi, -Hmag, phi(idxR), Hx(idxR), phi(idxF), Hx(idxF));
set(p, {'Color'}, {'k', 'k', 'b', 'r'}');
legend([p(1) p(3) p(4)], {'mod', 'rise', 'fall'}, 'Location', 'NorthEast');
xticks((0:0.25*pi:2*pi) * nPeriods);
xticklabels({'$0$', '$8\pi$', '$16\pi$', '$24\pi$', '$32\pi$', ...
    '$40\pi$', '$48\pi$', '$56\pi$', '$64\pi$'});
xlim([0 phi(end)]);
ylim([Hmin Hmax]);
xlabel('$\phi$ in rad, Periode $\times 10$');
ylabel(sprintf('$H_x(\\phi)$ in %s', kApm));
title(sprintf('a) $f_m = %1.2f$ %s, $f_c = %1.2f$ %s', fm, Hz, fc, Hz));

% Hy stimulus
nexttile;
p = plot(phi, Hmag, phi, -Hmag, phi(idxR), Hy(idxR), phi(idxF), Hy(idxF));
set(p, {'Color'}, {'k', 'k', 'b', 'r'}');
legend([p(1) p(3) p(4)], {'mod', 'rise', 'fall'}, 'Location', 'NorthEast');
xticks((0:0.25*pi:2*pi) * nPeriods);
xticklabels({'$0$', '$8\pi$', '$16\pi$', '$24\pi$', '$32\pi$', ...
    '$40\pi$', '$48\pi$', '$56\pi$', '$64\pi$'});
xlim([0 phi(end)]);
ylim([Hmin Hmax]);
xlabel('$\phi$ in rad, Periode $\times 10$');
ylabel(sprintf('$H_y(\\phi)$ in %s', kApm));
title(sprintf('b) $f_m = %1.2f$ %s, $f_c = %1.2f$ %s', fm, Hz, fc, Hz));

% polar for rising modulation
nexttile;
polarplot(phi(idxR), Hmag(idxR), 'b');
p = gca;
p.ThetaAxisUnits = 'radians';
title('c) $|\vec{H}(\phi)| \cdot e^{j\phi}$,  $0<\phi<320\pi$');
```

```
% polar for rising modulation
nexttile;
polarplot(phi(idxF), Hmag(idxF), 'r');
p = gca;
p.ThetaAxisUnits = 'radians';
title('d) $|\vec{H}(\phi)| \cdot e^{j\phi}$,  $320<\phi<640\pi$');

% figure 2 cosinus bridge outputs
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
fig2 = figure('Name', 'Cosine and Sine Bridge', 'Position', [0 0 33 30]);

tiledlayout(fig2, 2, 2);

% title and description
disp("Title: Cosine and Sine Bridge. Measured Bridge Outputs" + ...
     " of Corresponding H_x-/ H_y-Amplitudes");
disp("Description: " + sprintf("H_x, H_y in %s, %d Steps in %.4f %s", ...
    kApm, Hsteps, Hres, kApm));
disp(["a) Cosine Bridge Rising H-Amplitudes"; ...
      "b) Cosine Bridge Falling H-Amplitudes"; ...
      "c) Sine Bridge Rising H-Amplitudes"; ...
      "d) Sine Bridge Falling H-Amplitudes"]);

colormap('jet');

% cosinus bridge recorded during rising stimulus
nexttile;
im = imagesc([Hmin Hmax], [Hmin Hmax], Vcos.Rise);
set(gca, 'YDir', 'normal');
set(im, 'AlphaData', ~isnan(Vcos.Rise));
caxis([Vmin, Vmax]);
xlim([Hmin Hmax]);
ylim([Hmin Hmax]);
axis square xy;
xlabel('$H_x$ in kA/m');
ylabel('$H_y$ in kA/m');
title('a) $V_{cos}(H_x, H_y)$');
yticks([-20 -10 0 10 20]);
xticks([-20 -10 0 10 20]);

% cosinus bridge recorded during falling stimulus
nexttile;
im = imagesc([Hmin Hmax], [Hmin Hmax], Vcos.Fall);
set(gca, 'YDir', 'normal');
set(im, 'AlphaData', ~isnan(Vcos.Fall));
caxis([Vmin, Vmax]);
xlim([Hmin Hmax]);
ylim([Hmin Hmax]);
axis square xy;
xlabel('$H_x$ in kA/m');
ylabel('$H_y$ in kA/m');
title('b) $V_{cos}(H_x, H_y)$');
yticks([-20 -10 0 10 20]);
xticks([-20 -10 0 10 20]);

% sinus bridge recorded during rising stimulus
nexttile;
im = imagesc([Hmin Hmax], [Hmin Hmax], Vsin.Rise);
set(gca, 'YDir', 'normal');
set(im, 'AlphaData', ~isnan(Vsin.Rise));
caxis([Vmin, Vmax]);
```

```
    xlim([Hmin Hmax]);
    ylim([Hmin Hmax]);
    axis square xy;
    xlabel('$H_x$ in kA/m');
    ylabel('$H_y$ in kA/m');
    title('c) $V_{sin}(H_x, H_y)$');
    yticks([-20 -10 0 10 20]);
    xticks([-20 -10 0 10 20]);

    % sinus bridge recorded during falling stimulus
    nexttile;
    im = imagesc([Hmin Hmax], [Hmin Hmax], Vsin.Fall);
    set(gca, 'YDir', 'normal');
    set(im, 'AlphaData', ~isnan(Vsin.Fall));
    caxis([Vmin, Vmax]);
    xlim([Hmin Hmax]);
    ylim([Hmin Hmax]);
    axis square xy;
    xlabel('$H_x$ in kA/m');
    ylabel('$H_y$ in kA/m');
    title('d) $V_{sin}(H_x, H_y)$');
    yticks([-20 -10 0 10 20]);
    xticks([-20 -10 0 10 20]);

    % add colorbar and place it overall plots
    cb = colorbar;
    cb.Layout.Tile = 'east';
    cb.Label.String = sprintf(...
        '$V(H_x, H_y)$ in %s, Gain $ = %.1f$', mV, gain);
    cb.Label.Interpreter = 'latex';
    cb.TickLabelInterpreter = 'latex';
    cb.Label.FontSize = 24;

%     yesno = input('Save? [y/n]: ', 's');
%     if strcmp(yesno, 'y')
%         % save results of figure 1
%         savefig(fig1, fig1Path);
%         print(fig1, fig1Path, '-dsvg');
%         print(fig1, fig1Path, '-depsc', '-tiff', '-loose');
%         print(fig1, fig1Path, '-dpdf', '-loose', '-fillpage');
%
%         % save results of figure 2
%         savefig(fig2, fig2Path);
%         print(fig2, fig2Path, '-dsvg');
%         print(fig2, fig2Path, '-depsc', '-tiff', '-loose');
%         print(fig2, fig2Path, '-dpdf', '-loose', '-fillpage');
%     end
%     close(fig1)
%     close(fig2)
end
```

Published with MATLAB® R2020b