Next Dimension Technologies
Sensor Data Processing

Introduction

Data processing consists of three steps:

1) Data Acquisition (low-level communications protocols)
2) Preprocessing (baseline correction, calculation of transients and $\Delta R/R$)
3) Identification (library matching)

Let's define a couple of terms that will be used through the rest of this document:

- A *sensor* is a device from NDT consisting of multiple *sensor elements*. A typical configuration might be 32 sensor elements per sensor, but more generally, this can be any N.
- A *reading* is a single resistance value from a single sensor element. It can be represented as a floating point number.
- A *scan* consists of readings from the entire sensor (i.e., all N sensor elements) at a given time point.
- A *channel* consists of all readings from a given sensor element at all time points.
- A *data file* consists of all readings from the entire sensor at all time points (you can imagine a datafile as a matrix, where rows are scans and columns are channels).

Data Acquisition

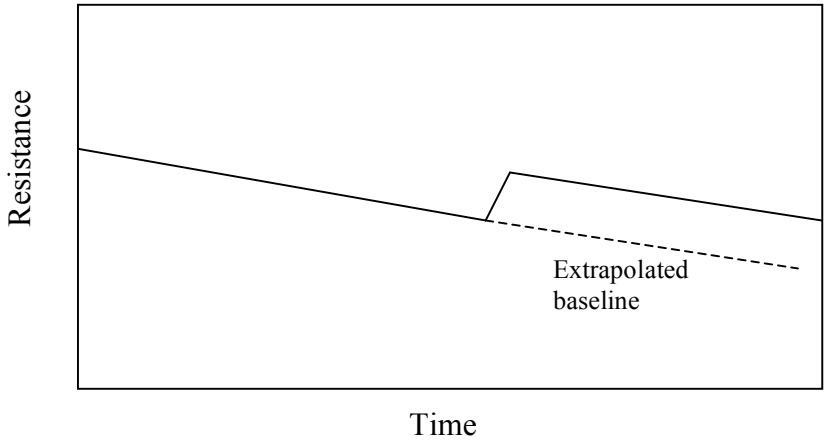
Data Acquisition is the process of acquiring *scans* from the device. The details are in the protocol document. The output is a time and a set of N resistance values.

Preprocessing

Preprocessing consists of baseline correction, event/discontinuity detection, and pattern calculation. It begins on a single channel and carries through to the entire sensor.

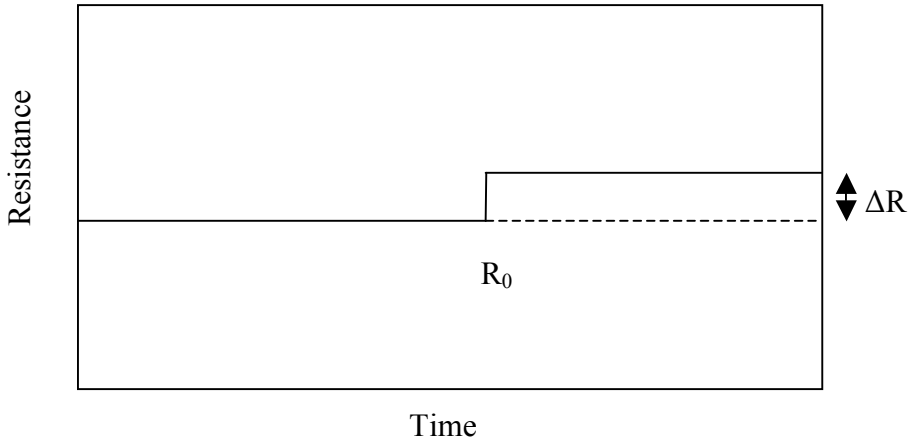
<u>Baseline correction</u>

Typical data for a single channel might look like this:



Note that the resistance drifts slowly (the slow negative slope). At some point in time, the resistance changes abruptly in response to a chemical signal.

The first step in preprocessing is *baseline correction*. A corrected version of the previous figure might look like this:



Our algorithm for baseline correction is simple:

- Keep a moving window of 10 data points per channel.
- Assuming the baseline is drifting linearly with time, find a best fit slope for the 10 data points.
- Subtract that from the actual data.
- When you hit a discontinuity (see below), use the baseline correction *prior* to the discontinuity, until you have identified the discontinuity.

<u>Detecting discontinuities</u>

Detecting discontinuities actually depends on the whole sensor, not an individual sensor element. The algorithm is:

- For each channel

    - Calculate the baseline correction from 10 points prior to the current reading
    - Calculate the standard deviation of the baseline window, post correction
    - If the current reading is more than 3 standard deviations above the baseline, mark this channel as discontinuous at this time.
- Count how many channels are marked as discontinuous.
- If the result is > ¼ N, the current time point is a discontinuity.

Delta R over R

Once you've detected a discontinuity across the sensor (not just one channel), you calculate R_0 as shown. Specifically, R_0 is the baseline corrected resistance at the point of discontinuity. (Here you probably want the time point before the discontinuity is detected, i.e., the last time point still in the baseline.)

At any subsequent time point, you can calculate ΔR. We typically use the values 1 minute after the point of discontinuity. From this value, you can calculate $\Delta R/R_0$ for each sensor element. The combination of all N of the $\Delta R/R_0$ values constitutes a pattern that is distinct for each gas.

Identification

We have a library of known gasses. For our current purposes, each gas is represented by a set of 2N numbers – N means and N standard deviations.

To match a pattern with our library, we do the following calculation:

$$P[gas] = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left[-\frac{1}{2}\left(\frac{x_i-\mu_i}{\sigma_i}\right)^2\right]$$

We then pick the gas with the highest probability match (i.e., do a forced choice). We've also tried using a threshold – either method works, but you have to be careful with the numerics, or else you just get probability = 0 always. (In other words, if you are comparing gases 1 and 2, you can just take the ratio of P(gas1)/P(gas2), or even the logarithm of that, and many of the terms cancel. See code for more details.