Sampling watt meter, power calculation algorithms

**Matlab implementations of JV sampling watt meter (a sub. set)**

Containing:

1. Simulated signal generation (Amplitude, phase, sampling rate, etc.)
2. Compensation for Frequency-dependent Gain and Phase errors.
   * Generation of a few pre-defined compensations:
     + Null-compensation
     + Cable-delay compensation
     + 3458A’s Frequency dependent response from Aperture size
   * Compensation algorithm (time-to-Frequenzy-to-time)
3. Calculation of base Power and PQ-parameters:
   * RMS-Voltage RMS-Current, AC and DC
   * Active Power
   * Reactive Power
   * Apparent Power
   * Power factor
4. Simple testing of result
   * Calculation of theoretical correct value
   * Calculate the deviation between algorithm output and theoretical values.

**1: Simulated signal generation (Amplitude, phase, sampling rate, etc.)**

|  |  |  |
| --- | --- | --- |
| **Generate Test signal (func.)** |  |  |
| testsignal.m | **Inputs:** |  |
|  | N | Numbers of samples |
|  | fs | Sampling frequency |
|  | bf | Base signal frequency (ex:50Hz) |
|  | amp | Base signal Amplitude (rms) |
|  | phi | Base signal phase in degrees |
|  | dc | Base signal DC-offset |
|  | noiseamp | random noise amplitude |
|  | **Output:** |  |
|  | s | 1-dim. Data array with the generated simulated data |

**2:** **Compensation for Frequency-dependant Gain and Phase errors.**

Functions:

|  |  |  |
| --- | --- | --- |
| **Main compensation function** | **Inputs:** |  |
| sFreqDep\_PG\_Comp.m | U1, U2 | Sampling buffer data arrays (uncompensated) |
|  | fft\_size | FFT size |
|  | CmpVector1  CmpVector2 | Complex vector holding Gain and Phase compensation |
|  | **Outputs:** |  |
|  | Uc1, Uc2 | Compensated Data arrays for Channel 1 & 2 |
|  | first, last | Index of beginning and end-part of the input buffer that the output represents (\*) |

(\* When compensating for phase, a certain numbers of samples at the ends will be unusable)

When compensating for phase, a certain numbers of samples at the ends will be unusable. The function only returns the useful data array in the middle, which is shorter than the Input array. The function returns two compensated arrays and indicate the index of ***first*** and ***last*** relative to where in the Input data array it originates.

As a thumb-of-rule, the function returns the array between index:

< fft\_size/2 : SIZE - ~fft\_size/2 > ,

and the cut-off at the end depends on the matches between the Input-length and the FFT-length-multiple. If an fft-buffer is incomplete, the surplus data will be discarded. The length of the output is (***last*** - ***first***).

Functions that generate a Compensation vector.

|  |  |  |
| --- | --- | --- |
| **Pre.made Compensation func.** | **Inputs:** | **Func. Description:** |
| NullCompVector.m | FFT\_size | Default Null compensation |
| ConstantDelayCompVector.m | fs, FFT\_size, delay\_s | Generate compensation for cable-delay (constant delay) |
| H3458ACompVector.m | fs, FFT\_size, intgration\_time | Generate compensation for the HP3458A frequency-dependant gain, as a function of the Aperture time. |
|  | **Outputs:** |  |
|  | CompVector | Complex array used as argument for sFreqDep\_PG\_Comp. |

I addition to these examples, the user should characterize their own setup, to identify the phase and gain corrections needed in the spectra.

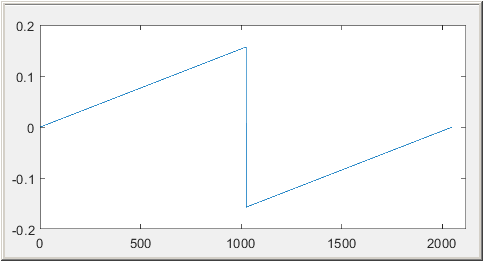
The CompVector has this form:

Complex array of length FFT\_size, the absolute value of the complex value is the gain, and the angel[rad] give the phase correction.

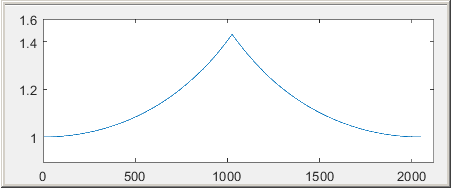
For element ***i***,

freq[hz]=***fs/FFT\_size\*i;*** for ***i*** <0,***FFT\_size***/2-1> , and

freq[hz]=***fs/FFT\_size\*(FFT\_size-i)***; for ***i*** <***FFT\_size/2, FFT\_size***-1 >

The ***Phi*** is complex-conjugated, and thus the phi-values are symmetric for the first and second half of the array, so the sign of the values and the order are flipped round ***FFT\_size/2***.

Gain: Gain values are mirrored round the center of the array.



Other functions:

|  |  |
| --- | --- |
| **Help-functions:** | Description: |
| HannFcompMask.m | Generate window masking edge-effects of the inv. FFT |
| PackMan.m | Generate indexing for position of FFT-window |
| sincm.m | The sinc-function: [sin(x)/x] |
| hanningw.m | Even-number Hanning-window function, with asymmetric peek-point |
| FDcomp.m | The core “*time-frequency-time*”-domain comp.function (FFT) |

**4: Calculation of base Power and PQ-parameters:**

Functions:

|  |  |  |
| --- | --- | --- |
| **Main Calculation function** | **Inputs:** |  |
| swm1.m | U1 | Compensated sampled data array for U1 (Voltage-channel) |
|  | U2 | Compensated sampled data array for U2 (Current-channel) |
|  | rootedWW | Windowing-function1(\*) |
|  | WW | Windowing-function2(\*) |
| Main Parameters calculated are: | **Outputs:** |  |
|  | U1rmws | RMS-value of U1(Voltage-channel) |
|  | U2rmws | RMS-value of U1(Voltage-channel) |
|  | U1dc | DC-offset of U1 (Voltage-channel) |
|  | U2dc | DC-offset of U2 (Current-channel) |
|  | Pact | RMS-value of Active Power |
|  | Prea | RMS-value of Reactive Power |
|  | Papp | RMS-value of Apparent Power |
|  | PF | Power factor |

(\* For efficiency, calculating the window once, and give the arrays as argument for repeated calls)

**4: Simple testing of result**

|  |  |  |  |
| --- | --- | --- | --- |
| **Example-implementation** | | **Input/output** | Example code, demonstrating the use of the algorithms for calculating Power and PQ-parameters on simulated input data |
| SWM\_PUI\_v1.m | | None |
|  |  | | |
| 1 | Data\_in: Two arrays of sampled or simulated 3458A voltage-data.  Simulated data used in this example: testsignal(N,fs,Hz,U,Udc,dPhi,noise);  Array-length: To Archive better then 1ppm max. error contribution from the calculation algorithm, The length of the data arrays should contain more data than 32(\*) periods of the base signal. In addition, since the compensation algorithm will throw away some data points at the start and end of the buffer, the size of the buffer should be at least be:  N= ***fs/BaseFreq\*32.0 + 2\*FFT\_size***;    (\* for Hanning-window, more the 32 periodes is needed for better then 1ppm) | | |
| 2 | Compensation of frequency-dependent errors (phase-Gain),  **sFreqDep\_PG\_Comp(U1,U2,fft\_size,CmpVector1,CmpVector2);**  Tree examples of simulated compensations is provided.  For example: compensation for cabel delay, or the Frequency-dependant gain of the HP3458A. | | |
| 3 | The main algorithm, The Time-based calculation of Base Power and PQ-Properties:    **[U1rmws, U2rmws, Pact, Papp, Prea, PF] = swm1(Uc1, Uc2, rootedWW);**  The two arrays from the Compensation algorithm is input here.  Main Paraneters calculated are:  RMS-value(U1,U2),DC(U1,U2), Active Power,Reactive Power,Apparent Power,Power factor  **ACCURRACY:**  Contribution of uncertainty depends on the Input data length relative to the Base signal period length. For better then max. 1ppm error contribution, the input data length must be longer than 32 periods of the Base signal. | | |
| 4 | Testing of the calculated values against the theoretical (ref) values. | | |