```
function data = regimeNormalization(data, centers, centerstats)
% Normalizes data overall for input into FinalProjectCode
conditionIdx = 3:5;
dataIdx = 6:26;

% Evaluate rows
data{:, dataIdx} = table2array(...
    rowfun(@(row) localNormalize(row, conditionIdx, dataIdx, centers, centerstats), ...
    data, 'SeparateInputs', false));
end

function rowNormalized = localNormalize(row, conditionIdx, dataIdx, centers, centerstats)
% Normalization

% Operating Points, Sensor Measurements
ops = row(1, conditionIdx);
sensor = row(1, dataIdx);

% Cluster Centers
dist = sum((centers - ops).^2, 2);
[~, idx] = min(dist);

% Data Normalization
rowNormalized = (sensor - centerstats.Mean{idx, :}) ./ centerstats.SD{idx, :};
rowNormalized(isnan(rowNormalized) | isinf(rowNormalized)) = 0;
end

function dataFused = degradationSensorFusion(data, sensorToFuse, weights)

% Data fuse - weights
DataToFuse = data{:, cellstr(sensorToFuse)};
dataFusedRaw = dataToFuse*weights;

% Relate fused data to mean
stepBackward = 10;
stepForward = 10;
dataFused = movemean(dataFusedRaw, [stepBackward stepForward]);

% Data offset
dataFused = dataFused + 1 - dataFused(1);
end
```