Implementation and Testing of a Real-Time Mode Estimation Algorithm using Ambient PMU Data

Vedran S. Perić[1], Maxime Baudette[1], Luigi Vanfretti[1,2], Jan O. Gjerde[2], Stig Løvlund[2]
[1]KTH Royal Institute of Technology, Stockholm, Sweden
[2]Statnett SF, Oslo, Norway

***Abstract* —This paper presents a software implementation of a real-time power system mode estimator which uses ambient synchrophasor data. The software is built using Statnett's Software Development Kit (SDK) as a platform for fast prototyping of real-time synchrophasor applications. The SDK extracts synchrophasor data received in the IEEE C.37.118 protocol and provides them as LabVIEW signals. These signals are preprocessed and mode frequencies and damping ratios are calculated by Yule-Walker's method. The implemented LabVIEW software employs state machine logics which enables modifications and upgrades to the algorithm.**

***Index Terms*—Synchrophasors, mode estimation, mode meters, state machine.**

I. Introduction

SYNCHROPHASOR technology is widely recognized as an important element that can aid in the enhancement of modern power systems monitoring and control [1]. One important synchrophasor application is the continuous real-time monitoring of electromechanical oscillations, which is usually referred to as mode estimator or mode meter. Mode meters apply signal processing algorithms to ambient data (process noise) which is present in all synchrophasor measurements [2].

Measurement based estimation of electromechanical modes is a complex problem due to the high dimensionality, nonlinearity, and time variant structure of the power system. This problem is a topic of ongoing interdisciplinary research between the power systems and the system identification communities.

In addition to the theoretical development of methods for mode estimation [3], it is important to test an integrated solution for mode estimation including measurement acquisition. This includes physical Phasor Measurement Units (PMUs) and the Information & Communication Technologies (ICT) systems that support the mode meter application.

This paper presents a mode meter application which is integrated with the entire Wide Area Measurements Systems (WAMS) ICT infrastructure. Statnett's Software Development Kit (SDK) allows fast prototyping and testing of the integrated WAMS solution [4]. It is used here as a platform for developing the mode estimator. The SDK extracts synchrophasor data received in the IEEE C.37.118 protocol and converts them to a more convenient form (LabVIEW signals). The mode estimator is implemented in the LabVIEW environment using state machine logic. This architecture enables easy modifications and further developments to the estimator.

The paper is organized as follows: Section II outlines the algorithm used in the mode meter application. Section III describes Statnett's SDK platform. The architecture of the developed mode meter software is given in Section IV, whereas the user interface and experimental testing results are given in Section V and Section VI, respectively. Conclusions are drawn in Section VII.

II. Mode Estimation Algorithm

During (quasi) steady state operation, it can be assumed that the power system is mainly disturbed by small random load variations. These load variations are caused by the random behavior of individual consumers at different voltage levels. Since these random load changes are small in magnitude, the power system behavior can be described by a linear model. Therefore, the power system model can be written as follows:

$$\mathbf{Y}(j\omega) = \mathbf{H}(j\omega) \cdot \mathbf{U}(j\omega)\,, \tag{1}$$

where:

$\mathbf{Y}(j\omega)$ – vector of measured electric variables;

$\mathbf{H}(j\omega)$ – transfer function matrix of the power system;

$\mathbf{U}(j\omega)$ – vector of inputs (load variations).

In order to demonstrate the mode estimation algorithm principle, we will consider only a single input-single output system [5]. Assuming that the individual load behavior is random and independent from other loads, the (input) load variations can be represented by white noise. Keeping in mind that the frequency spectrum of white noise is a constant function, it is follows that the spectrum of the measured signal is proportional to the amplitude response of the system ($|H(j\omega)|$). Therefore, the mode estimation algorithm determines the coefficients of a rational transfer function whose amplitude response is proportional to the spectrum of measured signal. This can be written in the discrete domain time domain as:

$$y(k) = -\sum_{i=1}^{p} a_i y(k-i) + \sum_{j=0}^{q} b_j u(k-j)\,, \tag{2}$$

where:

$y(k)$ – measured output signal at time point k;

$u(k)$ – random load input at time point k (assumed to be white noise);

$a_i\,, b_j$ (i=1,…, p, and j=0,…, q) – unknown coefficients of the rational transfer function;

p, q – orders of the numerator and denumerator of the estimated rational transfer function, respectively.

Multiplying both sides of (2) by $y(k-n-l)$, taking the expected value and using the definition of autocorrelation (r) [6], the following matrix equation can be written [7]:

$$\begin{bmatrix} r(q) & \cdots & r(q-p+1) \\ \vdots & & \vdots \\ r(q+p-1) & \cdots & r(q) \end{bmatrix}\begin{bmatrix} a_1 \\ \vdots \\ a_p \end{bmatrix} = -\begin{bmatrix} r(q+1) \\ \vdots \\ r(q+p) \end{bmatrix} \tag{3}$$

Note that the following property is used in the derivation: $E\{u(k-j)\cdot y(k-q-l)\}=0$, for l=1,..,p. (4)

Autocorrelations in (3) are estimated using (5):

$$r(n)=\frac{1}{N}\sum_{k=0}^{N-n-1} y(k+n)y(k) \tag{5}$$

The solutions of the system (3) are the autoregressive (AR) coefficients of the model, which are sufficient to compute the modes of the system. In case that the moving average (MA) part is also of the interest, it can be computed using Durbin's method [8].

The characteristic equation of the system is defined by the computed AR coefficients as follows:

$$1+a_1+\ldots+a_p=0\,. \tag{6}$$

The roots of the characteristic equation represent the modes of the system in the z-domain. These modes (denoted by **z**) can be transformed easily to the s-domain using:

$$\mathbf{s}=\boldsymbol{\sigma}+j\boldsymbol{\omega}=\frac{1}{T_s}\ln(\mathbf{z})\,, \tag{7}$$

where T_s is the signal's sampling period. $\boldsymbol{\sigma}$ and $\boldsymbol{\omega}$ are real and imaginary components of the modes in the s-domain (**s**), respectively. Once the s-domain modes of the system are calculated, the damping ratio of the i-th pole (ξ_i) is computed using the following formula:

$$\xi_i=\frac{-\sigma_i}{\sqrt{\sigma_i^2+\omega_i^2}}\,. \tag{8}$$

where σ_i and ω_i are i-th elements in $\boldsymbol{\sigma}$ and $\boldsymbol{\omega}$, respectively.

III. Statnett's Software Development Kit (SDK)

Statnett's Software Development Kit (SDK) enables easy real-time access to PMU and PDC streams [4]. The LabVIEW platform provides easy integration with different hardware equipment as well as intuitive graphical programing language (G language). The main benefit of the software development toolkit is that it exempts a developer of complicated synchrophasor data handling. Instead, the developer is required only to set the appropriate PDC connection parameters such as PDC ID, PDC host address and the port number to connect to the PDC stream.

The general architecture of the SDK is shown in Fig. 1.

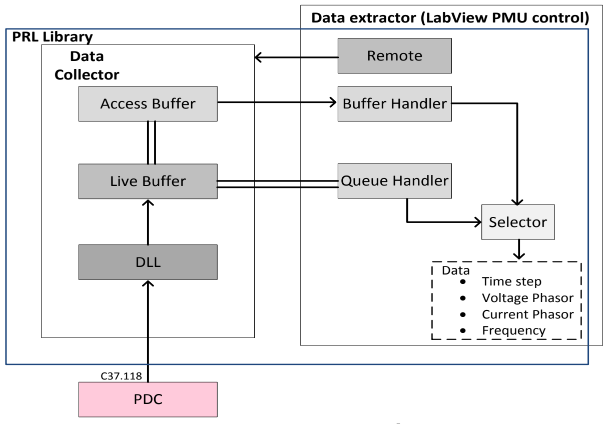


Fig. 1. Statnett's SDK architecture.

The SDK has two major components, *Data Collector* and *Data Extractor*.

A. *Data Collector*

The Data Collector reads the data from the PDC/PMU and stores them in configurable buffers. This component uses a Dynamic-Dink Library (DLL) developed in the C programming language to connect to the PDC stream via the IEEE C37.118.2 protocol. In addition, the DLL reads the configuration data of the PDC stream, such as channel names, scaling and number of the measured signals by type (analog, phasors or digital signals).

The incoming data from the DLL are stored in a Live Buffer. When the data buffer is full, the data is put in a queue that sends data to the Access Buffer, the data then can be received by a user application using the Queue Handler. The Access Buffer size decides on the amount of data history to be kept in memory, and the data that can be read from the buffer in the Custom Application via the Buffer Handler.

B. *Data extractor – LabVIEW PMU control*

The data extractor is a collection of functions (VIs) that allows the user to access the buffers and queues in the Data Collector. It reads the data from the buffers and provides the user with control over the data streams in a form suitable for further processing in the main application (as a signal data type in LabVIEW). The interface of this VI is shown in Fig. 2.

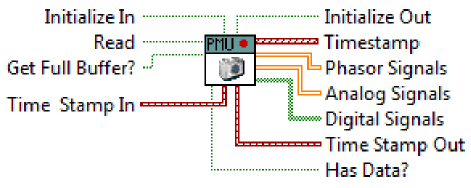


Fig. 2. LabVIEW PMU Control.

Other parameters such as PMU selection, data length, and others, are set in an auxillary user interface. Additional LabVIEW VIs are provided in the SDK for data handling and processing [4].

IV. Mode Meter Software Architecture

The state machine architecture is chosen for the mode meter application development. This architecture allows to decouple different tasks and to develop them independently. The block diagram of the state machine is given in Fig. 3:

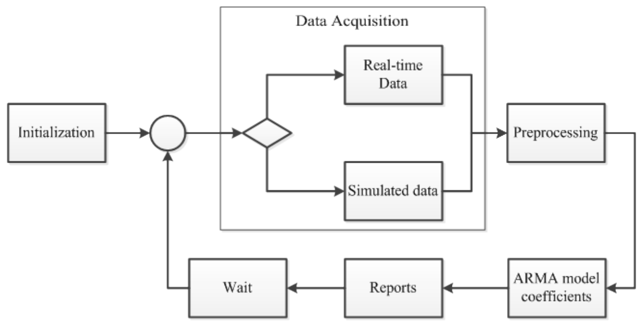


Fig. 3. Global block diagram of the implemented mode meter.

The software's structure, which is typical for real-time applications, consists of an initialization step and a main loop.

The initialization step is performed only once and it sets all variables to their initial values, as well as populating initial settings such as data source, refresh rate, etc. After the initialization step the program enters the main loop which consists of four blocks, namely: data acquisition, preprocessing, estimation of ARMA coefficients and reports.

A. *Data acquisition*

Data acquisition imports a parcel of data which is used for mode estimation. The length of data is defined in the software's options and usually is in the range of 10-15 min. There are two operating modes of data acquisition:

- Acquisition through a PDC and Statnett's SDK. The signal used for mode estimation is selected among available real-time measurements in Statnett's SDK interface [4].
- Acquisition from an internal data generator. The internal data generator provides Gaussian white noise filtered by linear an IIR filter. The coefficients of the IIR filter are set manually by the user in the *Testing* tab (this tab is described in Section V).

Software modularity is achieved through a standardized interface among the blocks. The following interface is adopted and implemented using a "type definition" LabVIEW structure (Fig. 4.):

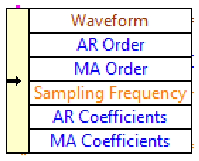


Fig. 4 Common data structure

This user defined type consists of 5 elements, namely:

- *Waveform* – Data parcel with corresponding time stamps;
- *AR Order* – Order of the estimated autoregressive part of the model;
- *MA Order* – Order of the estimated moving averge part of the model;
- *Sampling frequency* - Sampling frequency of the signal;
- *AR Coefficients* – Calculated autoregressive coefficients;
- *MA Coefficients* – Calculated moving average coefficients.

This user defined type is sufficient for communication between different blocks. This means that the block's functionality is well defined and it is easy to maintain the interfaces between blocks. This also means that each individual block can be developed independently, what makes the software maintenance easier.

B. *Preprocessing*

After the data parcel is imported, the parcel is preprocessed in order to make it more adequate to apply the system identification algorithm. This includes signal downsampling, mean and outlier removal. Downsampling greatly improves the accuracy of the estimation because only a frequency range of the interest is considered, i.e. the estimator is not constrained by fitting dynamics at higher frequencies. Also, in the case of high sampling frequency data, the modes in z-domain tend to be grouped around point (1,0) in the complex plane what creates numerical difficulties. More details can be found in [9].

C. *Estimation of ARMA coefficients*

A preprocessed signal is fed into the main computation block which calculates the ARMA coefficients of the stochastic process. The algorithm used in this block is given in Section II. This block does not perform any further computation because it is important to keep it independent from other blocks in order to make the core algorithm easy to update or replace. The computed ARMA coefficients are stored in the interface structure (Fig. 4.). The coefficients are later used for calculation of the estimation results presented to the user.

D. *Reports*

The computed ARMA coefficients are used in the Report block to present results to the user. This block computes the spectrum, and poles in the discrete and continuous domain (described by (7) and (8)). Furthermore, this block buffers results from previous iterations. Each of these calculations are performed in sub-routines (sub VIs).

The algorithm described is executed in a timed loop, and paused after the Report block has been executed. The next iteration starts at the time specified in the settings of the timed loop.

Note that overlapping in the algorithm is defined by the loop period and the parcel data length.

V. User Interface

The user interface is designed to provide relevant information about the estimated modes of the system. There are two main parts of the interface: 1) Time domain signal plots (upper part) and 2) Computed results and estimator's settings (lower part).

In the upper part, the measured signal is shown in the time domain, as well as the signal which is obtained after preprocessing (Fig. 5).

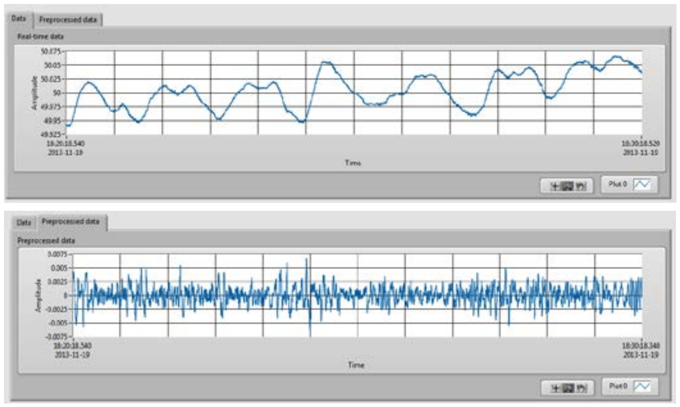
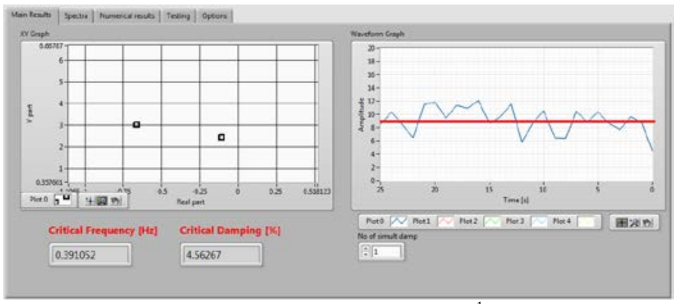


Fig. 5. Time domain signal representation.

The lower part is composed of five parts (tabs), namely:

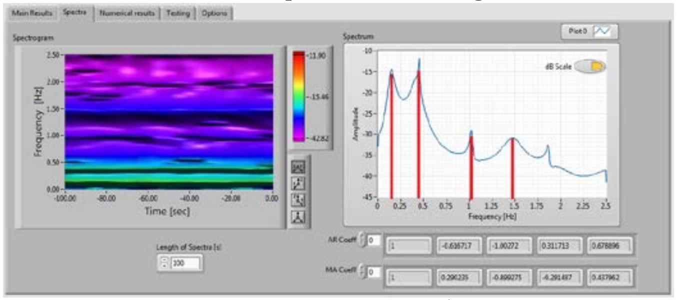
- *Main Results* tab (Fig. 6). This tab shows the estimated modes location in the complex plane as well as the history of estimated damping ratios. In addition, the frequency and damping ratio of the most critical mode are shown.



Critical Frequency [Hz] 0.391052
Critical Damping [%] 4.56267
No of simult damp 1

Fig. 6. Main results interface[1].

- *Spectra* tab (Fig. 7.). Shows the measured signal in the frequency domain. The spectrogram also gives information how the spectrum has changed over time.



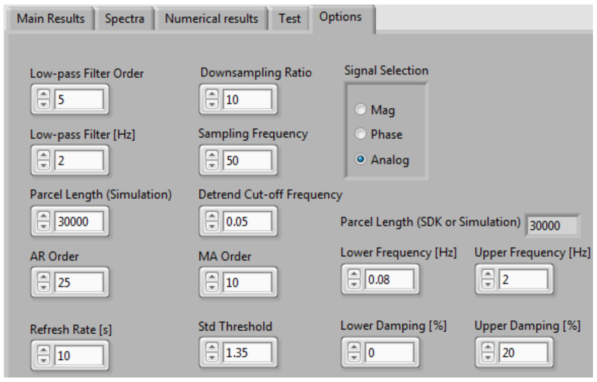
Length of Spectra [s] 100
AR Coeff 0: 1, -0.616717, -1.00272, 0.311713, 0.678896
MA Coeff 0: 1, 0.290235, -0.899275, -0.291487, 0.437962

Fig. 7 Spectral results[1].

- *Numerical results* tab. This tab presents the estimation results in form of numbers which are required for detailed analysis.
- *Test* tab. Defines the parameters of the IIR filter used in the simulation operating mode.
- *Options* tab (Fig. 8). Defines the parameters of the estimator such as data length, etc.

VI. Experimental Results

The developed mode meter is first tested in KTH SmartTS Lab using real-time hardware in-the-loop simulation with Opal-RT simulator and physical PMUs connected to it [10]. In the second test, a PMU connected to low voltage grid is used to for mode estimation in the Nordic grid.

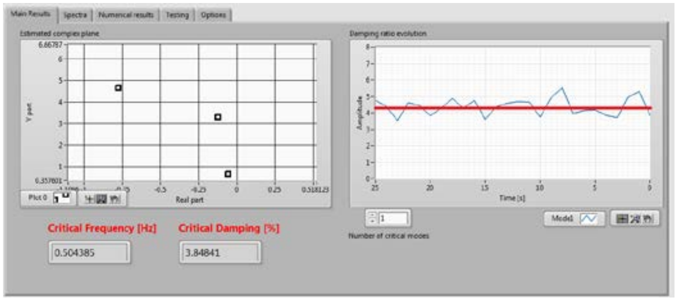


Low-pass Filter Order 5
Downsampling Ratio 10
Signal Selection: Mag [ ] Phase [ ] [✓ Analog]
Low-pass Filter [Hz] 2
Sampling Frequency 50
Parcel Length (Simulation) 30000
Detrend Cut-off Frequency 0.05
Parcel Length (SDK or Simulation) 30000
AR Order 25
MA Order 10
Lower Frequency [Hz] 0.08
Upper Frequency [Hz] 2
Refresh Rate [s] 10
Std Threshold 1.35
Lower Damping [%] 0
Upper Damping [%] 20

Fig. 8 Options tab of the mode estimator.

A. *Real-time hardware-in-the-loop test*

Opal-RT simulator is used for the hardware-in-the-loop (HIL) simulation of the KTH Nordic 32 test system [11]. This system has a dominant mode at 0.4987 Hz with 3.5223 % damping ratio (computed using classical small signal stability analysis). An ambient response of the system is simulated by imposing random load variations in all load buses. It is assumed that these variations are described by Gaussian white noise. Voltage waveforms at bus No. 49 are chosen for mode estimation because that bus has the highest observability of the dominant mode [12]. The simulated signals are amplified by Megger Smrt-1 Amplifiers and fed into a National Instruments PMU implemented in the CompactRIO platform [13]. The PMU computes the frequency using the three phase signals and sends it the mode meter application. The data parcel length used for estimation is chosen to be 10 minutes.



Critical Frequency [Hz] 0.504385
Critical Damping [%] 3.84841
Number of critical modes 1

Fig. 9 Mode estimation results using signals generated by the simulated power system

The results of this test are depicted in Figs. 8-9. It can be seen that the estimator finds the 0.5 Hz mode to be the most

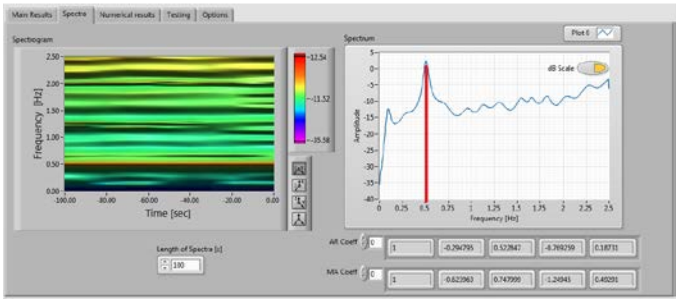
[1] The red lines in Figs. 6-7 are added to show important values (results) obtained from the tests and they are not part of the application interface.

critical with damping ratio oscillating around the value of 4 %. Since this is a stochastic process (estimation), these results are not sufficient for a full assessment of the estimator's performances. To get better insight into the estimator's performances, a large number of estimates (120 estimates based on 120 different 10-minutes data parcels) is recorded and the mean value and variance of the estimates are computed using:

$$\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{9}$$

$$\mathrm{var}(x) = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^2 \tag{10}$$

where $\overline{x}$ is the mean value of the all estimates and N is the number of estimates (N=120).



Length of Spectra [s]: 100

AR Coeff 0: 1, -0.294795, 0.522847, -0.769259, 0.18731

MA Coeff 0: 1, -0.622963, 0.747999, -1.24941, 0.49291

Fig. 10 Spectrum of the signal generated by the real-time hardware-in-the-loop simulation.

Further, these results are compared against an off-line mode estimator implementation in Matlab, without physical PMUs involved (no HIL). This means that it is possible to distinct the error caused by the estimator itself from measurement errors (there are no measurement errors within the Matlab simulation environment).

TABLE I STOCHASTIC PROPERTIES OF THE MODE ESTIMATOR

	Hardware in the loop	Software simulation
Mean {f} [Hz]	0.5073	0.4985
Mean {ξ} [%]	4.0910	3.6905
Var {f}	1.1037e-04	7.5797e-05
Var {ξ}	1.9088	1.7184

It can be seen that HIL test introduces additional error which can be explained by the imprecision of the HIL chain, including limits of the PMU's accuracy.

B. *Test using measurements from the Nordic grid*

After having validated the software in the laboratory, the software was tested using real-time measurements from a PMU in the Nordic grid. The results from this test are shown in Figs. 5-7, whereas the options used are shown in Fig. 8. Several oscillatory modes can be observed in Figs.6-7. The most critical mode has a frequency of around 0.39 Hz and damping around 9 % (in average). Other modes that are observable appear at 0.2 Hz, 1 Hz and 1.4 Hz, while a 0.5 Hz mode sporadically appears as a poorly damped mode. These estimates are quite in accordance with the results reported in [14]. Also, the mode with frequency of 1 Hz has been reported in [15] and (supposedly) it can be classified as a forced oscillation.

CONCLUSION

The paper presents a real-time mode meter software implementation and testing in the real-life conditions. The mode meter uses the well-known methodology based on Yule-Walker equations. The performed tests show that the tool provides results comparable to those obtained in the software simulation mode. In addition, the paper demonstrates that with the tools used, a powerful real-time WAMS application can be developed in relatively short time.

In order to make a full use of the ambient data-based mode estimation approach, further improvements of the tool are necessary. In addition to the implementation of more sophisticated and precise mode estimation algorithms, one of the main requirements is that the algorithm provides confidence intervals for the estimates. Confidence intervals are necessary for operators to take corrective actions initiated by the mode estimator results. Furthermore, the tool should be able to use all available synchrophasor signals.

REFERENCES

[1] S. Horowitz, A.G. Phadke and B. Renz, "The Future of Power Transmission", *IEEE Power Energy Mag.*, vol.8, no.2, pp.34-40, Mar.-Apr. 2010.

[2] J.J. Sanchez-Gasca (Ed.), "Identification of electromechanical modes in power systems", IEEE Task Force Report, Special Publication TP462, 2012.

[3] V.S. Perić and L. Vanfretti, "Mode Estimation using Load Spectral characteristics", *IEEE Trans. Power Syst.*, accepted for publication, 2013.

[4] L. Vanfretti, et al., "A Software Development Toolkit for Real-Time Synchrophasor Applications", *in Proc. PowerTech 2013*, 16-20 June 2013.

[5] B. Friedlander and B. Porat, "The Modified Yule-Walker Method of ARMA Spectral Estimation", *IEEE Trans. Aerosp. Electron. Syst.*, vol.AES-20, no.2, pp.158-173, Mar. 1984.

[6] M. H. Hayes, Statistical Digital Signal Processing and Modeling, John Wiley & Sons Inc., 1996.

[7] V.K. Madisetti (Ed.), "The Digital Signal Processing Handbook", CRC Press, 2009.

[8] N. Sandgren, P. Stoica and P. Babu, "On moving average parameter estimation", *in Proc. Eur. Signal Process. Conf.* (*EUSIPCO*), 27-31 Aug. 2012.

[9] L. Vanfretti, S. Bengtsson and J.O. Gjerde, "Preprocessing synchronized phasor measurement data for spectral analysis of electromechanical oscillations in the Nordic Grid", in press, *Eur. Trans. Electr. Power*, 2013.

[10] M.S. Almas et al., "Synchrophasor Network, Laboratory and Software Applications developed in the STRONg2rid project", submitted to *IEEE PES General Meeting*, 2014.

[11] Y. Chompoobutrgool, W. Li and L. Vanfretti, "Development and implementation of a Nordic grid model for power system small-signal and transient stability studies in a free and open source software", *in Proc. IEEE Power Eng. Soc. General Meeting 2012*, 22-26 July 2012.

[12] A.M. Almutairi and J.V. Milanović, "Comparison of different methods for optimal placement of PMUs", *in Proc. IEEE PowerTech 2009*, 28 June - 2 July 2009.

[13] NI Advanced PMU Development System, www.ni.com

[14] K. Uhlen, et al., "Wide-Area Power Oscillation Damper Implementation and Testing in the Norwegian Transmission Network", in Proc. *IEEE PES General Meeting 2012*, 22-26 July 2012.

[15] L. Vanfretti. et al., "Spectral estimation of low-frequency oscillations in the Nordic grid using ambient synchrophasor data under the presence of forced oscillations", *in Proc. IEEE PowerTech 2013*, 16-20 June 2013.