Table of Contents

closed_loop.m

This script is for generating calibration models for the closed-loop test between the ELT S300 sensor and LICOR LI-7810 reference instrument. This script does not support multiple calibration files currently.

Lincoln Scheer 7/8/2024

This script has not been optimized and can take a while to run, sit back.

```
clc, clear, close all
```

Load Data

```
% import elt sensor dataset
daq = IMPORTDAQFILE("data/7.8.2024/daq_7_8_24.csv");
daq = [daq; IMPORTDAQFILE("data/7.8.2024/daq_7_9_24.csv")];
daq = rmmissing(daq);


% from elt sensor dataset, grab per-sensor dataset
% sensor 1 & 2 (CA & CB)
sensors = {[daq(:,[2,3,4])], [daq(:,[5,6,7])]};

% import licor reference instrument dataset
licor = IMPORTLICORFILE("data/7.8.2024/licor.txt");
licor = rmmissing(licor);
```

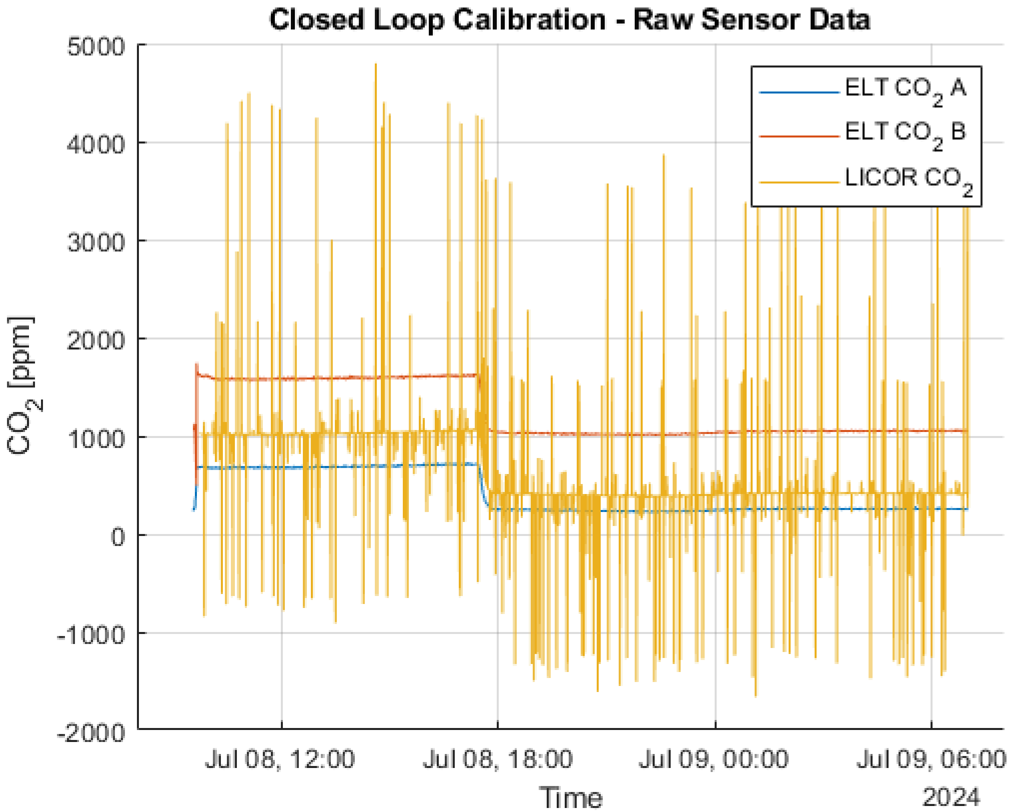
Remove Errors

```
errorVals = [-999, 500, 2815, 64537, 231753, 65535, 2500, 2559];

% remove known error values from each dataset
for index = 1:2
    sensor = sensors{1,index};
    sensor = timetable2table(sensor);
    errorMask = (ismember(table2array(sensor(:,2)), errorVals));
    sensor = sensor(~errorMask, :);
    sensor = table2timetable(sensor);
```

```
    sensors{1,index} = sensor;
end
```

Plot Raw Data

```
figure();
hold on; grid on;
plot(daq.T, daq.CA,'DisplayName', 'ELT CO_2 A');
plot(daq.T, daq.CB,'DisplayName', 'ELT CO_2 B');
plot(licor.T, licor.C,'DisplayName', 'LICOR CO_2');
xlabel("Time")
ylabel("CO_2 [ppm]")
legend();
title("Closed Loop Calibration - Raw Sensor Data");
```



Retime, Smooth, and Remove Outliers

section settings

```
smooth_dt = minutes(5);
retime_dt = seconds(5);
outlier_bounds = [10, 90];
outlier_remove = true;

% smooth and retime sensor datasets
for index = 1:2
```

```
    sensor = sensors{1,index};
    sensor = retime(sensor,"regular", 'mean', 'TimeStep', retime_dt);
    sensor = smoothdata(sensor, 'movmean', smooth_dt);
    if (outlier_remove)
        sensor = rmoutliers(sensor, 'percentile', outlier_bounds);
    end
    sensors{1,index} = sensor;
end

% smooth and retime reference dataset
licor = retime(licor,"regular", 'mean', 'TimeStep', retime_dt);
licor = smoothdata(licor, 'movmean', smooth_dt);
if (outlier_remove)
    licor = rmoutliers(licor, 'percentile', outlier_bounds);
end
```

Synchronize Datasets

```
% sync sensors with reference
for index = 1:2
    sensor = sensors{1,index};
    sensor = synchronize(sensor, licor);
    sensor = rmmissing(sensor);
    sensors{1,index} = sensor;
end
```

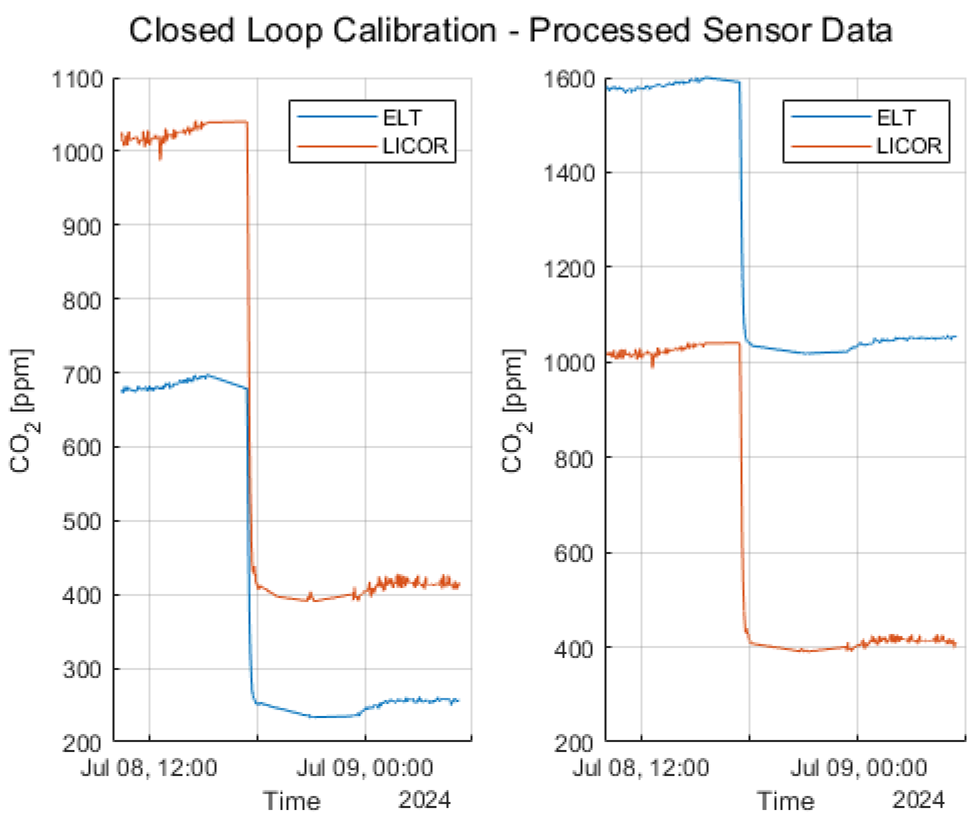
On-The-Fly Timestamp Fix

```
% look for timestamp lags, and automatically fix based on cross corelation
% of lagged datasets
for index = 1:2
    sensor = sensors{1,index};
    best_corr = 0;
    opt_lag = -inf;

    for lag = -height(sensor):height(sensor)
        if lag > 0
            shifted_C = [nan(lag, 1); sensor.(1)(1:end-lag)];
        elseif lag < 0
            shifted_C = [sensor.(1)(-lag+1:end); nan(-lag, 1)];
        else
            shifted_C = sensor.(1);
        end

        % calculate correlation, ignoring NaNs
        valid_idx = ~isnan(sensor.C) & ~isnan(shifted_C);
        if sum(valid_idx) > 100
            current_corr = corr(sensor.C(valid_idx), shifted_C(valid_idx));
            % update best correlation and lag
            if current_corr > best_corr
                best_corr = current_corr;
                opt_lag = lag;
            end
```

```
        end
    end

    % apply best lag and shift dataset
    fields = 1:3;
    for field = fields
        sensor.(field) = SHIFTDATA(sensor.(field), opt_lag);
    end
    sensor = rmmissing(sensor);
end
```

Plot Smooth Data

```
figure();
sgtitle("Closed Loop Calibration - Processed Sensor Data");
for index = 1:2
    subplot(1,2,index);
    sensor = sensors{1,index};
    hold on; grid on;
    plot(sensor, 1, 'DisplayName', 'ELT');
    plot(sensor, 4, 'DisplayName','LICOR');
    xlabel("Time");
    ylabel("CO_2 [ppm]")
    legend();
    sensors{1,index} = sensor;
end
```

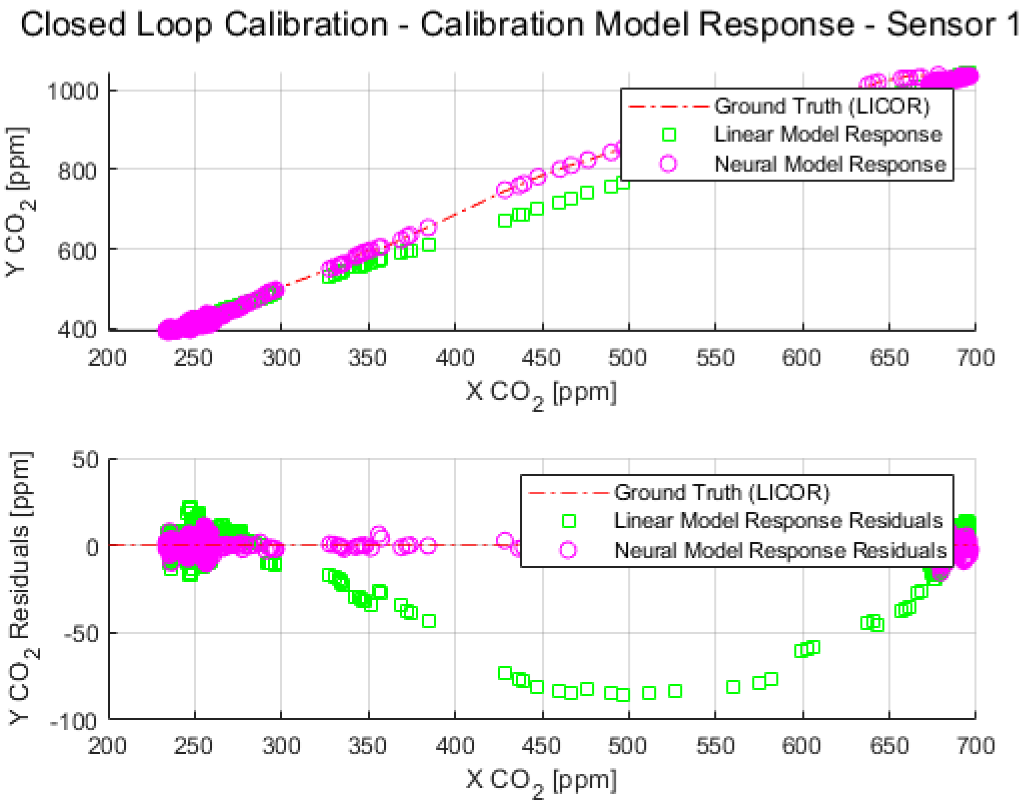


Generate Calibrations

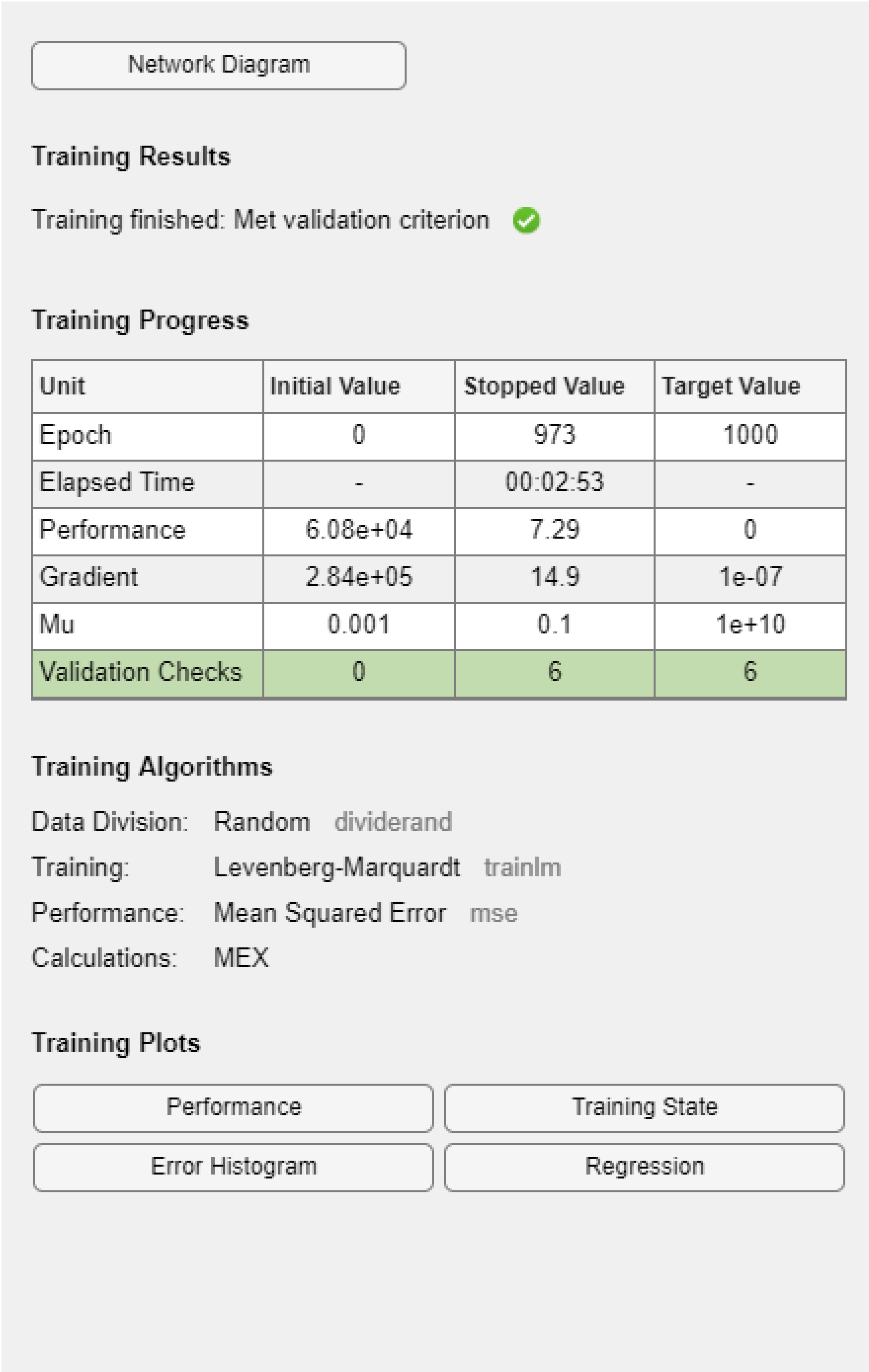
```
fprintf("Closed Loop Calibration - Calibration Results\n")
models = cell(2,2);
for index = 1:2
    sensor = sensors{1,index};
    sensor = timetable2table(sensor);

    % partition data
    cv = cvpartition(size(sensor,1), 'HoldOut', 0.3);
    trainX = table2array(sensor(~cv.test, 2:4));
    trainY = table2array(sensor(~cv.test, 5));
    testX = table2array(sensor(cv.test, 2:4));
    testY = table2array(sensor(cv.test, 5));

    % linear regression
    models{1,index} = fitlm(trainX, trainY);

    % network regression
    models{2,index} = feedforwardnet([16, 16]);
    models{2,index} = train(models{2,index}, trainX', trainY');

    % calculate metrics
    lin_pred = predict(models{1,index}, testX);
    net_pred = models{2,index}(testX')';
    lin_r2 = 1 - ((sum((lin_pred - testY).^2))/(sum(((testY -
 mean(testY)).^2))));
    net_r2 = 1 - ((sum((net_pred - testY).^2))/(sum(((testY -
 mean(testY)).^2))));
    lin_rmse = models{1,index}.RMSE;
    net_rmse = sqrt(mean((net_pred - testY).^2));

    % plot metrics
    figure()

    subplot(2,1,1)
    hold on; grid on;
    plot(testX(:,1), testY, 'r-.', 'DisplayName', "Ground Truth (LICOR)");
    plot(testX(:,1), lin_pred,'gs', 'DisplayName', "Linear Model Response");
    plot(testX(:,1), net_pred,'mo', 'DisplayName', "Neural Model Response");
    legend();
    xlabel("X CO_2 [ppm]")
    ylabel("Y CO_2 [ppm]")
    subplot(2,1,2)
    hold on; grid on;
    yline(0, 'r-.', 'DisplayName', "Ground Truth (LICOR)")
    plot(testX(:,1), lin_pred-testY,'gs', 'DisplayName', "Linear Model
 Response Residuals");
    plot(testX(:,1), net_pred-testY,'mo', 'DisplayName', "Neural Model
 Response Residuals");
    legend();
    xlabel("X CO_2 [ppm]")
    ylabel("Y CO_2 Residuals [ppm]")
```

```
    sgtitle("Closed Loop Calibration - Calibration Model Response - Sensor " +
 index)

    % print metrics
    fprintf("Sensor %d\n", index);
    fprintf("\tLinear\n")
    fprintf("\t\tRMSE:\t%0.2f\n", lin_rmse);
    fprintf("\t\tR^2:\t%0.5f\n", lin_r2);
    fprintf("\tNetwork\n")
    fprintf("\t\tRMSE:\t%0.2f\n", net_rmse);
    fprintf("\t\tR^2:\t%0.5f\n", net_r2);

    sensor = table2timetable(sensor);
    sensors{1,index} = sensor;
end
```

```
Closed Loop Calibration - Calibration Results
Sensor 1
 Linear
  RMSE: 10.24
  R^2: 0.99890
 Network
  RMSE: 3.11
  R^2: 0.99989
Sensor 2
 Linear
  RMSE: 7.27
  R^2: 0.99942
 Network
  RMSE: 2.96
  R^2: 0.99990
```



Closed Loop Calibration - Calibration Model Response - Sensor 1
Ground Truth (LICOR)
Linear Model Response
Neural Model Response
Y CO2 [ppm]
X CO2 [ppm]
Linear Model Response Residuals
Neural Model Response Residuals
Y CO2 Residuals [ppm]



Network Diagram

Training Results

Training finished: Met validation criterion

Training Progress

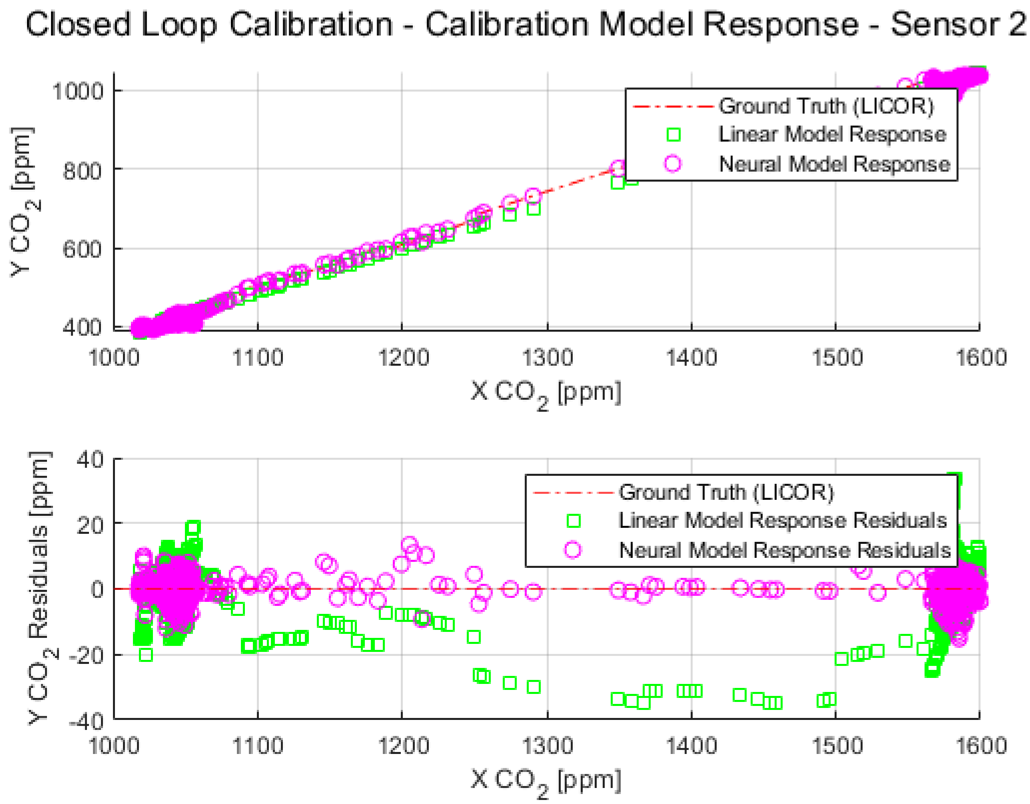
Unit	Initial Value	Stopped Value	Target Value
Epoch	0	973	1000
Elapsed Time	-	00:02:53	-
Performance	6.08e+04	7.29	0
Gradient	2.84e+05	14.9	1e-07
Mu	0.001	0.1	1e+10
Validation Checks	0	6	6

Training Algorithms

Data Division: Random dividerand
Training: Levenberg-Marquardt trainlm
Performance: Mean Squared Error mse
Calculations: MEX

Training Plots

Performance | Training State
Error Histogram | Regression



Export Models

```
sensor_id = "C";
training_id = "CL";
model_1_lin = models{1,1};
model_1_net = models{2,1};
save("model/"+sensor_id+"-"+training_id+"-
linear-"+datestr(datetime("today")),"model_1_lin");
save("model/"+sensor_id+"-"+training_id+"-
net-"+datestr(datetime("today")),"model_1_net");

sensor_id = "E";
training_id = "CL";
model_2_lin = models{1,2};
model_2_net = models{2,2};
save("model/"+sensor_id+"-"+training_id+"-
linear-"+datestr(datetime("today")),"model_2_lin");
save("model/"+sensor_id+"-"+training_id+"-
net-"+datestr(datetime("today")),"model_2_net");
```

Published with MATLAB® R2022a