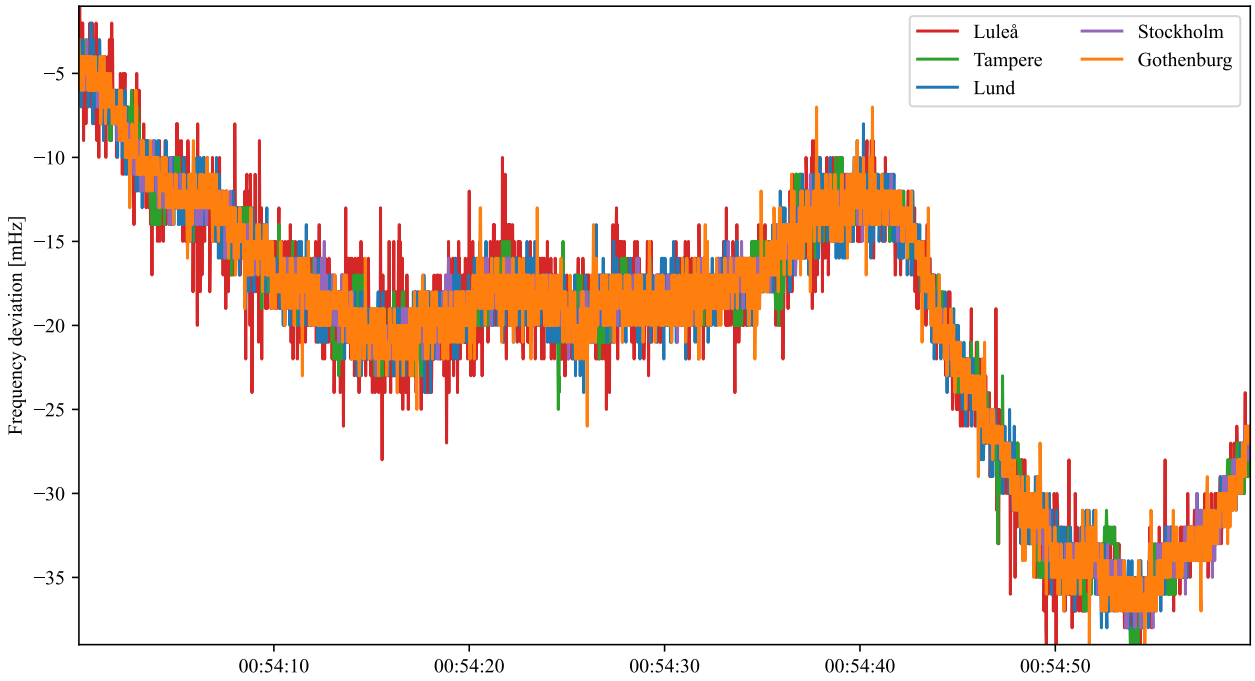
CHALMERS



Estimation of inter-area oscillations in the Nordic power system using dynamic mode decomposition

Master of Science Thesis in Electric Power Engineering

Børge Lund

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2023

Master Thesis EENX30

Estimation of inter-area oscillations in the Nordic power system using dynamic mode decomposition

CHALMERS

Department of Energy and Environment
Division of Electric Power Engineering
Chalmers University of Technology
Gothenburg, Sweden 2023

Estimation of inter-area oscillations in the Nordic power system using dynamic mode decomposition
Børge Lund

Department of Electrical Engineering
Division of Electric Power Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone +46 31 772 1000

Thesis supervisor
Mattias Persson
RISE

Thesis examiner
Peiyuan Chen
Department of Electrical Engineering
Division of Electric Power Engineering
Chalmers University of Technology, Gothenburg, Sweden

Cover image: One minute of frequency deviation data from 31 January 2016 from 5 university PMUs in Sweden and Finland.

Typeset in LaTeX
Gothenburg, Sweden 2023

Abstract

Electrification plays an important part in reducing greenhouse gas emissions to halt climate change, but creates growing demand for electric power. At the same time as electric power generation must be increased to meet this demand, the electric power industry must also transition to more climate-friendly energy sources. Therefore a large share of the power will have to come from converter-based generation such as wind and solar in the foreseeable future. Stability issues in the electric power system are likely to arise because converter-based generators do not contribute to the inertia of the power system. One stability issue which may become bigger in a low-inertia power system is inter-area power oscillations, which happen when synchronous generators in different geographical areas start to swing against each other. Inter-area oscillations already limit the transmission capacity of certain transmission corridors in the Nordic power system.

Dynamic mode decomposition (DMD) is an algorithm for oscillation detection which has been noted as promising for power-system monitoring in several studies. However, a lot of research remains on the parameterisation of this algorithm and analysis of its estimation capabilities. This project provides validation of the algorithm's ability to recognise known structures in data. Statistical tests were performed to chart the noise-sensitivity of DMD with respect to changes in sampling rate, data shift-stacking, the number of data channels, and the length of the analysis window.

DMD was implemented with a randomised sampling technique and a rank selection method which had not been combined earlier, and which had not been tested for mode-tracking in PMU data over time. The algorithm's performance was first tested on dynamic simulation data from the power-systems simulation software PSS/E. A user-defined load model was written to introduce stochastic consumption variations as a random walk in order to simulate ambient power-systems behaviour. DMD mode estimates were compared with results from stochastic subspace identification (SSI) and the multivariate autoregressive method (MAR), two algorithms which have already been in use by Nordic TSOs. DMD was found to give comparable results to SSI and MAR both under very noisy ambient conditions and on undamped oscillations after a disturbance, while having significantly lower computational cost.

DMD, SSI and MAR were applied to tracking a mode in real-life PMU data from the Nordic power system under ambient conditions. Again DMD was found to give comparable results to SSI and MAR. A possible limitation in DMD's ability to estimate the damping when using long analysis windows was identified.

Keywords: *Power-system stability, inter-area oscillations, modal analysis, dynamic mode decomposition*

Acknowledgements

I would like to thank my examiner Peiyuan Chen for introducing me to the subject of this thesis, for his enthusiastic teaching of power-system stability, and for all comments and support. I would like to thank my supervisor Mattias Person for all the input and support and for giving me great opportunities at RISE during this master thesis project and on previous occasions. To everybody in the power system unit at RISE, thank you for welcoming me and taking interest in my work. Especially thanks to Emil Hillberg for feedback and Erik Weihs for help with code. A big thank you to Morten Hemmingson at the University of Lund for supplying the immensely interesting PMU data on short notice. To my opponent and master thesis worker colleague at RISE, Mahantha Ampavatina Kambagiri, thanks for invaluable input and for all the great discussions and the bike rides. Finally, and most of all, I would like to express my deep gratitude toward my life partner Tone Fredrikke Strøm for her unfaltering patience and support during six years of university studies for a mid-life career change.

Table of Contents

1 Introduction

Electrification is a key ingredient in the energy transition needed to combat climate change. The global surface temperature has risen 1.1 degrees Celsius since pre-industrial times due to greenhouse gas emissions from human activities, and we are not on track to limit warming to 2 degrees. Among the consequences are extreme weather, irreversible damage to ecosystems, negative effects on human health as well as economic losses, all of which will continue to intensify in the years to come [1].

Electrification is happening much faster in the Nordic power system than previously anticipated: The Swedish transmission system operator Svenska Kraftnät recently updated its prognosis for the Swedish electric energy demand in 2040 from 180 to 290 GWh – twice the Swedish electric energy consumption today. Much of the increase in supply to cover the growing demand will come from intermittent sources such as wind and solar generation, leading to a less predictable power system. Substantial challenges to the stability of the power system will arise because wind and solar generators are connected to the grid via power-electronic converters and therefore do not contribute to the inertia traditionally provided by the rotating mass of synchronous generators. Among these challenges are inter-area power oscillations, which occur when synchronous generators in different areas of the grid start to swing against each other. This phenomenon, which falls under the scope of rotor-angle stability, can lead to a collapse of the power system if oscillations become undamped and are allowed to grow unchecked. The problem of power oscillations is likely to grow as larger amounts of power are transferred over longer distances, increasing transmission angle differences and therefore decreasing the margins of rotor angle stability [2].

This master thesis project concerns the implementation, validating and testing of a data-driven method for detection of inter-area power oscillations using PMU (phasor measurement unit) data. The thesis work was conducted in collaboration with RISE (Research Institutes of Sweden). RISE is currently participating in Nordic Early Warning Early Prevention System (NEWEPS) together with Svenska Kraftnät, Statnett, SINTEF, NTNU and KTH. NEWEPS is a research project with the aim of developing and demonstrating methods for power-system surveillance and control in the Nordic power system. Hopefully, the results of this project can be of value in that context.

1.1 Background

Inter-area power oscillations are electromechanical phenomena, existing both as mechanical oscillations in the rotational speed of synchronous generators and as oscillations in the electrical power transfer in transmission lines. It is customary use the term oscillatory *modes*, which refers to how the oscillations not only have a certain *frequency* and *damping*, but are also distributed geographically. The relative amplitudes and phases with which modes are expressed at different locations in the power system are known as *mode shapes*. These terms will be given a more precise definition in Section 2.1.

Oscillations can be divided into natural modes and forced oscillations. Natural modes are properties of the system itself and describes the system's response to disturbances. The expression of natural modes can be further divided into ambient and transient responses. Ambient responses are the system's behaviour under normal operation, when it is continually subjected to stochastic consumer behaviour and variations in generation. Transient responses are the system's reactions to disturbances such as a fault or the loss of a load, a generator or a transmission line [3]. Forced oscillations, on the other hand, are

the result of a periodic input from an external source, such as a malfunction in a generator or a load. A forced oscillation with a frequency close to the frequency of a natural mode may trigger resonance, amplifying the oscillations [4].

The oscillation frequencies of inter-area modes generally lie in the range of 0.1 to 1 Hz. Several such modes are known to exist in the Nordic power system, with the following characteristics [4, 5, 6, 7]:

- A mode with oscillation frequency in the range of 0.3–0.4 Hz, which limits power transmission capacity between Finland and Sweden. This mode has also been involved in resonance events. In this mode, generators in the south of Finland swing against generators separated from them by up to 2000 kilometres of transmission lines, situated in the south of Sweden and Norway.

- A mode with oscillation frequency of around 0.5 Hz, where generators in southern Sweden oscillate against generators in the south of Norway and Finland.

- A mode with oscillation frequency of around 0.8, where generators in the northern areas of Norway and Finland swing against the rest of the Nordic system. This mode limits power transmission from the north to the south during summer. The frequency depends on the topology of the northern grid, which can operated in either a ring configuration or a radial configuration.

Given the potential harm that oscillations can do, it is important to know the system's modal properties. One way is to build a detailed model of the power system, linearise it about a given operating point, and then determine the modes through eigenanalysis of the linearised model's state matrix [8]. It is also possible to infer a model of system dynamics directly from measured system output, with no parameterised models of system components and topology included. Such methods are referred to as measurement-based or data-driven. System linearisations and data-driven methods should be viewed as complementary tools [3].

Observation of inter-area modes with data-driven methods requires measurements taken with a sampling rate comfortably above the oscillation frequencies of interest. For example, discrete Fourier analysis can only identify frequency content below the Nyquist frequency, which is half of the sampling rate [9]. Since mode shapes relate the phase of oscillations, streams of measurements from separate geographical locations must also have a common time reference. Measurements fulfilling both these requirements can be obtained by placing phasor measurement units (PMUs) in power-system substations. PMUs stream out measurements in the form of synchrophasors – phasor measurements synchronised against a fixed time reference. Data streams from PMUs are time-stamped using the Global Positioning System (GPS) at a rate of up to 50 or 60 Hz depending on the nominal frequency of the power system. According to the IEEE standard for PMU data transfer, PMU streams can contain a selected number of phasor measurements (voltage and/or current), and must also include frequency, the rate of change of frequency (rocof), and information about measurement quality and time synchronisation status [10].

An early method capable of modal estimation is Prony analysis, which fits a function consisting of a sum of damped sinusoids to a signal [11]. Another strategy is to fit a time-series model such as ARMAX to the signal. An example of such an algorithm is the Multivariate Autoregressive method (MAR) [12], which has been tested for online modal monitoring of the Nordic power system by Finland's transmission system operator, Fingrid. Many strategies rely on estimating the parameter matrices of a state-space model.

An algorithm of this type is Stochastic Subspace Identification (SSI) [13], which has been tested on Nordic power-system data by Sweden's TSO, Svenska Kraftnät.

Dynamic mode decomposition (DMD) is a modal estimation algorithm which originated in the field of fluid dynamics but has later been applied with success to fields such as neuroscience, epidemiology, finance and video processing. DMD is an equation-free method, in the sense that it does not make any assumption about the underlying system, but relies only on a time-map between system outputs measured at different points in time [14]. It was applied to power-system data in [15] in 2015. A few further studies have been done [16, 17, 18]. The Italian TSO, Terna, has tested DMD for online modal monitoring according to [19].

1.2 Problem description

There are no examples in the literature of DMD used for oscillation monitoring in the Nordic power system. Much of the literature also focuses on analysis of post-disturbance data or large oscillation events [15, 17, 18, 19], while one interesting application of modal analysis algorithm is to track modes when they are still small in amplitude, in order to be able to issue a warning if they become dangerously undamped and start growing. This project aims to make a contribution to the research by applying DMD to data from the Nordic power system, and by testing the algorithm's mode-tracking ability when the system is operating under ambient conditions. Mode estimates will be compared to results from the SSI and MAR algorithms mentioned above.

The random sampling method developed in [20, 21] was applied to power-system data in [18], but only on post-disturbance data, and only on static data windows covering whole events. No examples have been found in the literature of random sampling used for mode-tracking over time, and no examples have been found of random sampling applied to ambient power-system data. These are research gaps which this project aims to address.

DMD performance is dependent on several parameters such as sampling frequency, window length, shift-stacking depth and the number of channels used. The effect of varying some DMD parameters has been tested to a certain degree [16, 22]. However, the noise sensitivity of DMD estimates with respect to all these parameters have not been thoroughly tested. Another important parameter is the rank or system order. Different methods for rank selection have not been sufficiently explored for power-system applications. This project will make investigations into both these matters.

In most of the studies mentioned above, validation of DMD mode estimates is done by comparing with results from other modal estimation methods and with system linearisations. This project takes a systematic approach by first doing statistical investigations into the estimation error of the DMD algorithm and its ability to recognise patterns in data.

1.2.1 Scope

The focus of this work will be to

- implement the DMD algorithm,
- implement add-ons to the DMD algorithm such as shift-stacking, random sampling and rank truncation methods,
- validate that the DMD algorithm is working correctly and gives reasonable estimates for mode frequency, damping and mode shape,

- test the DMD algorithm's sensitivity to noise,
- apply the DMD algorithm to simultaneous mode-tracking on several channels of data,
- apply the DMD algorithm to PMU data from the Nordic power system under ambient operation,
- and compare the DMD algorithm's performance to results from established methods of modal monitoring (SSI, MAR).

The modes of interest will be oscillatory modes with frequency in the inter-area range.

1.2.2 Limitations

The hope is that the contributions from this work can be a step along the way to developing wide-area online modal monitoring for warning and control purposes. However, the following limitations are set for this project:

- Integration with real-time PMU data collection and processing is beyond the scope of the study, so the application will be limited to historical PMU data.
- Strategies for effective handling of data issues such as missing or corrupted data, network delays and cyber-attacks are also outside the scope of the study.
- Code for the SSI and MAR algorithms is provided by RISE and will be used as received. These algorithms will be used for comparison purposes only, and an independent evaluation of their implementation and noise sensitivity are outside the scope of this project.

1.3 Societal, ethical and ecological aspects

The power system is needed for most parts of modern society to function. Power-system stability will be more difficult to achieve in the grid of the future, with more intermittent generation, less inertia and less load frequency dependence. The aim of this study is to contribute to the development of tools to aid power-system monitoring and control. It is hard to imagine any negative societal impacts of improved power-system stability.

No particular ethical issues have been identified with the subject matter of this study or the methods used.

From a climate perspective, we need electrification to achieve the necessary energy transition, and inter-area oscillations are one of several power-system stability challenges in the future grid. From an ecological perspective, power oscillations limit transmission capacity, and being able to use existing transmission lines fully could mean having to build less new infrastructure, which again means less impact on our natural environment. If wide-area modal monitoring can be developed to play a role in acheiving better control over inter-area modes, it will be entirely positive from both these perspectives.

1.4 Thesis structure

Section 2 explains inter-area power oscillations from a physical and mathematical perspective, before introducing the DMD algorithm and giving a brief description of Fourier methods, SSI and MAR. Methods for validation and testing of the DMD algorithm will be described, including generation of synthetic and simulated data.

Section 3 provides validation of the DMD implementation and its mode estimates.

Section 4 contains tests of noise sensitivity, rank reduction methods and mode-tracking capabilities.

Section 5 tests DMD on real-world PMU data from the Nordic power system and compares its results with results from SSI and MAR.

Section 6 contains a discussion about the results and their implications, and suggestions for further work.

Section 7 contains the final conclusion.

Appendix A includes Python code for the implementation of the DMD algorithm used in this project, and a minimal working example of how it can be used.

Appendix B contains details about the implementation of stochastic load variations, including Fortran code for the file-reading PSS/E user-defined load model, and Python scripts for creating random walks and running a simple dynamic simulation using the load model.

2 Methods of modal monitoring

This section begins with a brief description of how oscillations in the power system can be expressed mathematically. The theoretical foundation of the DMD algorithm will be presented, and the concepts of rank truncation, shift-stacking and randomisation will be introduced. Further, a description will be given of Fourier methods along with MAR and SSI, the algorithms against which DMD will be compared. The concept of participation factors and its relevance in the context of DMD will be discussed. Finally, the methods used to validate the algorithm and test its performance will be presented.

2.1 A mathematical model of inter-area power oscillations

Consider a simple single-generator–infinite bus (SGIB) system as in Figure 2.1, as inspired by the similar example in [8]. All losses in the system are neglected. All values except rotor angles are given in per unit on the same base. A power plant is represented by the classical generator model, as a voltage source $E_\mathrm{g}\angle\delta$ behind a transient reactance $X'_\mathrm{d} = 0.25$ per unit. The power grid is represented as an infinite bus, i.e. its voltage and frequency are constant under all circumstances. The power plant is connected to the grid via a step-up transformer with reactance $X_\mathrm{Tr} = 0.1$ per unit and a transmission line with reactance $X_\mathrm{L} = 0.45$ per unit. The voltage is $E_\mathrm{t} = 1\angle\delta_\mathrm{t}$ at the generator terminals and $E_\mathrm{B} = 1\angle 0°$ at the infinite bus. The current flowing from the generator to the grid is

$$I = \frac{E_\mathrm{t} - E_\mathrm{B}}{j(X_\mathrm{Tr} + X_\mathrm{L})}. \tag{2.1}$$

The apparent power output of the generator is

$$S = E_\mathrm{t} I^* = P + jQ, \tag{2.2}$$

where substituting (2.1) for I gives

$$P = \frac{|E_\mathrm{t}||E_\mathrm{B}|}{(X_\mathrm{Tr} + X_\mathrm{L})} \sin\delta_\mathrm{t}, \tag{2.3}$$

$$Q = \frac{|E_\mathrm{t}|^2 - |E_\mathrm{t}||E_\mathrm{B}|\cos\delta_\mathrm{t}}{(X_\mathrm{Tr} + X_\mathrm{L})}. \tag{2.4}$$

The active power transmitted from the generator to the grid is $P = 0.85$ per unit. From (2.3) it then follows that $\delta_\mathrm{t} = \sin^{-1}\left(\frac{(X_\mathrm{Tr}+X_\mathrm{L})P}{|E_\mathrm{t}||E_\mathrm{B}|}\right) = 27.87°$. Inserting δ_t into (2.4) gives the reactive power output $Q = 0.211$ per unit.

Assume that the system is in equilibrium and operating at the nominal frequency of 50 Hz, so that the generator is spinning at the rated synchronous speed of $\omega_0 = 314.2$ electrical rad/s and with an angle δ_0 referenced to the infinite bus. After a disturbance giving a rotor speed deviation of $\Delta\omega = \omega - \omega_0$ in absolute value, the new rotor angle will be $\delta = \delta_0 - \Delta\omega t$. Differentiating twice and expressing in per unit,

$$\frac{\mathrm{d}^2\delta}{\mathrm{d}t^2} = \omega_0 \frac{\mathrm{d}\Delta\omega}{\mathrm{d}t} = \omega_0 \frac{\mathrm{d}\omega}{\mathrm{d}t}. \tag{2.5}$$

When the power system is in equilibrium, the mechanical torque T_m acting on the rotor from the prime mover and the electrical torque T_e on the rotor from the grid are equal. If this equilibrium is disturbed, the relation between the change in electrical torque and

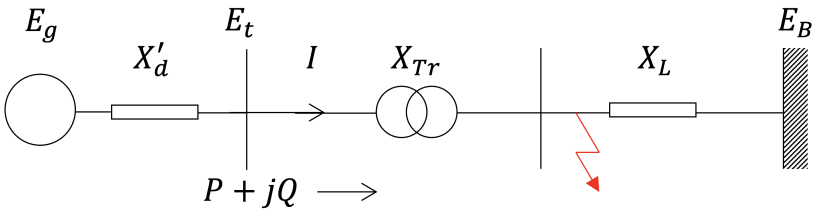


Figure 2.1: A single-generator–infinite-bus system [8].

the excursion in rotor angle is nonlinear. The assumption of small-signal analysis is that changes are small enough that the system can be linearized about the initial operating point. If this holds, (2.3) can be used since $\Delta T_\mathrm{e} = \Delta P$ in per unit for a lossless generator model, so that

$$\Delta T_\mathrm{e} = \frac{\partial T_\mathrm{e}}{\partial \delta}\Delta\delta = \left(\frac{|E_\mathrm{g}||E_\mathrm{B}|}{X'_\mathrm{d} + X_\mathrm{Tr} + X_\mathrm{L}}\cos\delta_0\right)\Delta\delta = K_\mathrm{S}\Delta\delta, \tag{2.6}$$

where K_S is the synchronising torque coefficient.

The kinetic energy stored in the rotating masses of the generator and the turbine is $W_\mathrm{k} = \frac{1}{2}J\omega^2$, where J is the moment of inertia. If there now occurs a disturbance so that there becomes an imbalance between the electrical and the mechanical torque, the generator will accelerate at a rate proportional to this imbalance and inversely proportional to the moment of inertia: $J\frac{\mathrm{d}\omega}{\mathrm{d}t} = T_\mathrm{m} - T_\mathrm{e}$ in absolute values. Linearizing this equation, and expressing J in terms of W_k, and substituting from (2.5) and (2.6), and then dividing by the generator's rated power, gives the swing equation on a per unit basis:

$$\frac{2H}{\omega_0}\frac{d^2\Delta\delta}{\mathrm{d}t^2} = \Delta T_\mathrm{m} - K_\mathrm{S}\Delta\delta - K_\mathrm{D}\Delta\omega, \tag{2.7}$$

where $H = \frac{W_\mathrm{k}}{S_\mathrm{base}}$ is the inertia constant, or the kinetic energy at rated speed in per unit of the generator's rated power. It has the units [MJ/MVA] or [s]. $K_\mathrm{D}\Delta\omega$ is an added damping torque term, where K_D is the damping torque coefficient [8].

Assuming that the generator in this example has no speed governor so that the mechanical torque is constant, this is a linear second-order homogeneous differential equation. It has a solution in the form of the damped oscillations $Ae^{(\sigma\pm j\omega)t}$, where σ is the exponential growth factor, $\omega = \sqrt{1-\zeta^2}\omega_\mathrm{n}$ is the damped frequency in rad/s and ω_n is the undamped natural frequency in rad/s, and the damping ratio is

$$\zeta = \frac{-\sigma}{\sqrt{\sigma^2 - \omega^2}}. \tag{2.8}$$

The damping ratio is often expressed in percent. A damping ratio of $\zeta < 3$ percent is considered low, while a mode with $\zeta > 5$ percent is considered well damped [4].

Equation (2.7) shows how the system's oscillating behaviour is fundamentally linked to the inertia of the rotating mass, represented by H. The oscillations are clearly also shaped by the operating point, represented in (2.6) and (2.7) by ω_0, δ_0, $|E_\mathrm{g}|$ and $|E_\mathrm{B}|$, and the physical characteristics of the generator and the transmission system, represented in simplified form by $K_\mathrm{D}, X'_\mathrm{d}$ and $X_\mathrm{Tr}, X_\mathrm{L}$ respectively. Another important factor is the grid impedance, which is neglected here since the grid is represented by an infinite bus. Rearranging (2.7) in a state-space representation gives

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\Delta\omega\\ \Delta\delta\end{bmatrix} = \begin{bmatrix}-\frac{K_\mathrm{D}}{2H} & -\frac{K_\mathrm{S}}{2H}\\ \omega_0 & 0\end{bmatrix}\begin{bmatrix}\Delta\omega\\ \Delta\delta\end{bmatrix} = A\begin{bmatrix}\Delta\omega\\ \Delta\delta\end{bmatrix}, \tag{2.9}$$

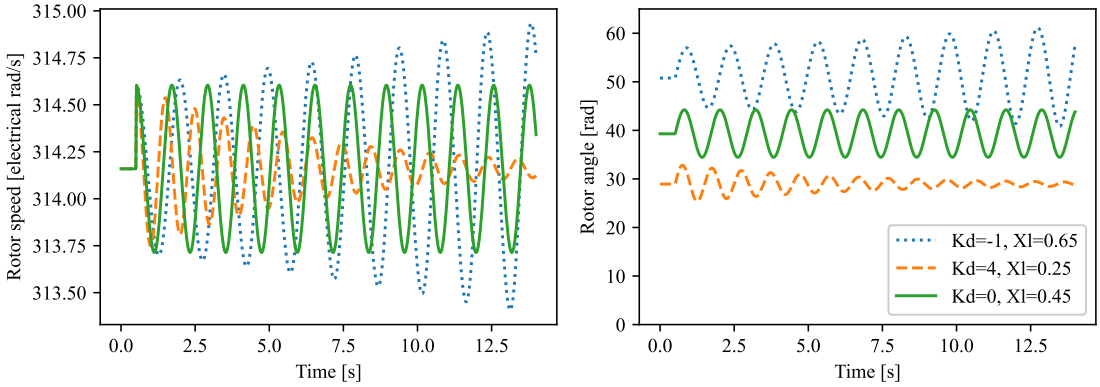


Figure 2.2: Rotor speed (left) and rotor angle (right) response of the SGIB system to a disturbance, with three different combinations of $K_\mathrm{d}, X_\mathrm{L}$.

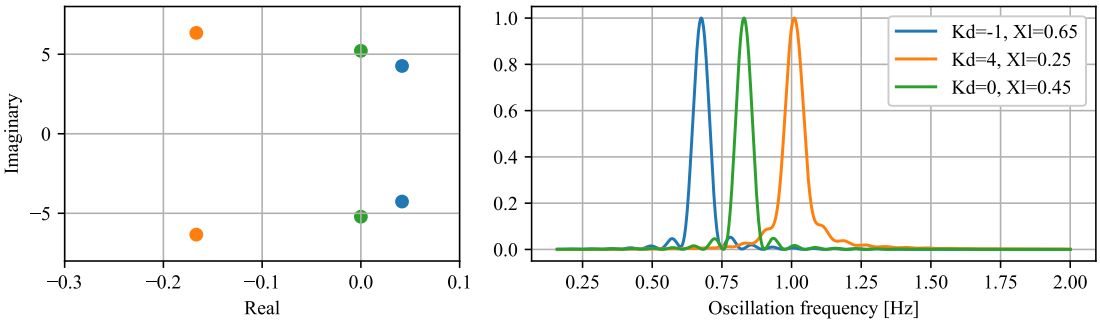


Figure 2.3: Real and imaginary components of the SGIB system's computed eigenvalues (left) and normalised periodograms of the rotor speed response to a disturbance (right), with three different combinations of $K_\mathrm{D}, X_\mathrm{L}$.

where A is the state matrix. The solutions to this system of linear first-order differential equations can be found by taking the eigendecomposition of A, by solving

$$A\mathbf{\Phi} = \mathbf{\Phi}\mathbf{\Lambda}, \tag{2.10}$$

where $\mathbf{\Lambda} = \mathrm{diag}(\boldsymbol{\lambda})$ is a diagonal matrix where $\boldsymbol{\lambda}$ are the eigenvalues of A, and $\mathbf{\Phi}$ is a matrix which columns are the corresponding right eigenvectors. The eigenvalues and their corresponding eigenvectors are the system's *modes* [8]. The eigenvalues are in the form $\lambda_i = \sigma_i \pm j\omega$, and are identical to the solutions to (2.7).

To study the oscillatory properties of the system in Figure 2.1, the initial operating point of the generator can be found using I from (2.1) to calculate the back EMF of the generator as $|E_\mathrm{g}|\angle\delta = E_\mathrm{t} + jX'_\mathrm{d}I = 1.074\angle 39.28°$. By (2.6), the synchronising torque coefficient is then $K_\mathrm{S} = 1.039$ [rad^{-1}]. Let the inertia constant of the generator be $H = 6.0$ s and the damping coefficient be $K_\mathrm{D} = 0$. This gives the eigenvalues $\lambda_1, \lambda_2 = 0 \pm j5.216$ rad/s which corresponds to an oscillation frequency of $f = \frac{\omega}{2\pi} = 0.83$ Hz and a damping ratio of $\zeta = 0$. Table 2.1 shows how the oscillation frequency and damping ratio depend on the generator damping coefficient and the system topology, where the latter is represented here by the line impedance.

A PSS/E simulation of the SGIB system's response to a disturbance was made. Solution parameters are listed in Section 3.2.1. The infinite bus was modelled with infinite inertia and with much higher power rating and much lower reactance than the generator. At time

Table 2.1: Calculated characteristics of the SGIB system with three different combinations of damping coefficient and line impedance

Case	$K_{\mathrm{D}} = -1, X_{\mathrm{L}} = 0.65$	$K_{\mathrm{D}} = 4, X_{\mathrm{L}} = 0.25$	$K_{\mathrm{D}} = 0, X_{\mathrm{L}} = 0.45$
Eigenvalues λ	$0.042 \pm j4.26$	$-0.17 \pm j6.34$	$0 \pm j5.22$
Damping ratio ζ	-0.0098	0.026	0
Oscillation frequency f **[Hz]**	0.68	1.01	0.83

t=0.5 seconds a line fault was applied to the transmission line. The fault was cleared 200 ms later, at which time the operation of the line is restored. Hence the post-fault topology of the system was identical to that of the pre-fault system. Figure 2.2 shows the response of the generator's rotor speed and rotor angle after a disturbance. The simulations were run with three different combinations of K_d, X_L. The results illustrate how negative damping gives an oscillation with an exponentially diverging envelope which eventually will lead to a system collapse. Positive damping means that the system tends towards returning to equilibrium. The impedance of the transmission line impacts stability by affecting the initial rotor angle and the frequency of the oscillations. These results are confirmed by Figure 2.3 (left), where a positive exponential growth factor σ corresponds to the unstable system and a negative σ corresponds to the damped system. The oscillation frequencies indicated by the periodogram in Figure 2.3 (right) match the calculated characteristics as listed in Table 2.1.

2.2 Dynamic Mode Decomposition

DMD is a data-driven, equation-free method of oscillation detection. The method makes no assumption about the system driving the dynamics. DMD relates to the Koopman operator, an infinite-dimensional linear representation of a nonlinear dynamical system. The modes computed by DMD are therefore a finite-dimensional approximation of the Koopman modes, rather than the modes of the state matrix of a linearisation about a given operating point of a nonlinear system model, such as in the SGIB example in Section 2.1 [14]. As such DMD results rely purely on the measurements which make up the data, rather than inferring parameters of an assumed underlying model. Of course, given enough context about the measurements and knowledge about the system from which they were taken, it is still possible to relate the results to parameters and states of this system.

In Section 2.1, the dynamics of the system were described using a continuous-time representation. The corresponding discrete-time representation is

$$\mathbf{x}_{i+1} = \mathbf{A}\mathbf{x}_i, \tag{2.11}$$

where $\mathbf{x}_i$ is a vector of measurements at time t, and $\mathbf{x}_{i+1}$ is a vector of measurements at time $t + \Delta t$ when using a sampling period of Δt, and $\mathbf{A}$ is a discrete-time map. Another way to phrase it is that the vectors $\mathbf{x}_i, \mathbf{x}_{i+1}$ are snapshots of the system's behaviour, sampled at two distinct points in time. It is worth to emphasise that $\mathbf{A}$ in the context of DMD is a discrete-time map, which is not *not* the same thing as the state matrix A of a state-space model such as in Section 2.1. This is an important distinction, which will be discussed in more detail later.

A series of m snapshots from a time-window of length $m\Delta t$ seconds can be organised a data matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m]$. From $\mathbf{X}$ extract two matrices $\mathbf{X}_1, \mathbf{X}_2$ which are overlapping except for the first and last element element of the time-window:

$$\begin{aligned} \mathbf{X}_1 &= [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{m-1}], \\ \mathbf{X}_2 &= [\mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_m]. \end{aligned} \tag{2.12}$$

Assuming that $\mathbf{A}$ is fairly constant throughout the time-window, $\mathbf{x}_1, \mathbf{x}_2$ in (2.11) can be replaced by $\mathbf{X}_1, \mathbf{X}_2$ to give $\mathbf{X}_2 \approx \mathbf{A}\mathbf{X}_1$. To solve for $\mathbf{A}$, a least-squares approximation is used since $\mathbf{X}_1$ is not in general a square matrix:

$$\mathbf{A} \approx \mathbf{X}_2 \mathbf{X}_1^{\dagger}, \tag{2.13}$$

where $[\cdot]^{\dagger}$ is the Moore-Penrose pseudoinverse [23]. Equation (2.11) is a system of first-order difference equations which can be solved by taking the eigendecomposition

$$\mathbf{A}\boldsymbol{\Phi} = \boldsymbol{\Phi}\boldsymbol{\Lambda}. \tag{2.14}$$

The solution to the difference equations is then

$$\mathbf{x}_k = \boldsymbol{\Phi}\boldsymbol{\Lambda}^k \mathbf{b}, \tag{2.15}$$

where $\mathbf{b}$ are the initial conditions expressed on the eigenvector basis so that

$$\mathbf{x}(t_0) = \boldsymbol{\Phi}\mathbf{b}. \tag{2.16}$$

The continuous-time eigenvalues of $\mathbf{A}$ are

$$\boldsymbol{\Omega} = \frac{\ln \boldsymbol{\Lambda}}{\Delta t}, \tag{2.17}$$

where $\boldsymbol{\Omega} = \text{diag}(\boldsymbol{\sigma} + j\boldsymbol{\omega})$, in which $\boldsymbol{\sigma}$ is a vector of the real parts of the continuous eigenvalues, and $\boldsymbol{\omega}$ is a vector of their imaginary parts. Mode frequencies in Hz are then $\mathbf{f} = \frac{\boldsymbol{\omega}}{2\pi}$ and the damping can be found by (2.8)

A flow chart with the steps of the basic DMD algorithm is shown in Figure 2.4. Central to the motivation behind DMD is that the dynamics of a system will often be dominated by a small number of modes. An example of this could be a power system where an inter-area mode has been excited and is expressed in large parts of the system, while other modes may be insufficiently excited to rise over the level of measurement noise. Say that the data contains n channels of measurements, and the system expresses n_{osc} oscillatory modes in the time-window analysed. If n exceeds $2n_{\text{osc}}$ (twice the number since modes come in conjugate pairs) plus whatever non-oscillatory modes are needed to account for offsets from zero in the signals, the rest of the eigenvalues and their corresponding eigenvectors will be spurious modes representing noise or modal content too weak to separate from the noise. Hence it is not needed to find the whole set of eigenvalues and eigenvectors, only the ones of interest. The eigenvalues and eigenvectors of interest will necessarily be the parts of the signal carrying the most energy.

In a signal processing context, *power* is defined as the amplitude squared [9].

Instead of analysing the full time-map $\mathbf{A}$, the eigenanalysis can be performed on a reduced-rank matrix $\tilde{\mathbf{A}}$ [14]. The first step in constructing $\tilde{\mathbf{A}}$ is to use a tool from linear

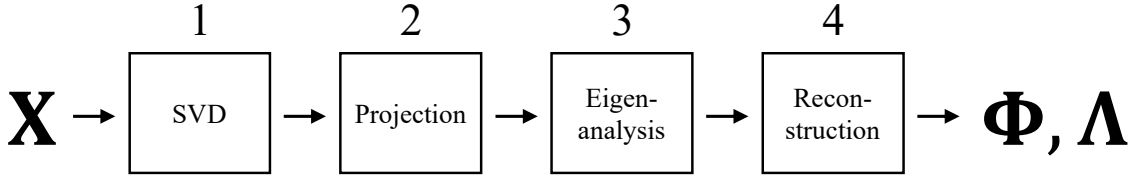


Figure 2.4: The basic DMD algorithm.

algebra called the singular value decomposition (SVD). SVD factorises $\mathbf{X}_1$ into three matrices:

$$\mathbf{X}_1 = \mathbf{U}\mathbf{S}\mathbf{V}^{\mathrm{T}}, \tag{2.18}$$

where $\mathbf{S}$ is a diagonal matrix with the singular values $\mathbf{s}$ on the diagonal [23]. $\mathbf{U}, \mathbf{V}$ are orthonormal matrices containing the left and right singular vectors respectively. The usefulness of SVD lies in that it finds correlations in the data. The singular values are by convention sorted in descending order. Large singular values represent dominating structures in the data. Hence an approximation of $\mathbf{X}_1$ can be made by getting rid of the smallest singular values and truncating the SVD to rank k by setting

$$\begin{cases} \mathbf{U} = \mathbf{U}[:, : k], \\ \mathbf{S} = \mathbf{S}[: k, : k], \\ \mathbf{V} = \mathbf{V}[:, : k]. \end{cases} \tag{2.19}$$

This gives the rank-k approximation

$$\mathbf{X}_1 \approx \mathbf{U}[:, : k]\, \mathbf{S}[: k, : k]\, \mathbf{V}[:, : k]^{\mathrm{T}}, \tag{2.20}$$

which keeps actual dynamic content in the data and gets rid of noise. From now on the indices $[\cdot, \cdot]$ will be dropped for clarity, so $\mathbf{U}, \mathbf{S}, \mathbf{V}$ will designate the truncated singular vectors and values as described by (2.19). The fact that $\mathbf{U}, \mathbf{V}$ are orthonormal can be used to express (2.13) as $\mathbf{A} = \mathbf{X_2}\mathbf{V}\mathbf{S}^{-1}\mathbf{U}^T$, removing the computationally costly matrix inversion of $\mathbf{X}_1$.

Further, the rank-k map $\tilde{\mathbf{A}}$ can now be computed by projecting $\mathbf{A}$ onto the column-space of $\mathbf{U}$:

$$\tilde{\mathbf{A}} = \mathbf{U}^T\mathbf{A}\mathbf{U} = \mathbf{U}^T\mathbf{X_2}\mathbf{V}\mathbf{S}^{-1}\mathbf{U}^T\mathbf{U} = \mathbf{U}^T\mathbf{X_2}\mathbf{V}\mathbf{S}^{-1}, \tag{2.21}$$

since $\mathbf{U}^T\mathbf{U} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. Eigenanalysis of $\tilde{\mathbf{A}}$ gives $\tilde{\mathbf{A}}\mathbf{W} = \mathbf{W}\mathbf{\Lambda}$, where $\mathbf{\Lambda}$ are the eigenvalues and $\mathbf{W}$ are right eigenvectors. Eigenvalues of $\tilde{\mathbf{A}}$ are also eigenvalues of $\mathbf{A}$ [14]. The eigenvectors of the full system can then be reconstructed as follows:

$$\mathbf{\Phi} = \mathbf{\Lambda}^{-1}\mathbf{X}_2\mathbf{V}\mathbf{S}^{-1}\mathbf{W} \approx \mathbf{U}\mathbf{W}, \tag{2.22}$$

where the two definitions will converge to the extent that the column spaces of $\mathbf{X}_1, \mathbf{X}_2$ are equal, such as in the case of time-series data [24].

2.2.1 Strategies for rank truncation

The rank truncation step described above has several benefits. First, it lightens the computational burden of the eigendecomposition by decreasing the matrix on which it is performed, thereby increasing the speed of the algorithm. Importantly, it also provides noise-filtering. The challenge is to estimate the correct rank, since setting it either too high or too low is detrimental to the performance of the algorithm. If k is set too high, some of the energy belonging to actual dynamic modes may be ascribed to spurious modes, decreasing the accuracy of mode estimates. If k is set lower than the actual rank of the dynamics, each of the algorithm's mode estimates will have to account for the energy of more than one dynamic mode, which means missing some dynamic modes and polluting the remaining estimates. A simple example is a rank-1 system with one oscillatory mode. Since describing an oscillation mode requires two eigenvalues and eigenvectors, the algorithm will miss the oscillation and wrongly estimate this as just an exponential growth or decay.

The most basic strategy for selection k is to simply fix it at a value low enough to give satisfactory performance while still minimising the risk of missing modes. However, there exist strategies of adaptive rank-selection. Ideally, most of the energy in the signal is contained in a few modes, in which case there is a sharp fall between the first few dominating singular values representing the actual dynamic content and the remaining singular values which represent the noise.

One strategy is to truncate where the cumulative sum of the singular values exceeds some preset value, which guarantees that at least a given percentage of the variance in the signal remains after truncation [14, 17]. The problem is that the estimated rank will be very high if the singular value spectrum is slowly decaying, as it often is with real-world data. If a relatively small maximum rank is imposed, the rank selected by the cumulative sum method usually will exceed this maximum, making the cumulative sum method equivalent to just selecting a fixed rank [17].

Another strategy is to find the largest drop in singular values and truncate there [13], with the argument that this must be where the dynamical content stops and the noise begins. This is not necessarily true if the signal contains a large offset from zero, such as with a power-system frequency signal where the common-mode oscillation can contain much more energy than inter-area oscillations. A minimum rank can be specified as a workaround for this issue, but the method would then depend on there being a second large drop in the singular value spectrum in order to give a meaningful result.

Instead of relying on a sharp drop in the singular values, the rank can be set to exclude singular values below a certain threshold. One way is to experimentally tune the threshold on observations from the singular value spectrum in historical data, which assumes that the data will have a fairly constant noise level. A more sophisticated strategy is found in [25], where the threshold τ is estimated from the data:

$$\tau \approx (0.56\beta^3 - 0.95\beta^2 + 1.82\beta + 1.43)s_{0.5} \tag{2.23}$$

where $\beta = \frac{n}{m}$ for a matrix with n rows and m columns, and $s_{0.5}$ is the median of the singular values. This is an approximation of an asymptotic optimum based on the Marcenko–Pastur distribution. The rank k is set equal to the number of singular values exceeding τ.

2.2.2 Shift-stacking

As mentioned above, the rank of the time-map must be high enough to account for all significant dynamical modes, which in the case of oscillatory modes come in conjugate pairs. The data must also be sufficiently rich in dynamical structure, otherwise the quality of estimates will suffer. If the dimensionality of the data is too low, an augmentation technique referred to as shift-stacking can be used [14]. Similar to what is known as a Hankel matrix, slices of the rows of the data matrix $\mathbf{X}$ are stacked so that

$$\begin{bmatrix} x_1 & x_2 & \dots & x_m \end{bmatrix} \longrightarrow \begin{bmatrix} x_1 & x_2 & \dots & x_{m-b} \\ x_2 & x_3 & \dots & x_{m-b+1} \\ \vdots & \vdots & \ddots & \vdots \\ x_b & x_{d+1} & \dots & x_m \end{bmatrix}. \tag{2.24}$$

where x_j is an element in row i of $\mathbf{X}$, and b is the number of rows to stack. No new information is added to the data set in this way, but by increasing the number of rows from n to bn, existing structure is reinforced by repetition, thereby improving mode estimates. There is a trade-off between this reinforcement and the shortening of the effective window

of analysis from m to $m - b$ samples, which means there may exist an optimal shift-stacking depth for a given data set. In this project the shift-stacking depth will be given as a percentage of the original matrix width (the number of snapshot vectors in the time-window of analysis). The number of columns in the stacked data matrix is $m_s = (1-a)m$ and the number of rows is $n_s = amn$ for a shift-stacking depth of $a\,\%$.

2.2.3 Randomisation

While shift-stacking can give greatly improved mode estimates, it increases the computational burden by inflating the size of the data matrix from $n \times m$ to $n_s \times m_s$. One solution to this problem is a method of randomisation introduced in [20] and adapted to DMD in [21], and tested for power-system oscillation detection in [18]. This method can lighten the computational burden significantly by performing the SVD on a low-dimensional approximation $\mathbf{H} \in \mathbb{R}^{l \times m_s}$ instead of on the full shift-stacked data matrix $\mathbf{X} \in \mathbb{R}^{n_s \times m_s}$, with the possibility of setting $l \ll n_s$. The trick is to find a near-optimal basis $\mathbf{Q}$ for $\mathbf{X}$ so that

$$\mathbf{X} \approx \mathbf{Q}\mathbf{Q}^{\mathrm{T}}\mathbf{X} = \mathbf{Q}\mathbf{H}. \tag{2.25}$$

Again, the assumption is of a high-dimensional system which dynamics can be almost fully explained by a small number of dominating modes. Determining $\mathbf{Q}$ starts with projecting the data onto a random matrix $\mathbf{G} \in \mathbb{R}^{m_s \times l}$:

$$\mathbf{Y} = \mathbf{X}\mathbf{G}. \tag{2.26}$$

In this project, the elements of $\mathbf{G}$ are drawn from the standard normal distribution, but it is also possible to use other types of random matrices. The projection of $\mathbf{X}$ onto $\mathbf{G}$ means that the columns of $\mathbf{Y}$ are linear combinations of the columns of $\mathbf{X}$, with random coefficients given by the columns of $\mathbf{G}$. In other words, $\mathbf{Y}$ randomly samples the column space of $\mathbf{X}$. To increase the probability of capturing all the structure in $\mathbf{X}$, a small amount of oversampling can be used so that $l = k + p$, where k is the assumed rank of $\mathbf{X}$ and p is the number of extra samples.

The QR-decomposition $\mathbf{Y} = \mathbf{Q}\mathbf{R}$, which decomposes the matrix $\mathbf{Y}$ into an upper triangular matrix $\mathbf{R}$ and an orthonormal matrix $\mathbf{Q}$ [23], then provides the orthonormal basis $\mathbf{Q} \in \mathbb{R}^{n_s \times l}$. It follows from (2.25) that

$$\mathbf{H} = \mathbf{Q}^T\mathbf{X}. \tag{2.27}$$

A bonus feature of the randomisation method is that a slowly decaying singular value spectrum can be made steeper by a technique called power iterations, which involves sampling $(\mathbf{X}\mathbf{X}^T)^q\mathbf{X}$ instead of $\mathbf{X}$, since $(\mathbf{X}\mathbf{X}^T)^q\mathbf{X} = \mathbf{U}\mathbf{S}^{2q+1}\mathbf{V}^{\mathrm{T}}$. The following iterative method for finding $\mathbf{Q}$ is more numerically stable than performing repeated matrix multiplications:

$\mathbf{Y} = \mathbf{X}\mathbf{G}$
for $j = 1, ..., q$
    $[\mathbf{Q}, \sim] = \mathrm{QR}(\mathbf{Y})$
    $[\mathbf{Z}, \sim] = \mathrm{QR}(\mathbf{X}\mathrm{T}\mathbf{Q})$
    $\mathbf{Y} = \mathbf{X}\mathbf{Z}$
end
$[\mathbf{Q}, \sim] = \mathrm{QR}(\mathbf{Y})$.

where QR$(\cdot)$ is the QR decomposition. Optimisation of the parameters p, q is beyond the scope of this project. Per recommendations in [21], they have been set to $p = 10, q = 2$ for all tests using randomised DMD described in this report. Experiments with perturbations of these parameters show no significant gain in the accuracy of DMD mode estimates by increasing p, q beyond this level, but a risk of losing accuracy if these parameters are decreased, which indicates that they have been set at a reasonable level.

After computing $\mathbf{H}$ by (2.27), DMD can be performed as described in section 2.2, letting $\mathbf{H}_1, \mathbf{H}_2$ segmented as in (2.12) replace $\mathbf{X}_1, \mathbf{X}_2$ in every step. Finally, full-dimensional eigenvalues can be reconstructed by inserting (2.25) into (2.22):

$$\mathbf{\Phi} = \mathbf{\Lambda}^{-1}\mathbf{Q}\mathbf{H}_2\mathbf{V}\mathbf{S}^{-1}\mathbf{W}. \tag{2.28}$$

2.2.4 Mode tracking

Tracking modes in time can be done by gliding the window of analysis over a longer time-series, using a given update interval. A known mode can be tracked by setting a target frequency f_{target}, and then for every time step finding the estimated mode with the closest matching frequency. For every update, the following steps are taken:

1. Perform DMD.
2. Using the continuous-valued eigenvectors computed by (2.17), find candidate modes with oscillation frequencies $f_{\text{cand}} \in [f_{\min}, f_{\max}]$ Hz, where $f_{\min}, f_{\max}$ delimit the frequency band of interest. Since oscillatory modes come in conjugate pairs, it is only necessary to look at positive frequencies.
3. Compute the absolute difference $|f_{\text{cand}} - f_{\text{target}}|$ between all target modes and all candidate modes, and choose the pairs with the minimum difference.
4. Discard any pairs where the difference exceeds the tolerance f_{tol}.
5. The remaining candidates are considered expressions of the target mode they are paired with. Save their eigenvalues λ, eigenvectors ϕ and mode amplitudes b. Target frequencies not matched with a candidate are considered not present at this point in time.

This is similar to the tracking strategies used in the received implementations of the SSI and MAR algorithms described in Sections 2.3.2 and 2.3.3, with the improvement that several modes can be tracked simultaneously.

2.2.5 Algorithm development and programming in Python

Programming of the DMD algorithm was done in the Python language. The development of the present algorithm implementation was undertaken as follows: First, a basic DMD version for analysis of a static window of data was written based on the Matlab code examples in [14]. Next, functionality for shift-stacking was included [14, 18]. A version of randomised DMD was made based on [18] and [21]. Methods of rank truncation were implemented based on [13, 14, 17, 25]. Finally, a script for mode-tracking over time was written.

Each step of development was thoroughly tested using randomised synthetic data generated as described in Section 2.5.1, and the methods described in Sections 2.5.2, 2.5.3 and 2.5.6.

Functions imported from from the Numpy [26] library were used for linear algebra operations such as matrix inversions, SVD, eigenanalysis and QR decomposition, and also for random number generation and general mathematical programming. Handling of input and output data was done with the Pandas package [27]. Signal processing tasks such as filter design, filtering and downsampling were done using tools from the Scipy package [28]. Visualisations of results were done using the Matplotlib environment [29].

2.3 Other methods for detecting oscillations

Other methods used to detect oscillations in this project are Fourier methods, SSI and MAR. The theoretical background of these methods is briefly explained below.

2.3.1 Fourier methods

Fourier methods for observing oscillations originate from the continuous Fourier transform:

$$X(f) = \int_{-\infty}^{+\infty} x(t)e^{-j2\pi ft}\mathrm{d}t. \tag{2.29}$$

For signals sampled at constant intervals in time, there is a discrete-time equivalent known as the discrete Fourier transform (DFT):

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j\frac{2\pi}{N}kn}, \tag{2.30}$$

where $k = \frac{f}{f_\mathrm{r}}$, with f being the oscillation frequency and f_r the fundamental frequency of the transform. Analysis of a time-series of N samples returns the frequency content at N equally spaced points in the frequency spectrum [9].

A periodogram is a plot showing the power spectral density (PSD) $P_x(k)$ of a signal, defined as

$$P_x(k) = \frac{1}{N}\left|X(k)\right|^2, \tag{2.31}$$

for $X(k)$ defined as in (2.30). Periodograms of real-world signals can be messy due to noise and changing frequency content, especially if N is large. A way to smooth out the periodogram is to use Welch's method, which divides the time series into many smaller windows and takes the average of periodograms computed for all these windows. It is customary to let the windows overlap, for example by half the window length. Multiplying the windows with a windowing function before computing the periodograms helps reduce distortion due to discontinuities at the start and end of the windows. One such function is the Hann window $w(k) = 0.5 + 0.5\cos\frac{2\pi k}{M}$ for a window length of M, which is the windowing function used in this project [30].

Periodograms only contain information in the frequency domain, and say nothing about when an oscillation appears or disappears, or if its amplitude changes during the time window analysed. A method for analysing non-stationary signals is the short-time Fourier transform (STFT), which in its discrete form glides a window over the signal and computes the DFT with regular intervals. The assumption is that the window is short enough that the signal is approximately stationary within it. The definition of the discrete STFT with a window length M is

$$X(n,k) = \sum_{m=0}^{M-i} x(n+m)w(m)e^{-j\frac{2\pi}{M}kn}, \tag{2.32}$$

where again a windowing function $w(m)$ is used to reduce distortion. [31]. The PSD for each time step n can then be computed by (2.31). The evolution of the frequency spectrum over time can then be visualised with what is often called a spectrogram – a plot with frequency on the y axis and time on the x axis, and the PSD either on a third axis or represented with a colour map.

A limitation of both Welch's method and the discrete STFT is that there necessarily is a trade-off between time resolution and frequency resolution. A data window of length T_{window} seconds, sampled with a sampling rate of f_{s}, has

$$M = f_{\text{s}} T_{\text{window}} \tag{2.33}$$

elements. The fundamental frequency f_r of the transform is

$$f_{\text{r}} = \frac{f_{\text{s}}}{M}. \tag{2.34}$$

Since all frequency outputs are integer multiples of the fundamental frequency f_{r}, it follows that f_{r} is also the frequency resolution of the transform. Now inserting (2.33) into (2.34), it is evident that

$$f_{\text{r}} = \frac{1}{T_{\text{window}}} \tag{2.35}$$

In other words, the frequency resolution is inversely proportional to the length of the time window, independently of the sampling rate. Hence it is possible to get good time resolution – i.e. use short time windows – or good frequency resolution, but not both at the same time. Another important limitation of Fourier methods in the context of modal analysis is that they only estimate oscillation frequency and not damping.

Tools from the Scipy [28] package were used to perform Fourier methods in this project.

2.3.2 Stochastic subspace identification

The SSI method is similar to DMD in that it is a data-driven method relying on singular value decomposition for noise-filtering. A difference is that it assumes a system model

$$\begin{aligned} \mathbf{x}_{i+1} &= \mathbf{A}\mathbf{x}_i + \mathbf{w}_i, \\ \mathbf{y}_i &= \mathbf{C}\mathbf{x}_i + \mathbf{v}_i, \end{aligned} \tag{2.36}$$

where $\mathbf{A}$ is the state matrix and $\mathbf{C}$ is the output matrix, and $\mathbf{v}_i, \mathbf{w}_i$ are noise terms. SSI exists in two versions, of which SSI-cov calculates the covariance matrix explicitly and SSI-data acts directly on the output data. The version used here is SSI-cov as described in [13]. It is defined as follows: Let $\mathbf{Y}$ be a data window of size $[n, m]$. Its covariance matrix is calculated recursively, using sets of past and future values $\mathbf{y}_i^-, \mathbf{y}_i^+$ defined so that

$$\begin{aligned} \mathbf{y}_i^+ &= [\mathbf{y}_i, \mathbf{y}_{i+1}, ..., \mathbf{y}_{i+I}]^{\text{T}}, \\ \mathbf{y}_i^- &= [\mathbf{y}_i, \mathbf{y}_{i-1}, ..., \mathbf{y}_{i-I}]^{\text{T}}, \end{aligned} \tag{2.37}$$

where I is the covariance matrix window size parameter. The covariance matrix window for time i is then computed by taking

$$\mathbf{H}_i = \mathbf{y}_i^+ \mathbf{y}_i^{-\text{T}}. \tag{2.38}$$

The covariance matrix $\mathbf{H}$ for the whole data window $\mathbf{Y}$ is then found by gliding $\mathbf{H}_i$ over all $i \in [I, m-I]$ with an update interval R, and summing the results. SVD is then performed on $\mathbf{H}$, and the singular values are truncated in the same way as for DMD to form a rank-k

approximation $\mathbf{H} \approx \mathbf{USV}^T$. The extended observability matrix $\mathbf{O}$ can then be obtained from the truncated SVD results:

$$\mathbf{O} = \begin{bmatrix} \mathbf{CA}^0 \\ \vdots \\ \mathbf{CA}^k \end{bmatrix} = \mathbf{US}^{\frac{1}{2}}. \tag{2.39}$$

Now defining $\mathbf{O_1}, \mathbf{O_2}$ as in (2.12) and $\mathbf{o}_j$ as the jth column of $\mathbf{O}$, estimates of the state and output matrices are:

$$\mathbf{A} = \mathbf{O}_1\mathbf{O}_2^\dagger, \tag{2.40}$$

$$\mathbf{C} = [\mathbf{o}_1, ..., \mathbf{o}_n]. \tag{2.41}$$

Now eigenanalysis of $\mathbf{A}$ can be performed by solving $\mathbf{AW} = \mathbf{W\Lambda}$, after which the system mode shapes can be found by the projection $\mathbf{\Phi} = \mathbf{CW}$ and continuous-time eigenvalues can be determined by (2.17).

Parameters which need to be tuned to track modes over time are the target frequency f_{cand} and frequency tolerance f_{tol}, the data window length n, the analysis update interval, the target rank k, and the window length and update rate parameters I, R for the recursive covariance matrix computation.

2.3.3 Multivariate autoregressive method

The MAR algorithm uses the standard autoregressive time-series model, extended to multiple channels. Hence the value of a channel at a given time depends not only on its own past values, but the past values of the other channels too. A MAR model of the dynamics in a data window $\mathbf{Y}$ set with size $[n, m]$, is defined as:

$$\mathbf{y}_i = \boldsymbol{\kappa} + \sum_{j=1}^{k} \mathbf{A}_j \mathbf{y}_{j-i} + \boldsymbol{\varepsilon}_i, \tag{2.42}$$

where k is the model order, which determines how many past values or *lags* the present value depends on. This model is then fit to the data window $\mathbf{Y}$, which means finding the vector of constant offsets $\boldsymbol{\kappa}$, the coefficient matrices $\mathbf{A}_i$ and the white noise term $\boldsymbol{\varepsilon_k} \sim N(0, \sigma_k)$ which minimises prediction error [12]. Model fitting in the received code was done with a Matlab package described in [32]. After model fitting, the coefficient matrices can be assembled into a larger matrix of the form

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & \cdots & \cdots & \mathbf{A}_k \\ \mathbf{I} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \mathbf{I} & 0 \end{bmatrix}. \tag{2.43}$$

Eigenanalysis of the matrix $\mathbf{A}$ will then give the poles of the system, from which the frequency and the damping can be computed in the usual way. A disadvantage of this algorithm is that it does not output mode shapes in the same way as SSI or DMD, which means that the amplitude of oscillations cannot be monitored, only their frequency and damping [6].

Algorithm parameters which need to be tuned to track modes over time are minimum and maximum values of the target frequency, the data window length n, the analysis update interval, and the system order k.

2.4 Eigenvectors and participation factors

Eigenvectors are not unique and can be scaled by a constant and still be eigenvectors. For any real or complex diagonal matrix $\mathbf{D}$ and the diagonal eigenvalue matrix $\mathbf{\Lambda}$, it holds that $\mathbf{\Lambda} = \mathbf{D}\mathbf{\Lambda}\mathbf{D}^{-1}$. By (2.14) it then follows that

$$\mathbf{A} = \mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Phi}^{-1} = \mathbf{\Phi}\mathbf{D}\mathbf{\Lambda}\mathbf{D}^{-1}\mathbf{\Phi}^{-1} = (\mathbf{\Phi}\mathbf{D})\mathbf{\Lambda}(\mathbf{\Phi}\mathbf{D})^{-1}, \tag{2.44}$$

which proves that $\mathbf{\Phi}\mathbf{D}$ is an eigenvector of $\mathbf{A}$ whenever $\mathbf{\Phi}$ is an eigenvector of $\mathbf{A}$. The output of an eigenanalysis function in a programming language depends on the convention of normalisation used. The ordering of eigenvalues and the angles and magnitudes of eigenvectors can therefore change depending on the choice of rank, and also on numerical differences between two windows analysed, for example because of noise. The relationship given by (2.44) means that it is always possible to express mode shapes in relation to a reference bus. This enables comparisons of mode shapes in the noise sensitivity analyses in Section 4.1.

Participation factors in the context of a linear state-space model are a measure of the relationship between modes and states. Right eigenvectors are associated with a mode's observability, while the left eigenvectors are associated with a mode's controllability. In the definition used by [8], the activity of each state variable in each mode is identified by the right eigenvectors $\mathbf{\Phi}$, while the left eigenvectors Ψ weigh the relative contribution of each state variable to each mode. The eigenvectors are normalised so that $\mathbf{\Psi}\mathbf{\Phi} = \mathbf{I} \iff \mathbf{\Psi} = \mathbf{\Phi}^{-1}$. The participation matrix $\mathbf{P}$ is defined such that the participation factor of state i in mode j is

$$p_{ij} = \phi_{ij}\psi_{ji} \tag{2.45}$$

which can also be interpreted as the sensitivity of the eigenvalue j to element i on the diagonal of the state matrix [8].

The above is true for a linearised state-space model, but will an identical interpretation hold for a time-map which relates only to the data and assumes no parameterised system model, and which is based on a set of signals which may be much fewer in number than the amount of states in a full system model? Having computed the left and right eigenvectors $\mathbf{W}^{-1}, \mathbf{W}$ of $\tilde{\mathbf{A}}$, reconstructing the full-dimensional DMD modes is unproblematic: $\mathbf{\Phi}$ can be found by (2.22), and

$$\mathbf{\Psi} = \mathbf{W}^{-1}\mathbf{U}^T, \tag{2.46}$$

as noted in [24], since $\mathbf{A} = \mathbf{U}\tilde{\mathbf{A}}\mathbf{U}^T = \mathbf{U}\mathbf{W}\mathbf{\Lambda}\mathbf{W}^{-1}\mathbf{U}^T = \mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Psi}$.

However, $\mathbf{A} = \mathbf{A}_k$ is only possible if $\tilde{\mathbf{A}}$ has the same rank as $\mathbf{A}$. If rank reduction is used, it is necessarily only possible to reconstruct $\mathbf{U}\tilde{\mathbf{A}}\mathbf{U}^T = \mathbf{A}_k$, which has the same rank as $\tilde{\mathbf{A}}$. The relationship $\mathbf{X}_2 \approx \mathbf{A}\mathbf{X}_1 \approx \mathbf{A}_k\mathbf{X}_1$ still holds, since both $\mathbf{A}$ and $\mathbf{A}_k$ capture the dynamics of the system to an extent. The eigenvalues and right eigenvectors representing the dominant modes can be approximately equal, but not in general identical since in general $\mathbf{A}_k \neq \mathbf{A}$. The question then is how sensitive the participation matrix $\mathbf{P}$ is to numerical differences and rank changes.

In contrast to the model-focused definition described above, Barocio et al. [15] view participation factors in relation to the data, defining it as the contribution of each mode to how the output of the channels evolves in time:

$$\mathbf{P} = |\mathbf{\Phi}\mathbf{\Lambda}\text{diag}(\mathbf{b})| \tag{2.47}$$

Comparing to (2.58) reveals that this quantity is related to the oscillation magnitude. The differences between these two definitions of the participation matrix will be investigated in Section 3.2.4.

2.5 Methods of algorithm validation

In the literature, several types of methods have been used to validate mode estimates made by algorithms:

- Comparing the estimated modes with known modes, for example determined by system linearisation tools [8, 19, 33], or by solving the governing equation of a simple linear system [8].
- Comparing results with results from other methods and algorithms [6, 15, 16, 19].
- Checking how well the algorithm's results can recreate the original signal [18].

In this project, all these three techniques were used to validate the DMD algorithm, both considering implementation and results. Below follows a description of how these methods were applied, and how data for algorithm validation was generated. A discussion about these validation methods and their strengths and weaknesses is included in Section 6.

2.5.1 Generating synthetic data for algorithm testing

Synthetic data sets were created with the purpose of having signals which properties were known with absolute certainty. Each of n signals consists of a sum of n_{osc} sinusoids with exponential envelopes:

$$x_i(t) = C_{\text{corr}} \sum_{j=1}^{n_{\text{osc}}} \mathbf{1}_{i,j} C_{i,j} e^{\sigma_j t} \cos(\omega_j t + \theta_{i,j}). \tag{2.48}$$

Inputs are the number of signals n, the oscillation frequencies ω_j and the exponential growth factors σ_j. The indicator $\mathbf{1}_{i,j} \in \{0, 1\}$ is chosen so that each signal contains a minimum of two of the modes, selected at random. The statistical expectation of the number of modes in each signal is $\mathrm{E}\left[\sum_{j=1}^{n_{\text{osc}}} \mathbf{1}_{i,j}\right] = \frac{2+n_{\text{osc}}}{2}$. Each oscillatory component's amplitude $C_{i,j} \in [0.1, 1]$ and phase $\theta_{i,j} \in [-\pi, \pi]$ are chosen randomly from uniform distributions. The reason for specifying a minimum amplitude is because this data will be used to test the algorithm's ability to recognise known patterns in the data, not to study how close to numerical precision it can operate. To make the average signal strength independent of the number of oscillatory components specified, a correction factor $C_{\text{corr}} = \frac{2\sqrt{2}}{0.55(2+n_{\text{osc}})}$ is included so that the an undamped signal will have an expected rms value

$$\mathrm{E}\left[x_i^{\text{rms}}(t) \middle| \, \sigma_j = 0\right] = C_{\text{corr}} \sum_{j=1}^{n_{\text{osc}}} \mathrm{E}\left[\mathbf{1}_{i,j}\right] \mathrm{E}\left[C_{i,j}\right] = 1. \tag{2.49}$$

This expectation is valid for an individual signal viewed independently. Further, each mode was set to be present in at least one signal. All parameters are constant throughout the duration of the constructed time series, so the synthetic data set has a constant discrete time-map in the form described in Section 2.2. This means that the eigensystem can be related directly to the input parameters of the synthetic data: oscillation frequency, damping, amplitude and phase. The following derivations concentrate on one single signal component and the subscripts i, j are therefore omitted for clarity.

2.5.2 Frequency and damping estimates

Let $y(t)$ be an oscillatory component of a signal in a synthetic data set constructed as described in Section 2.5.1. Then, according to (2.17) and (2.58), the signal can be reconstructed in the form of $y_{\text{DMD}}(t) = f(\sigma, \omega, \phi, b)$ where $\{\omega, \phi, b\}$ are one of the conjugate set of oscillation frequencies, eigenvector elements and mode amplitudes estimated by DMD for this mode. The original signal component, on the other hand, was created in the form of $y_{\text{data}}(t) = g(\sigma, \omega, \theta, C)$. The parameters of both $y_{\text{data}}(t)$ and $y_{\text{DMD}}(t)$ describe the characteristics of the oscillatory component completely and equivalently. Hence DMD estimates of σ, ω can be validated by comparing them directly with the corresponding data parameters. This also validates the damping ratio estimate since ζ is a function of σ, ω.

2.5.3 Eigenvectors

The relation between ϕ, b and θ, C can be found by comparing the two expressions of $y(t)$:

$$\begin{aligned} y_{\text{data}}(t) &= Ce^{\sigma t}\cos(\omega t + \theta) \\ &= Ce^{\sigma t}\left[\cos\theta\cos\omega t - \sin\theta\sin\omega t\right] \end{aligned} \tag{2.50}$$

$$\begin{aligned} y_{\text{DMD}}(t) &= \phi b e^{(\sigma+j\omega)t} + (\phi b)^* e^{(\sigma-j\omega)t} \\ &= e^{\sigma t}\left[\text{Re}\{\phi b\}\left(e^{j\omega t} + e^{-j\omega t}\right) + j\text{Im}\{\phi b\}\left(e^{j\omega t} - e^{-j\omega t}\right)\right] \\ &= 2e^{\sigma t}\left[\text{Re}\{\phi b\}\cos\omega t - \text{Im}\{\phi b\}\sin\omega t\right] \end{aligned} \tag{2.51}$$

Setting $y_{\text{data}}(t) = y_{\text{DMD}}(t)$, the exponential term $e^{\sigma t}$ is present on both sides of the equality sign and can be eliminated, and it is clear that $2\text{Re}\{\phi b\} = C\cos\theta$ and $2\text{Im}\{\phi b\} = C\sin\theta$. This further means that $2\phi b = C(\cos\theta + j\sin\theta)$, so that we can estimate signal phase and amplitude to be

$$\begin{cases} \theta_{DMD} &= \angle(\phi b) = \angle\phi + \angle b \\ C_{DMD} &= 2|\phi b| = 2|\phi||b| \end{cases} \tag{2.52}$$

which provides validation for the DMD eigenvector estimates.

2.5.4 Using simulation data from models with known properties

While synthetic data generation provides an infinite number of different test cases on which results can be validated, these cases lack dynamical aspects of a power system. The simulation software PSS/E was used to construct data containing power-system dynamics, where signals represent system states at buses in the system. To provide validation for the results from DMD, a simple power-system model with known modes was used. Preliminary tests were done on the SGIB system described in section 2.1 where the characteristics can be calculated since the dynamics can be represented by a second-order differential equation. To elicit a modal response, a branch fault was applied for 200 ms before being cleared. This leaves the post-fault system configuration identical to the pre-fault system configuration and exerts a shock similar to an impulse response test, so that the system is expected to express oscillation and damping very close to the computed solutions in Table 2.1.

Further tests were performed on the IEEE Two-Area Four-Generator benchmark model [34], which will from now on be referred to as the Two-Area model. It is a PSS/E implementation of a model which was thoroughly analysed in [8], so that modal properties are

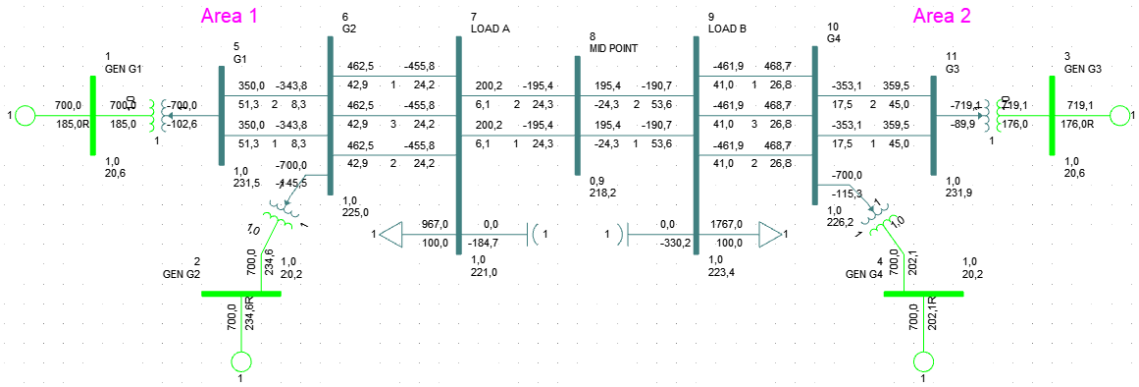


Figure 2.5: PSS/E single-line diagram of the IEEE Two-Area benchmark model [34].

known. Figure 2.5 shows a single-line diagram of the Two-Area model drawn in PSS/E. The nominal frequency in this model is 60 Hz. There is a synchronous generator at each of the system's four 20 kV buses. These buses are connected via step-up transformers to the transmission system, which has seven 230 kV buses, of which two are equipped with one load and one shunt capacitor each. This system has a weak inter-area connection and is designed to have one inter-area mode and two local modes. In this case a bolted branch fault could not be used, since it made the system so unstable that the simulation solver would not converge. Instead a step in the voltage reference for all the generators was used, identical to that in [34], designed to excite both the local modes and the inter-area mode. This disturbance does not change the topology of the system, but it does change its operating point somewhat. It is important here to note that while linearised modal analysis will reveal important aspects of the system's stability, it cannot predict exactly the modal properties of the system's reaction to any given disturbance. The system's initial response can be highly nonlinear, which may also affect how much different modes are excited. This will be further explored in Section 3.2.3. Simulation settings are listed in Table 3.2.

2.5.5 Estimating properties with peak finding and curve fitting

If a system is simple enough, modal parameters can be determined directly from post-disturbance simulation output using basic mathematical methods. A requirement for getting valid results is that any initial nonlinear response after a disturbance must have settled, so that most of the energy in the signals is concentrated in a small number of dominating modes. According to [3], a useful rule of thumb for large transients is to exclude the two first swings. Since the system's behaviour in such a case is approximately linear, it can be expressed as a sum of sinusoids with exponential envelopes. If the frequencies of these sinusoids are reasonably far apart, the modes can be separated by filtering. In any window of time where the assumptions of linearity and separability hold, the contribution from a given oscillatory mode can be expressed in the form:

$$Ce^{\sigma t}\cos(\omega t + \theta), \tag{2.53}$$

where σ is the exponential growth rate, ω is the oscillation frequency in rad/s, θ is the oscillation's phase and C is a real constant. Note that this is also the motivation for Prony analysis, which was briefly mentioned in Section 1.1. However, the assumption of only one oscillatory component in the filtered signal allows a much simpler approach: If the times

t_1, t_n of the first and last of a series of n_{peaks} are found, the oscillation period p can be determined simply by

$$p \approx \frac{t_n - t_1}{n_{\text{peaks}} - 1}. \tag{2.54}$$

The oscillation frequency is then $f = \frac{1}{p}$ Hz or $\omega = \frac{2\pi}{p}$ rad/s.

The damping can be then determined by curve fitting to the envelope of the oscillation. Let $\mathbf{t}_{\text{peaks}}$ be an array of the times of all the peaks within a chosen window. The peaks lie along the upper envelope of the oscillation and occur where $\omega t + \theta = 0$. This reduces (2.53) to

$$\mathbf{x}_{\text{peaks}} = \pm C e^{\sigma \mathbf{t}_{\text{peaks}}}, \tag{2.55}$$

where $\mathbf{x}_{\text{peaks}}$ are the magnitudes of the peaks. If the number of peaks in the window exceeds two, (2.55) is an overdetermined set of equations with unknowns C, σ. After taking the logarithm of both sides to make the system linear, it can be solved using linear regression:

$$\begin{bmatrix} \ln C \\ \sigma \end{bmatrix} \approx \begin{bmatrix} \mathbf{1} & \mathbf{t}_{\text{peaks}} \end{bmatrix}^{\dagger} \ln(|\mathbf{x}_{\text{peaks}}|) \tag{2.56}$$

where $\mathbf{1}$ is a vector of ones with the same length as $\mathbf{x}_{\text{peaks}}$. The damping ratio can then be calculated using (2.8). Further, it follows from 2.53 that

$$\theta \approx -\frac{\sum \omega t^{+}_{\text{peaks}} + \sum(\omega t^{-}_{\text{peaks}} - \pi)}{n_{\text{peaks}}}, \tag{2.57}$$

where $t^{+}_{\text{peaks}}, t^{-}_{\text{peaks}}$ are the times of the positive and negative peaks in the window, respectively. The method described above will from now on be referred to as «peak finding». Note that using peak finding for modal parameter determination is valid only under the strict assumptions made above, and its application under these specific circumstances in this project does no imply that modal analysis is in general equivalent to curve fitting.

2.5.6 Reconstructing data

An easy way to evaluate DMD results is to reconstruct signals using the DMD estimates and compare the reconstructions to the original signals. This can be done by using the continuous-time equivalent of 2.15:

$$\mathbf{x}(t) = \mathbf{\Phi} \mathrm{e}^{\mathbf{\Omega} t} \mathbf{b} \tag{2.58}$$

The evaluation can be done by using some metric such as the mean squared error (MSE) to quantify the difference between the data and the reconstruction:

$$MSE = \frac{1}{M} \sum_{i=1}^{M} (x_i - \hat{x}_i)^2, \tag{2.59}$$

where M is the number of data points compared, and $x_i, \hat{x}_i$ are arrays of original data points and estimates, respectively. Plotting a signal from the data together with the reconstructed signal based on DMD modal estimates gives a visual impression of the quality of the reconstruction. A limitation of this validation method is that any linear modal method can only express its results in the form of sums of complex exponentials. Any other signal structure, such as noise, offsets and nonlinear dynamics, can therefore only be approximated by complex exponentials or sums of complex exponentials. In cases where there is a lot of non-periodic structure in the data, this means that the rank of the system must be increased significantly in order to build a faithful copy of the original signal, if at all possible.

2.6 Testing algorithm performance

2.6.1 Noise level

A way to specify the level of noise in a signal is to use the signal-to-noise ratio (SNR)

$$SNR = 10 \log \frac{P_{\text{signal}}}{P_{\text{noise}}} \tag{2.60}$$

$$P = \frac{1}{m} \sum_{i=1}^{m} x_i^2 \tag{2.61}$$

for a signal of m samples [9]. Brown et al. [35] noted that a range of 10-100 dB had been assumed for the noise level of PMU measurements in the literature, and concluded based on empirical PMU data that a SNR of 45 dB is a good approximation. However, since SNR is a ratio, it depends a lot on what is considered to be the signal.

Figure 2.6 shows the power spectral density of frequency deviation data recorded with a sampling frequency of 50 Hz by a PMU situated in the power system lab at Chalmers during the month of January 2016. This signal was retrieved from a data set which will be described in more detail in Section 5. The regular periodogram gives a picture of the entire frequency content, while the averaging in the Welch periodogram emphasises periodic structure and reduces random noise. The power spectral density varies a lot over the frequency spectrum. Looking first at the Welch curve, the flattening toward the higher end of the frequency spectrum likely indicates the noise floor or level of measurement noise [9]. The spiky appearance in both graphs in this region indicates that there is not only white noise here, i.e. there exists some correlation in time, which may be related to other types of noise originating from the distribution grid. If the entire frequency deviation is considered as the signal, the very dominating ultra-low frequency common-mode oscillations with a frequency of around 0.05 Hz and below dominate. With this frame of reference, the highest noise peaks in the regular periodogram are separated from the signal by two orders of magnitude, or an SNR of 20. However, what is important in the context of modal estimation is how large oscillations in the range of interest must be to be visible in a signal contaminated with noise. The Welch periodogram has several bumps in the inter-area range. With reference to these oscillations, for example those around 0.4 Hz, the SNR is negative.

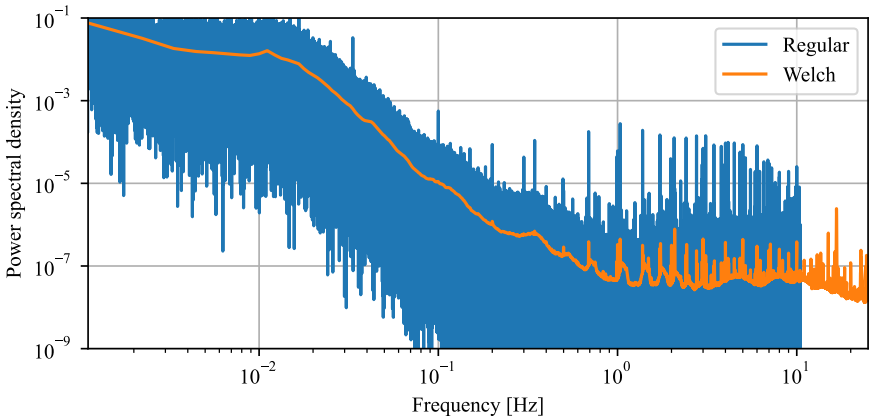


Figure 2.6: Regular periodogram and Welch periodogram (window length 15 minutes) of one month of frequency data from the Chalmers PMU.

In this project, SNR is used to describe the strength of white noise applied to synthetic or simulated signals. The frame of reference is the frequency range of inter-area oscillations, so that P_{signal} is the power of signal content in the range from 0.1 Hz and above, and the noise is an additive Gaussian white noise process Z_i with zero mean and a constant standard deviation σ_{noise} for all $i \in \{1, 2, ..., m\}$. Signal power can then be found by applying (2.61) to the signal after high-pass filtering. Equation (2.61) is also recognisable as $P = \mathrm{E}[X^2] = \sigma^2_{\text{noise}}$, so that for a given SNR,

$$\sigma_{\text{noise}} = 10^{-\frac{\text{SNR}}{20}} \sqrt{P_{\text{signal}}}. \tag{2.62}$$

2.6.2 Estimation error

One way to quantify the error of DMD estimates is MSE, as mentioned in Section 2.5.6. However, MSE is only valid for real numbers. A metric for estimation errors of complex-valued data is the total vector error (TVE):

$$TVE = \sqrt{\frac{(\mathrm{Re}\{\hat{x}\} - \mathrm{Re}\{x\})^2 + (\mathrm{Im}\{\hat{x}\} - \mathrm{Im}\{x\})^2}{\mathrm{Re}\{x\}^2 + \mathrm{Im}\{x\}^2}} = \frac{|\hat{x} - x|}{|x|} \tag{2.63}$$

where $x, \hat{x}$ are the original data point and the estimate, respectively. This is the metric used to quantify phasor measurement error in IEEE's PMU measurement standard [36]. As is clear from (2.63), TVE is the Euclidean distance between the data point and its estimate, normalised by the magnitude of the data point. It is a dimensionless quantity.

2.6.3 Simulated data with stochastic load perturbations

To investigate the modal properties of a system model under ambient operation, stochastic load perturbations can be used to excite the natural modes of the system. This strategy mimics how real-world frequency changes are driven by constant imbalances between generation and load due to the aggregated behaviour of power consumers and producers. One alternative is to model the load variations by adding a white noise to loads in a power-system model [12, 13]. However, it is fair to ask whether it is the demand itself which should be modelled as a random variable, or rather the *change* in demand. A white noise has by definition a flat power spectral density. A random walk, which is the integral of a white noise, has a power spectral density with the characteristic k/f^2, where k is some constant and f is the frequency [37]. Note the shape of the Welch periodogram of the Chalmers PMU frequency deviation in Figure 2.7. There are resonant bumps corresponding to system dynamics, notably the common mode in the ultra-low frequency range and inter-area oscillations in the 0.2-1 Hz range, and the curve levels off when the noise floor is reached, as discussed above. However, the general tendency in the region above the noise floor follows the k/f^2 characteristic, which is included in the plot as a dotted line, which suggests that the system is perturbed by something more akin to an integrated noise process than a noise process. A random walk was therefore chosen as the model of load perturbation. By the central limit theorem, aggregated actions of a large number of individual customers will approach a normal distribution, so a Gaussian white noise was used to model the load change:

$$S_n^+ = \sum_{i=1}^{n} \varepsilon_i, \tag{2.64}$$

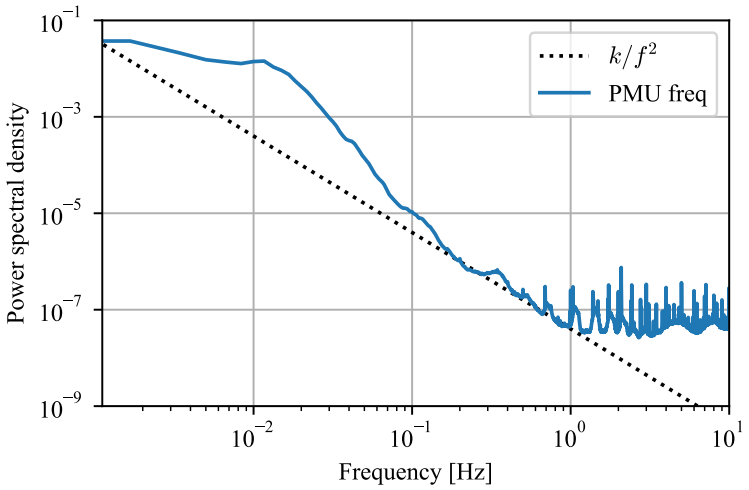


Figure 2.7: Power spectral density of one month of frequency data from the Chalmers PMU.

where S_n^+ is the magnitude of the apparent power perturbation at simulation step n, and $\varepsilon_i \sim N(0, \sigma_\varepsilon) \forall i$ is a Gaussian random variable with zero mean and standard deviation σ_ε. This random walk must then be scaled to give perturbations of desired magnitude. Note that the standard deviation $\sigma_{S_n^+}$ of the random walk at any step n is related to the standard deviation of the Gaussian random variable by

$$\sigma_{S_i^+} = \sigma_\varepsilon \sqrt{i}. \tag{2.65}$$

The reason is that $\mathrm{E}[\sigma_\varepsilon] = 0$, so by using the linearity of the expectation [38] it can be shown that $\sigma^2_{S_n^+} = \mathrm{E}\left[(\sum \varepsilon_i)^2\right] - \mathrm{E}\left[\sum \varepsilon_i\right]^2 = \sum_{i=1}^n \mathrm{E}[\varepsilon^2] = n\sigma_\varepsilon^2$.

To implement load perturbations in dynamic simulations with the Two-Area system, a user-defined load model for PSS/E was written in Fortran. The load model was designed to read a time-series from file and add it to whatever load was specified in the load-flow case. Hence the apparent power of the load at any step n of the simulation is

$$S_n = S_n^{\mathrm{ZIP}} + S_n^+, \tag{2.66}$$

where $S_n^{\mathrm{ZIP}} = P_n^{\mathrm{ZIP}} + jQ_n^{\mathrm{ZIP}}$ is the original load given in the load-flow case, divided into constant admittance, constant current and constant power parts as specified during conversion for dynamic simulation. The model is constructed so that the voltage dependency of S_n^{ZIP} is maintained throughout the simulation.

The input file for the added perturbation is opened during initialisation of the simulation. For every simulation step i, a new value S_n^{file} is read from the file and scaled with a scaling parameter σ to give $S_n^+ = \sigma S_n^{\mathrm{file}}$ in per unit on the system base. A power factor parameter $\cos\varphi$ allows $P_n^+ + jQ_n^+ = \cos\varphi S_n^+ + j\sqrt{1 - \cos^2\varphi}$ to be computed. Both real and reactive parts of the added load perturbation is defined to have a constant power characteristic. A parameter t_{start} regulates the starting-time of the perturbations, so that the simulation can be run «flat» for a desired amount of time prior to perturbation to check that the system is initially converged.

Further details about the stochastic load input, including dynamics data specifications and Fortran code, are included in Appendix B.

3 Algorithm validation

The aim of this section is to validate that the DMD algorithm was correctly implemented and gives reasonable estimates. Apart from transforming frequency deviation signals from per unit to Hz, no preprocessing or filtering of the data was done before applying DMD.

3.1 Validating DMD using synthetic data

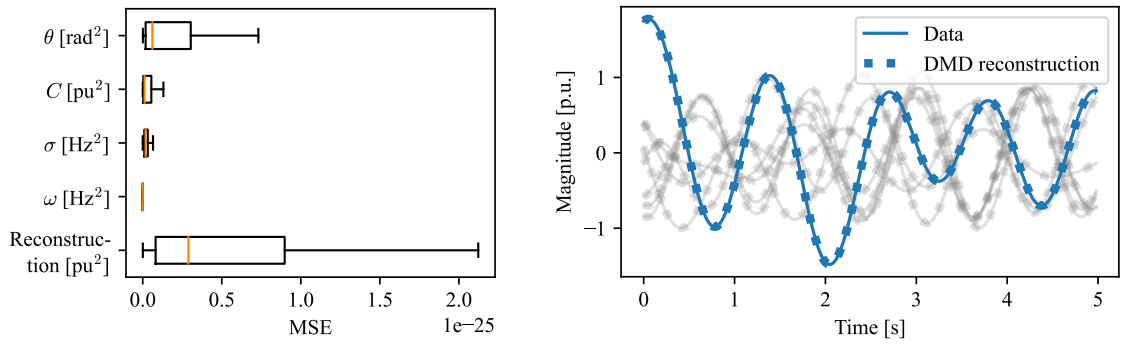
The intention in this round of tests was to investigate the DMD algorithm's ability to identify known data structures correctly. This was done both to confirm that the algorithm was implemented without mistake, and to evaluate the accuracy of its estimates. Modal analysis was done on 5-second windows of data containing 8 signals, where each signal contained at least 2 of a total of 4 oscillatory modes. The data sets were created as described in (2.5.1), with a sampling rate of 50 Hz.

To gain a solid statistical basis to conclude from, results were based on output from 10000 iterations of analysing 5 seconds of newly generated data, introducing as much randomness as possible was within the scope of the test: For each iteration, a new set of oscillation frequencies and exponential growth factors were chosen from uniform random distributions such that $\boldsymbol{\sigma}_{\text{data}} \in [-0.5, 0.5]$ rad/s and $\boldsymbol{\omega}_{\text{data}} \in [0.2\pi, \pi]$ rad/s, with a minimum difference between oscillation frequencies of 0.02π rad/s. The minimum difference was set to ensure that what was tested was the algorithm's ability to accurately identify modes known to exist in the data, and not how closely spaced modes it is able to separate. The randomness introduced here comes in addition to the randomness already present in the data generation function as defined in Section 2.5.1. Assuring in this way that the DMD algorithm was presented with a completely new data set for each iteration of the test eliminates any false advantage gained from inadvertently selecting a trivial test case.

Since the aim was to test the algorithm's ability to recognise known structures under optimal conditions, no noise was added to the data. For the same reason, and to eliminate any impact from rank selection strategies from the test, the rank was fixed to $k = 8$, which is the exact rank needed to estimate the 4 oscillatory modes. Because shift-stacking will be included in further tests, it was included here to validate correct implementation. A shift-stacking depth of 20 percent was used as a default value based on preliminary experiments. Shift-stacking would not be expected to make any significant difference in estimates under optimal conditions such as in this test, so any errors in the implementation would result in errors in the DMD estimates.

For each run, the full set of signals were reconstructed using (2.58), and compared to the data set using MSE. The DMD estimates $\{\boldsymbol{\omega}_{\text{DMD}}, \boldsymbol{\sigma}_{\text{DMD}}, \boldsymbol{\Phi}, \mathbf{b}\}$ were compared with the data input parameters $\{\boldsymbol{\omega}_{\text{data}}, \boldsymbol{\sigma}_{\text{data}}, \boldsymbol{\Theta}_{\text{data}}, \mathbf{C}_{\text{data}}\}$. The estimates $\{\boldsymbol{\Theta}_{\text{DMD}}, \mathbf{C}_{\text{DMD}}\}$ were calculated from $\{\boldsymbol{\Phi}, \mathbf{b}\}$ by (2.52) in order to allow direct comparisons. Again MSE was used to quantify the estimation error.

The box plot in Figure 3.1a shows the range of the mean squared error of the DMD estimates. A direct comparison between the different quantities is not possible since they have different magnitude ranges. However, in all cases the order of magnitude of the MSE reflects that all absolute errors are close to the machine precision of $\approx 10^{-16}$, which means that they are practically zero. The time series plot in Figure 3.1b depicts a randomly chosen case and demonstrates that all 8 signals could be reconstructed from the DMD estimates. Likewise the estimated mode shapes in Figure 3.2 could be related to the amplitude and phase of the signal. The reason why the mode shapes do not have the same number of elements is that 2.48 picks a random number of modes to be present in each



(a) Estimation errors. **(b)** Reconstructions of all 8 signals.

Figure 3.1: DMD validation results. Note the estimation errors in the order of magnitude 10^{-25}.

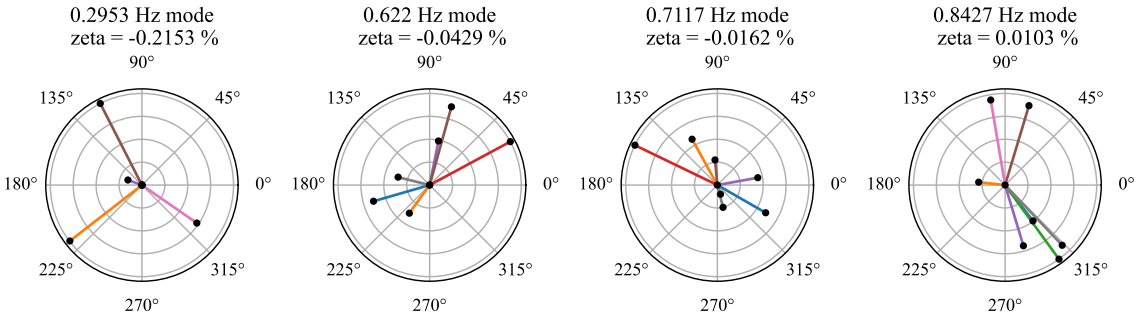


Figure 3.2: Estimated mode shapes (coloured lines) and their relation to signal amplitude and phase in the data (black dots). Estimated frequencies and damping were identical to data inputs.

signal, so that not all signals participate in each mode. The results presented here are valid also for damping ratio $\boldsymbol{\zeta}$ and the participation matrix $\mathbf{P}$, since these quantities are functions of the compared parameters, which completely describe a linear system as long as the rank is correct.

3.2 Validating DMD using simulation data

The synthetic data used so far has the advantage that all quantities are completely known. A disadvantage is that these quantities do not represent any actual dynamics, and that this data is ideally linear, contrary to data from power-system simulations or real-life power-system data. Therefore the next step is to test the algorithm on power-system simulation data, using system models which are simple enough that the output can be interpreted clearly.

3.2.1 SGIB model

DMD was tested on output from PSS/E simulations with the SGIB system shown in Figure 2.1. Solution parameters for these simulations are listed in Table 3.1. The signal used for analysis was the frequency deviation in Hz, which differs from the rotor speed shown in Figure 2.2 (left) only by magnitude and offset. DMD oscillation frequency and damping estimates were identical to the computed values in Table 2.1 for all three cases. The

Table 3.1: PSS/E dynamics solution parameters for SGIB simulation.

Argument	Parameter	Value
INTGAR(1)	Network solution maximum number of iterations	99
REALAR(1)	Acceleration factor	1
REALAR(2)	Convergence tolerance	0.0001
REALAR(3)	Simulation time step (DELT)	0.01 s
REALAR(4)	Filter time constant for bus frequency deviations	0.04 s

peak-finding method described in Section 2.5.5 also gave identical results. The oscillation frequency results were further confirmed by FFT, see Section 2.3.1.

3.2.2 Two-Area model

PSS/E simulations were run using the Two Area model. The frequency deviation output was used as the signal for analysis. A simultaneous step in the voltage references of all generators, designed to excite both the inter-area mode and the local modes of the system as described in [34], was applied at $t = 10$ seconds:

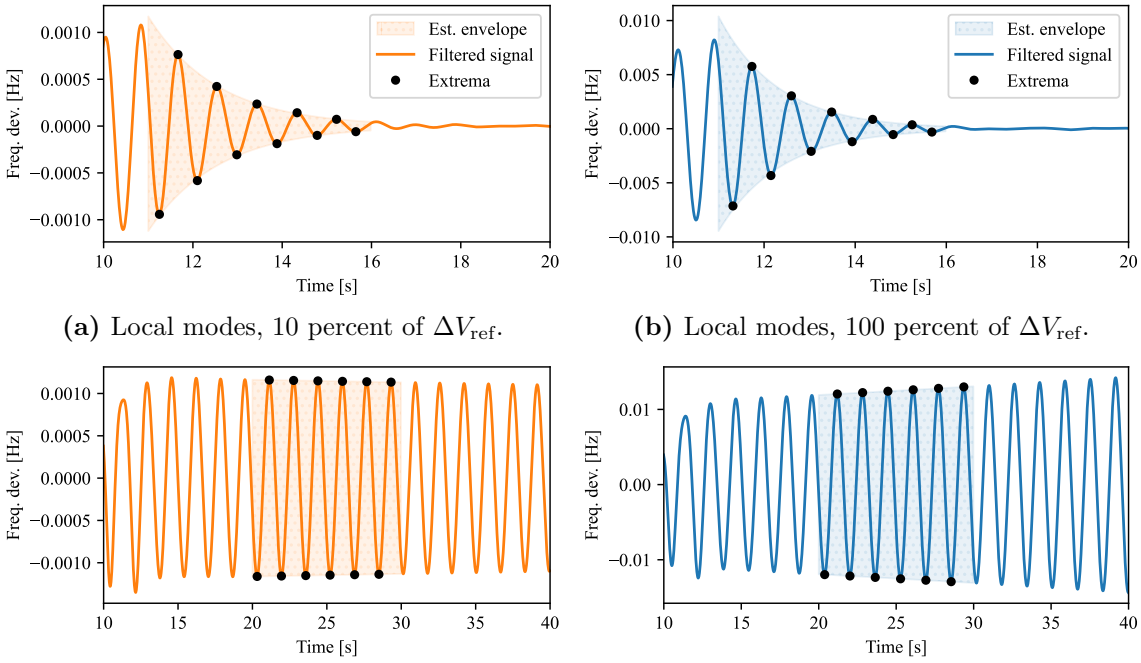
$$\Delta V_{\text{ref}} = \begin{cases} 0.03 \text{ p.u. for generator } G_1, \\ -0.01 \text{ p.u. for generator } G_2, \\ -0.03 \text{ p.u. for generator } G_3, \\ 0.01 \text{ p.u. for generator } G_4, \end{cases} \tag{3.1}$$

This disturbance is used for the remainder of this project. In sections below, scaling of this disturbance will be referred to as e.g. 10 percent of ΔV_{ref}, which means that the voltage reference step i $10\% \cdot 0.03 = 0.003$ per unit at generator G_1 and so on for the rest of the generators. Initial experiments showed that the inter-area mode of the Two-Area system exhibited different damping ratios when disturbances of different magnitudes were applied. Results in this section are given for both 10 percent and 100 percent of ΔV_{ref} below to illustrate this.

To separate out modes for the peak-finding method, a Butterworth band-pass filter of order 2 was applied, with forward-backward filtering to get zero-phase filter response. A minimum of 10 seconds of data at either end of the filtered signal was omitted from estimates to avoid corruption due to boundary effects of the filter. The filter's passband was set to $f_c = \{0.3, 1.0\}$ Hz to analyse the inter-area mode and $f_c = \{1.0, 1.5\}$ Hz to analyse the local modes. To reduce errors from any remaining trend or nonlinear components not fully eliminated by the filter, the mean of estimates of upper and lower envelopes were used. Estimates ω_{Peaks} and $\sigma_{\text{Peaks}}, C_{\text{Peaks}}$ using (2.54) and (2.56) respectively were made

Table 3.2: PSS/E dynamics solution parameters for Two-Area simulation.

Argument	Parameter	Value
INTGAR(1)	Network solution maximum number of iterations	99
REALAR(1)	Acceleration factor	1
REALAR(2)	Convergence tolerance	0.0001
REALAR(3)	Simulation time step (DELT)	0.003333 s
REALAR(4)	Filter time constant for bus frequency deviations	0.033333 s



(a) Local modes, 10 percent of ΔV_{ref}.

(b) Local modes, 100 percent of ΔV_{ref}.

(c) Inter-area mode, 10 percent of ΔV_{ref}.

(d) Inter-area mode, 100 percent of ΔV_{ref}.

Figure 3.3: The peak-finding method's envelope and frequency estimates for the local modes and the inter-area mode after a disturbance of 10 and 100 percent of ΔV_{ref}, using filtered signals.

for each signal in the data (all buses in the system), and the mean of these was used as the final estimate. The standard deviation was also computed to assess the uncertainty of the data. Finally θ_{Peaks} was found using (2.57). The filtered signals together with the exponential envelopes determined by the peak-finding method are shown in Figure 3.3.

For the local modes, a 5-second time-window starting at $t = 11$ seconds was chosen, since this was the only window where peak-finding was possible to use because the mode decayed so fast, as seen in Figure 3.3a and b. It was not possible to find a long enough time-window to get reasonable results with FFT for these modes. A rank of $k = 20$ was used for DMD since there is a lot going on in this time-window, judging by Figure 3.4 (right). For the longer-lasting inter-area mode, a time-window starting at $t = 20$ seconds and with a duration of 10 seconds was used. In this time-window, there seems to be just one dominating oscillation, so the rank was lowered to $k = 7$.

Table 3.3 summarises the estimated modes of the system exposed to 100 percent of ΔV_{ref} by using DMD, the peak-finding method, analytical linearisation [8] and FFT. The two local modes cannot be filtered apart before peak-finding because they have nearly the same frequency and damping, so the result for peak-finding is for a combination of these two modes. The 1.134 Hz mode found by DMD had much larger amplitude for generators G_1 and G_2 than for generators G_3 and G_4, and the mode angle between these two generators was close to 180 degrees, confirming that this is the local mode where generators in area A swing each other. Conversely, the 1.159 Hz mode had a mode shape dominated by generators 3 and 4 swinging against each other, confirming that this is the local mode of area B. This is consistent with the linearisation in [8], where the area B mode has a slightly higher frequency than the area A mode. For both modes, DMD estimates a slightly lower frequency and a slightly higher damping than predicted by the linearisation

Table 3.3: Local modes of the Two-Area system estimated with various methods.

	DMD	Peaks	Linearisation [8]
$\sigma + j\omega$ [rad/s]	$-0.683 \pm j7.126$ $-0.595 \pm j7.284$	$-0.625 \pm j7.179$	$-0.49 \pm j7.15$ $-0.496 \pm j7.35$
f [Hz]	1.134 1.159	1.142	1.14 1.17
ζ [%]	9.536 8.143	8.725	7 7

Table 3.4: Inter-area mode of the Two-Area system estimated with various methods.

	DMD	Peaks	Linearisation [8]	FFT
$\sigma + j\omega$ [rad/s]	$0.009 \pm j3.846$	$0.009 \pm j3.846$	$0.031 \pm j3.84$	
f [Hz]	0.612	0.612	0.61	0.62
ζ [%]	-0.239	-0.239	-0.8	

in [8]. The result from the peak-finding method is close to an average of the DMD results for the two modes, which is expected since it describes a combination of the two modes.

Oscillation frequency estimates for the inter-area mode in Table 3.4 show even closer agreement between DMD and the peak-finding method. The results of the peak finding method were likely more precise here than for the local modes because of the longer time window, and also the difference between the natural frequency and the damped frequency is insignificant when the damping ratio is so small. The FFT result is based on a 50 second window and therefore has frequency increments of 0.02 Hz, which means that it confirms the DMD result within what is possible with this resolution. The frequency in Kundur's linearisation was given with only two decimals, which again confirms the DMD result within the precision used.

The damping that the inter-area mode expresses in the simulation output for a disturbance magnitude of 100 percent of ΔV_{ref} is much less negative than what was predicted by the linearisation. As noted in Section 2.5.4, linear modal analysis cannot exactly predict a nonlinear system's behaviour after a disturbance. Both short-term and longer-term behaviour depends on the system's initial nonlinear response, which affects how much different modes are excited. The initial nonlinear response, in turn, depends on the nature

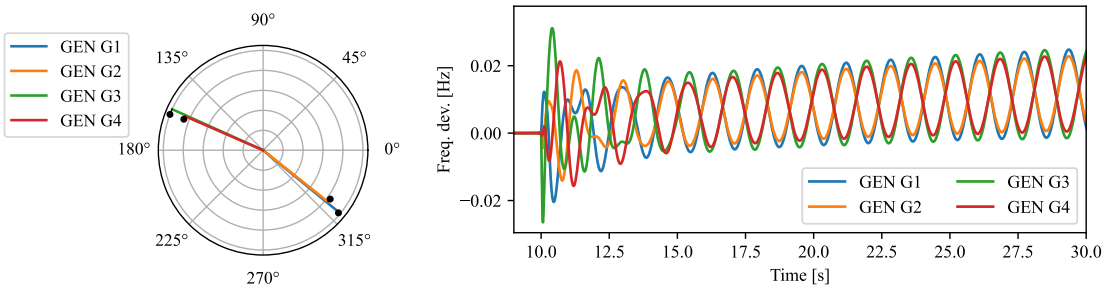


Figure 3.4: Left: Inter-area mode shapes estimated by DMD (coloured lines), compared with oscillation magnitude and phase estimated by peak finding (black dots). Only the four generator buses are included. The disturbance is 100 percent of ΔV_{ref}. Right: Unfiltered frequency deviation.

and the magnitude of the disturbance applied. This is clearly visible in Figures 3.3 c) and d): For 10 percent of ΔV_{ref}, the damping actually changes sign and becomes slightly positive. For 100 percent of ΔV_{ref}, the signal is slightly undamped, in accordance with the DMD and peak-finding method results in Table 3.4. In both cases the plots provide visual confirmation that the exponential envelopes are correctly determined by the peak-finding method, and therefore that the DMD damping estimates are correct, since they are practically identical.

The DMD mode shape estimates in Figure 3.4a agree quite well with the oscillation magnitudes and phases determined by the peak-finding method. Note that DMD estimates are based on eigenvectors of a discrete time map considering all signals, while the result from the peak-finding method is based on individual phase estimates for each signal, which can account for the small difference in mode angle. A visual inspection of Figure 3.4b that generators G_1, G_2 swing against generators G_3, G_4 almost in counter-phase, with generators G_1, G_3 having somewhat higher amplitude than G_2, G_4, which agrees well with the estimated mode shapes.

3.2.3 The effect of disturbance magnitude on damping

To further investigate the effect of disturbance magnitude, a parameter sweep over ΔV_{ref} was performed. Figure 3.5 shows the oscillation frequency and damping ratio estimated by DMD when ΔV_{ref} was varied from 0.5 percent to 250 percent of the original steps given above. The figure shows only results for the inter-area mode.

There are several interesting aspects of these results. First, estimates from DMD and the peak-finding method follow each other closely, even toward the lower range of disturbance magnitude where the curves start getting jagged.

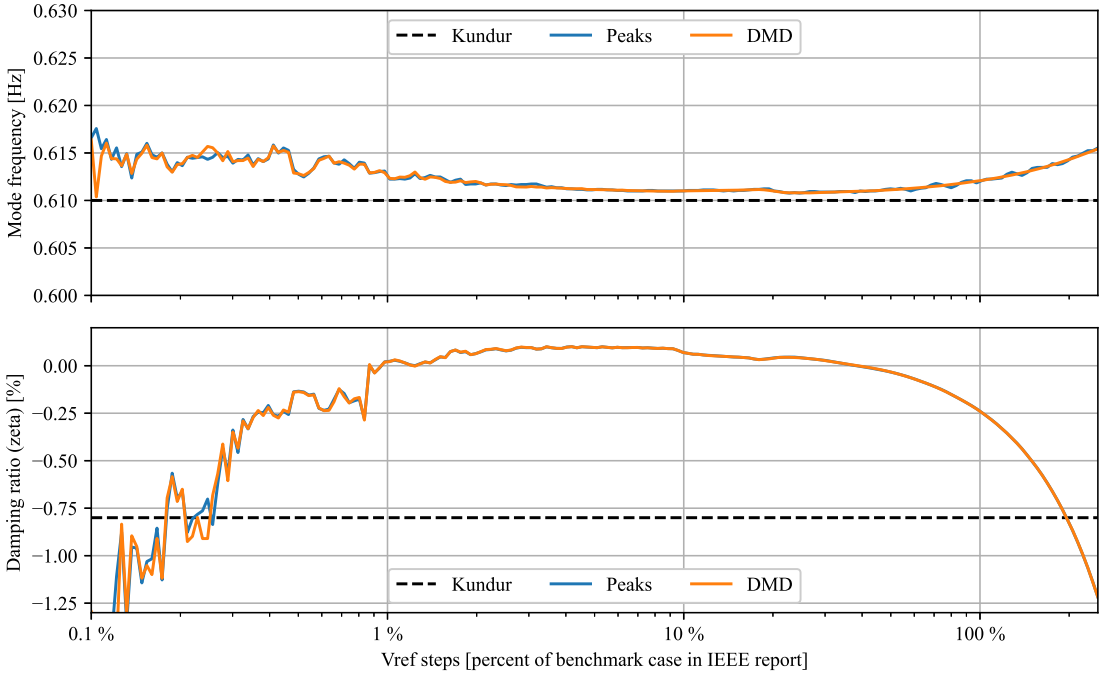


Figure 3.5: Frequency and damping estimates from DMD and the peak-finding method over a range of disturbance magnitudes, compared with the linearisation from Kundur [8].

A striking feature is the large variation and symmetry in the damping ratio ζ. The damping is actually positive when the disturbance magnitude is between 0.9 and 50 percent, meaning that the oscillation is decaying, not growing. Decreasing or increasing the disturbance magnitude beyond this range gives an increasingly undamped system, to the point where the negative damping ratio equals and then proceeds to continue below Kundur's linear modal analysis result. There is no sign of convergence to the the result from the linearisation, rather the curve seems headed toward negative infinity in both directions. If the disturbance magnitude is increased much above 250 percent of ΔV_{ref}, oscillations grow so rapidly that the system quickly becomes unstable, resulting in convergence problems in PSS/E. The validity of the system response for the range below about 3 percent of ΔV_{ref} is uncertain. The reason for the jaggedness is likely that the dynamic response here is affected by rounding errors in the single-precision floating-point format used by PSS/E [39]. This is corroborated by the observation that the trend of the curve for a damping magnitude below 1 percent depends on the PSS/E simulation step setting: While the curve always has a similar shape, the shape is moved to the right if the simulation step is decreased and to the left if the simulation step is decreased. There is no such movement in the shape for high damping magnitudes. DMD was implemented in the double-precision floating-point format in Python, and so will give reasonable estimates as long as the assumption of a linear time-invariant system holds within the time-window analysed.

The DMD oscillation frequency estimate generally differs very little from the value from Kundur's analysis. Note that Kundur's value was given with a precision of only two decimals, so the difference is within rounding error for most of the disturbance magnitude range. The frequency estimates were further confirmed by FFT, where a window of 100 seconds of filtered data was used, yielding a frequency resolution of 0.01 Hz. FFT gave a stable result of 0.61 Hz over most of the ΔV_{ref} range for the signal from all 11 buses.

The results from the parameter sweep over disturbance magnitude confirms that while a linearisation can provide important information about the stability of a system, it is not possible to extrapolate from a linearisation how the system will behave after a nonlinear response. It should be taken into account that for such a small and unstable system such as the Two-Area model, changing the voltage reference of all the system's generators simultaneously can change the operating point quite significantly from what it was when the linearisation was made. The difference in damping predicted by the linearisation and estimated from the data is therefore neither surprising nor an error. Again it should be noted that as a data-driven method, DMD does not identify system parameters directly, it can only estimate system dynamics as they are expressed in the data, and will always portray the system behaviour in a given window of time. Results from two different time windows will not and should not be identical if the system's dynamics are not constant.

An indicator of accuracy is the agreement between DMD and the peak-finding method over the whole range of disturbance magnitude. This confirms that the DMD algorithm has been correctly implemented, at that it gives credible estimates of structures in the data.

3.2.4 Comparing participation factors

As discussed in Section 2.4, the DMD time-map of a system's outputs is not identical to a state matrix, which has implications for the relevance of participation factors as defined for a linearised system model in [8]. A set of tests was done to investigate how the participation factors of [8] and participation factors as defined in [15] depend on the choice

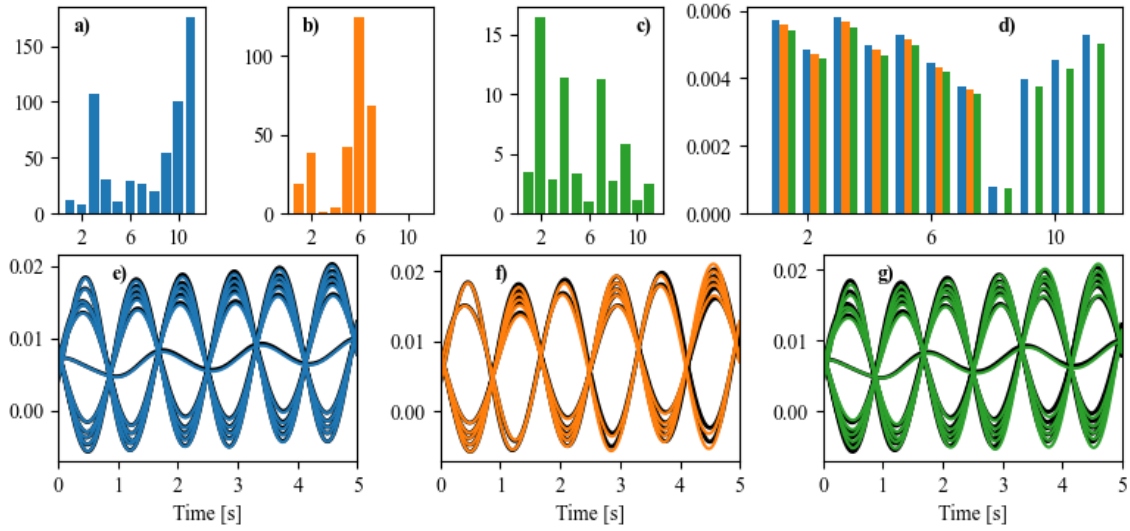


Figure 3.6: Participation factors based on eigendecomposition of a full time map with 11 channels (blue) and 7 channels (orange), and based on DMD with 11 channels and rank reduced to 5 (green). **a**)–**c**) show participation factors by the definition $p_{ij} = |\phi_{ij}\psi_{ji}|$; **d**) uses the definition $p_{ij} = |\phi_{ij}\lambda_j b_j|$; **e**)–**g**) show signal reconstructions with the original signal in black.

of signals used for analysis, and on the rank reduction step in the DMD algorithm. The tests were done on 5 seconds of post-disturbance frequency deviation data from a PSS/E dynamic simulation on the Two-Area models with a disturbance magnitude of 100 % of ΔV_{ref} as described in Section 3.2.2. The cases are as follows:

Case 1 Eigenanalysis was performed directly on a time-map $\mathbf{A}_{11}$ for a data set consisting of all 11 frequency channels. This time-map was computed directly by (2.13). The SVD and projection steps of DMD were omitted.

Case 2 Eigenanalysis was performed directly on a time-map $\mathbf{A}_7$ Consisting of only the first 7 of the frequency channels. All else is equal to case 1. The purpose of including this case was to observe how the selection of signals influenced the participation factor according to the two definitions mentioned above.

Case 3 Basic DMD with no shift-stacking or randomisation was performed on the full data set as in case 1. The rank was truncated to 5, so that the eigenanalysis was performed on a rank-5 map $\tilde{\mathbf{A}}$. Left and right eigenvectors of $\tilde{\mathbf{A}}$ were then projected back to full dimensions using (2.22) and (2.46). It was confirmed that $\mathbf{\Psi}_{DMD}\mathbf{\Phi}_{DMD} = \mathbf{I}$. Increasing the DMD rank so that it is equal to the full system rank of 11, $\mathbf{U}\tilde{\mathbf{A}}\mathbf{U}^{\mathrm{T}}$ is identical to the full-dimension time-map $\mathbf{A}_{11}$, further confirming that the DMD transformations were correctly implemented. The purpose of this test case was to observe the impact of the rank-truncation and projection step of DMD in the participation factor according to the two definitions mentioned above.

To visually observe how the eigensystem estimated in each case captures the dynamics in the data, reconstructions of the original signals were made using (2.58). Figure 3.6 e)–g) shows a good overlap between the reconstructions in colour and the original signal plotted in black in the background. The three signal reproductions are practically alike.

Differences between eigenvalues estimated in the three cases are very small. Right eigenvectors differ only by a complex-valued constant, as expected. However, the left

eigenvectors do *not* differ by a constant. Some right eigenvector elements may have real or imaginary components which are zero or close to zero. Hence a small change in $\boldsymbol{\phi}_i$ may necessitate a big change in $\boldsymbol{\psi}_i$ in order for $\boldsymbol{\phi}_i^T \boldsymbol{\psi}_i = 1$, and to maintain orthogonality so that $\boldsymbol{\phi}_i^T \boldsymbol{\psi}_i = 0 \, \forall \, i \neq j$.

The numerical instability of the left eigenvectors makes the participation factors computed by (2.45) highly sensitive to the choice of rank and the number of channels, as shown in Figures 3.6 a)–c), where it is hard to spot any common traits even though the three sets of participation factors are all computed from eigenanalyses which capture the same dynamics with a comparable level of accuracy. This corresponds to the argument made in Section 2.4 that participation factors as defined by [8] are not applicable to a low-rank approximation of system dynamics.

Participation factors as defined by [15], on the other hand, give stable results for different ranks and numbers of channels, as shown in Figure 3.6 d). This definition therefore seems more relevant in the case of DMD.

4 Algorithm performance and sensitivity

This section investigates the noise sensitivity of DMD with respect to parameters such as sampling rate, shift-stacking depth, window length and the number of channels. Several methods of rank reduction are tested and their merits discussed. Finally, the mode-tracking capabilities of DMD are tested and compared with SSI and MAR.

4.1 Noise sensitivity

The tests in this section were designed to evaluate the effect of noise on DMD performance. The DMD estimate of $\{\sigma, \omega, \boldsymbol{\phi}\}$ for the 0.61 Hz mode 5 seconds after a disturbance of 100 percent of ΔV_{ref} (Table 3.4, Figure 3.4) was used as a baseline. The results describe how much DMD estimates deviate from this baseline when the signal is contaminated with additive white noise. Parameter sweeps of sampling rate, shift-stacking depth, the number of channels and the length of the time-window of analysis are performed to study how they affect the noise-sensitivity of the DMD algorithm. The noise level is quantified using the ratio between the power of the 0.61 Hz mode and the power of the noise component (2.60).

For all tests, 10000 iterations were done for each change of each parameter at each noise level. Variation was provided by refreshing the signal with new noise for each iteration. Frequency and damping results in this section are given as the mean square error (2.59), while the total vector error (2.63) is used for the mode shapes. As shown in Section 2.4, mode shapes are not unique in magnitude and angle but only in how their magnitudes and angles relate to each other. To allow comparison between test iterations, eigenvectors were therefore normalized so that the element for bus 1 was set to $1\angle 0°$.

Since mode shapes are arrays with one entry per channel, the mean error of all channels was used. Aggregated results are presented in Figures 4.1–4.4, where shaded areas show the 10–90 percent confidence interval and solid lines show the median error. Note that the quantities studied are in different ranges of magnitude, and that results for different quantities cannot therefore be directly compared.

Table 4.1 lists all the DMD parameters for the noise-sensitivity tests. All the parameters except the one swept in a specific test were kept constant. The rank was fixed to $k = 10$ for all these tests to avoid any influence from rank estimation techniques on the results.

4.1.1 Sampling rate

Modal analysis is often performed on downsampled data [12, 15, 16], either because of storage limitations, computational burden or because it can provide some filtering of high-frequent noise. However, downsampling the data to half the frequency means that

Table 4.1: Parameters for the noise-sensitivity tests. All parameters are kept constants at the value given here, except for the parameter swept in each test.

Parameter	**Value**
Sampling rate	60 Hz
Window length	5 s
Shift-stacking depth	20%
Number of signals	11
Rank	10

the algorithm has only half the data to work with, contrary to strategies such as shift-stacking which strive to increase the size of the data matrix. In addition to the original 120 Hz PSS/E output, the sampling frequencies included in Figure 4.1 are some of the standard PMU reporting rates for 60 Hz systems (equivalents for 50 Hz systems are 50, 25, 10 Hz) [10].

Frequency and damping estimates have good accuracy even with low sampling rates. The MSE of 10^{-4} at the top of the confidence band for the 10 Hz case corresponds to an error of ± 0.01 Hz. The unit of the damping estimates is percent, so an MSE of 10^{-2} corresponds to an error of ± 0.1 percent. One thing to note is the very wide confidence bands, which imply a lot of variation between results. Mode shape TVE is up to 0.3 per unit for a 10 Hz sampling rate when SNR=0, which is not very impressive.

The noise-sensitivity of all quantities decreases when the sampling rate increases. This comes at the disadvantage of higher computational cost, but seems necessary given the poor mode shape estimates. The maximum standard PMU reporting rate of 60 Hz was therefore used in the remaining tests in this section.

4.1.2 Shift-stacking depth

In [18], the optimal shift-stacking depth is mentioned to be 30–40 percent. In the tests performed in the present project, no significant benefit was found from increasing the shift-stacking depth beyond 20 percent, as shown in Figure 4.2. Errors in mode shape estimates are actually better with minimal shift-stacking and starts increasing when the shift-stacking depth exceeds 10 percent. Further investigations indicate that the optimal shift-stacking likely depends on signal length, and it may also depend on the number of channels used. Interestingly, there exists an optimum with regard to shift-stacking depth when SNR=20 dB, but not for higher amounts of noise. The reason for this is unknown, but with lower levels of noise there seems to be a trade-off between the accuracy of mode-shape estimates and the accuracy of frequency and damping estimates for shift-stacking in the region 0–20 percent. Note that these results are from analysis of a static window of data where the dynamics are reasonably constant. Possible limitations with shift-stacking when used with moving time-windows and in the presence of transients are discussed in Section 6.

4.1.3 Number of channels

Intuitively, increasing the number of channels increases the amount of information present in the analysed data, while shift-stacking only augments the data by repeating information which is already there. Indeed, adding more signals gives better frequency and damping estimates as shown in Figure 4.3, with no sign of an optimum haven been reached when using the maximum number of available channels, which was 11.

The sharp increase in mode shape TVE for all noise levels may be related to the test setup, where channels were always added in numerical order to allow comparisons with the baseline. Channel 8 is from the the midpoint bus where the oscillations are much smaller than at the other buses, so that the SNR for this mode at this bus is much lower than at the other buses. This shows how the accuracy of estimates not only depends on the number of channels present, but also on the quality of information provided by each channel. This points to the importance of optimising the locations of PMUs used for modal analysis, which however is outside the scope of this project.

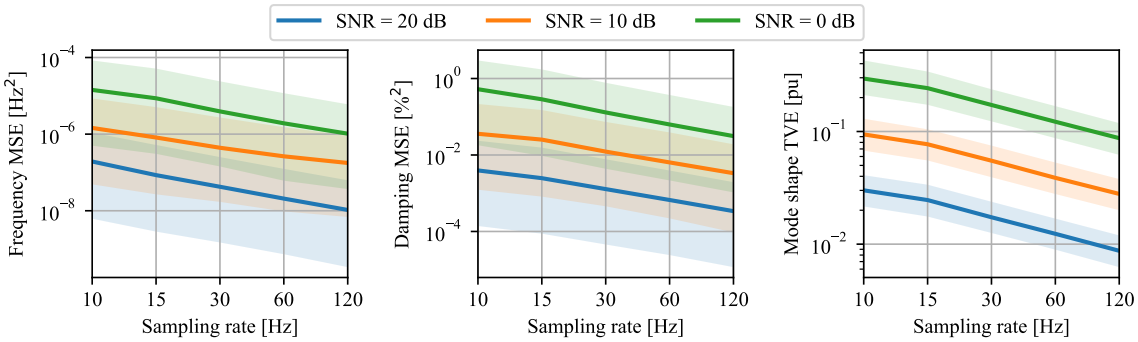


Figure 4.1: Noise sensitivity with different sampling rates. The shaded areas are 10-90% confidence bands.

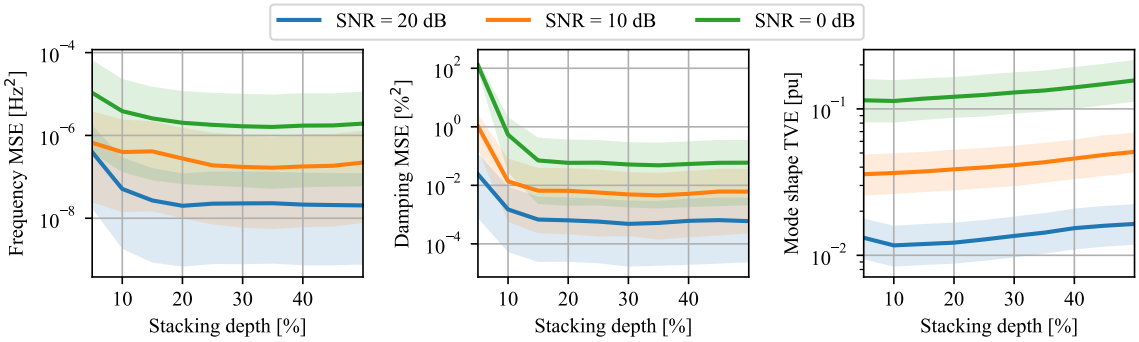


Figure 4.2: Noise sensitivity with different shift-stacking depths.

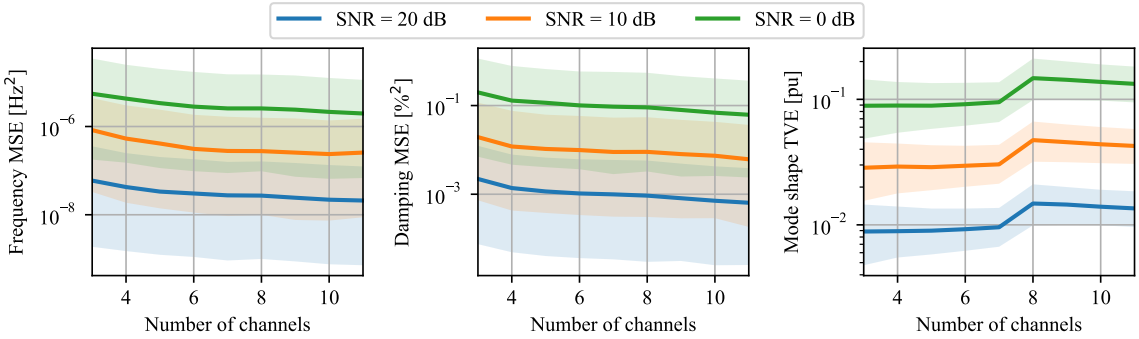


Figure 4.3: Noise sensitivity when varying the number of channels.

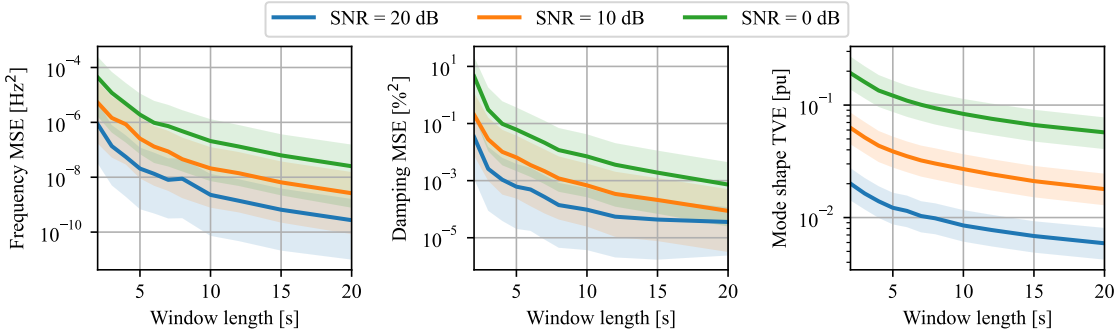


Figure 4.4: Noise sensitivity with different window lengths.

4.1.4 Window length

As shown in Figure 4.4, Increasing the window length gives better estimates for all quantities, which is not surprising. The most dramatic decrease in error is from 2 to 5 seconds, but results for all quantities except the frequency indicate that there is reason to consider using an even longer window than this. However, it is important to note that these results are for a time-window where the dynamic structure is fairly constant. The benefits noted here may not apply if the amplitude of a mode changes during the time-window, which will be discussed further in Section 6. Increasing the window length will also increase the delay between a transient and the time at which it is caught by the analysis.

4.2 Rank selection

This section investigates different methods of rank selection. The data set used in this section is 62 seconds of output from the Two-Area system with with a sampling rate if 60 Hz. Frequency channels from all 11 buses were used for analysis, with measurement noise added at a level of SNR=0. The SNR of the measurement noise again refers to the ratio of the noise power to the power of the 0.61 Hz mode, measured in a reference window of $t \in [15, 20]$ seconds as in previously described tests. The noisy signal from Bus 1 is included in Figure 4.5. Voltage reference steps ΔV_{ref} are applied at 10 seconds. The length of the time-window of analysis was set to 5 seconds, and analyses were made with intervals of 0.5 seconds. All parameters are listed in Table 4.2.

Rank selection methods seek to find the optimal number of singular values to keep in order to capture actual dynamic content and remove noise. Figure 4.5 (left) shows how the ten first singular values evolve over time in the simulation data described above. As mentioned in Section 2.2, the singular value decomposition is a mathematical tool which factorises a rectangular matrix into three separate matrices according to (2.18), where the middle matrix $\mathbf{S}$ is a diagonal matrix with the singular values on the diagonal, and the left and right matrices consist of the left and right singular vectors, respectively. The singular value decomposition finds correlated patterns in the data [14]. The singular values are by convention sorted in decreasing order, so by truncating from the bottom as described in Section 2.2, it can work as a type of noise filtering [23]. Before the disturbance, the singular values have a flat profile since the signal from 0 to 10 seconds only contains noise and there is no dynamic content. From 10 seconds and onward the time-window starts including the disturbance, and there is a bump where up to 7 singular values rise visibly above the noise. This is where the well-damped local modes have been excited and create correlations in the data until they decay to the level of the noise. The sum of singular values, which represents the total energy in the signal, (Figure 4.5 centre right) rises sharply until $t = 15$ seconds when only the fault and its immediate aftermath is

Table 4.2: Parameters for tests of rank selection methods.

Parameter	**Value**
Measurement noise SNR	0 dB
Sampling rate	60 Hz
Window length	5 s
Update interval	0.5 s
Shift-stacking depth	20%
Number of signals	11

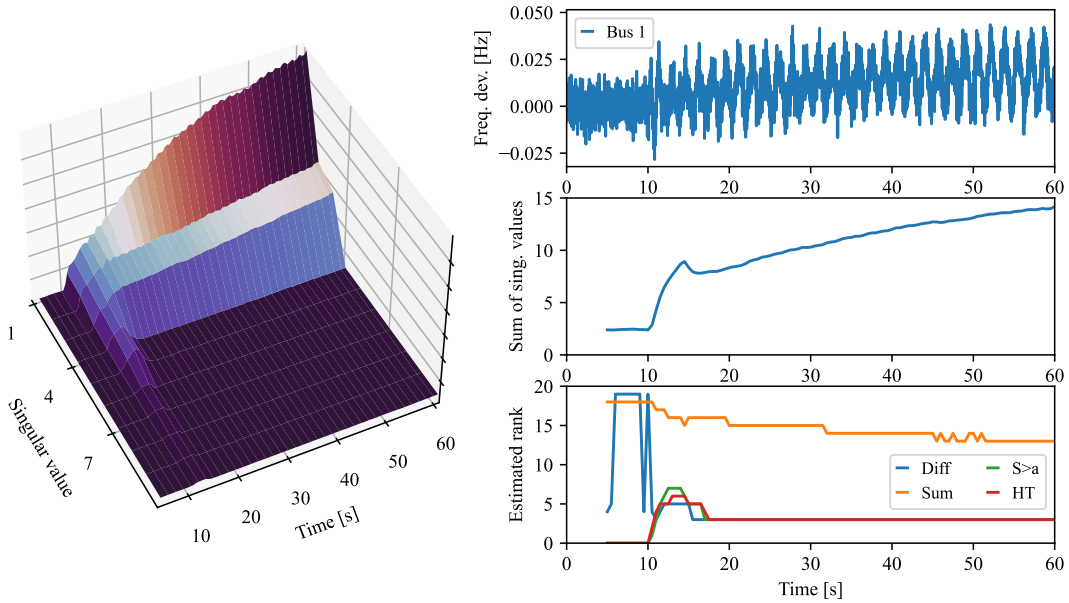


Figure 4.5: Singular values (left), signal from one channel (top right), sum of singular values (centre right), rank estimated by four methods (bottom right).

included in the time-window. There is a slight decay as the initial nonlinear response and the local modes die off, until the increasing offset from zero and the resonating 0.61 Hz mode create a rising trend in the total energy. At this point the bump in Figure 4.5 (left) has passed and the three first singular values dominate for the rest of the time included in the data set.

The four methods of rank selection described in Section 2.2.1 were implemented and tested, as shown in Figure 4.5 (bottom right), in all cases with the maximum rank set to 20.

For the difference method as described in [13], the difference between the the two first singular values was excluded from the comparison. Taking the difference between the first two singular value into consideration would give a rank estimate of $k = 1$ for most of the duration of the data set since the offset from zero is the dominating dynamic structure. The results of the difference method are very inconsistent when the offset is zero and the signal contains no oscillatory components, as in the first 10 seconds of Figure 4.5 (bottom right). The rank is estimated to 5 in the immediate aftermath of the disturbance when the local modes are active, and settles to 3 for the rest of the data set's duration because of the large drop-off from singular values 3 to 4.

For the cumulative sum method, the cutoff was was set to 90 percent, which is much lower than the cutoff suggested in [17], but this method still estimates the rank to be very high even for such a small system. The rank estimate decreases with time because the signal's offset and oscillations take an increasing share of the total energy in the signal.

For the empirical threshold method, the rank was set to the total number of singular values exceeding $a = 0.4$. This threshold was chosen so that it was somewhat higher than the average of the singular value spectrum in the pre-fault region which contains only noise. In the pre-fault region it estimates the rank to 0, i.e. that there is no dynamic content and only noise, which is correct. From 17 seconds and onward it agrees with the difference method. In the disturbance aftermath the empirical threshold method gives

slightly higher rank estimates than the difference method, but this is highly dependent on the chosen threshold, and at $a = 0.6$ the two methods give almost identical results from $t > 10$ seconds.

The hard-thresholding method of [25] gives very similar results as the empirical threshold method. However, this method relies on statistical properties of the data instead of a preset value, and is therefore more adaptive than the empirical threshold method. The hard-thresholding method will therefore be used in the following.

4.3 Mode tracking over time

An important property of an algorithm is its ability to track modes and their damping over time. This applies both for analysis of historical data and for online monitoring purposes. The algorithm should not just be able to analyse strong oscillations after a disturbance, but also catch oscillations when they are still small to give early warning if they should become dangerously undamped. In this section, the ability of DMD to monitor a system both under ambient condition and after a disturbance is investigated. The data used is from a PSS/E dynamic simulation of the Two-Area system with load perturbations as described Section 2.6.3.

File-reading load models were assigned to both loads in the Two-Area system, which are located at buses 7 and 9 as seen in Figure 2.5. For each load a random walk with $\sigma_\varepsilon = 1$ was constructed by (2.64) and written to file. The standard deviation of the load change per simulation step was set to $\sigma_\varepsilon = 0.0005$. With a simulation step of 1/300 s, this is equivalent to a standard deviation per hour of $0.0005\sqrt{3600 \cdot 300} \approx 0.5$ per unit or 50 MVA, or approximately 6.5 MVA per minute, as per (2.65). Since the load perturbations introduce a power mismatch in the system, all the four generators were equipped with speed governors to give the model frequency containment capabilities. Full details are included in Appendix B.

The random walk was initiated after 1 second in order to give the simulation a «flat» start to confirm full convergence before starting perturbations. Then the simulation was left to run with only the load perturbations to simulate ambient system behaviour. At 210 seconds, a 100 percent ΔV_{ref} disturbance as defined in (3.1) was applied so the algorithm's

Table 4.3: Parameters and run times for DMD, SSI and MAR analysis of simulation data.

DMD		**SSI**		**MAR**	
Target freq.	0.61 Hz	*Target freq.*	0.61 Hz	*Freq. low*	0.51 Hz
Tolerance	±0.1 Hz	*Tolerance*	±0.1 Hz	*Freq. high*	0.71 Hz
Max. rank	12	*System order*	12	*AR order*	30
Sampling rate	60 Hz	*Sampling rate*	60 Hz	*Sampling rate*	60 Hz
Data window	10 s	*Data window*	10 s	*Data window*	10 s
Update step	1 s	*Update step*	1 s	*Update step*	1 s
Stacking depth	10 %	*Covariance matrix window*	2 s		
		Cov. matrix update step	0.0167 s		
Run time (Python 3.10)	6.5 s	Run time (Matlab 2021b)	70 s	Run time (Matlab 2021b)	680 s

performance during a transient could be observed. The simulation was then left to run until 310 seconds, with the random walk load perturbations continuing after the disturbance.

It takes some time before the load perturbations get the system dynamics going. The first 10 seconds of the simulation were therefore discarded, leaving 300 seconds of data with a disturbance at $t = 200$ seconds. The frequency deviation signals were then converted from per unit to Hz, and the data was downsampled to a sampling rate of 60 Hz by simple decimation with no filtering.

Considering again the Welch periodogram of Chalmers PMU data in Figure 2.6, it seems fair to assume a high signal-to-noise ratio for ambient power-system oscillations. To add noise to the data, the power of the ambient inter-area oscillations in the simulated data was measured by applying a second order Butterworth band-pass filter with a passband of $[0.3, 1.2]$ Hz was therefore applied before computing P_{signal} by (2.61) on the signal segment $t \in [25, 100]$ seconds. White Gaussian noise with a standard deviation σ_{noise} computed by (2.62) for $SNR = 0$ was then added to the unfiltered data. The result was a data set with noise at the same power level as the signal of interest. Frequency deviation channels for the four generator buses are shown in Figure 4.6a.

DMD was then used to track the inter-area mode. To evaluate its performance, results from SSI and MAR were used for comparison.

For DMD, the hard-thresholding method described in Section 2.2.1 was used for rank truncation. The SSI and MAR algorithms were set up to have fixed rank or system order. Parameters for the algorithms were chosen with the aim of comparing the algorithms on equal terms, while also getting as good performance as possible from each. Parameterisation for DMD was partly informed by the sensitivity tests in Section 4.1. Further tuning of DMD and tuning of SSI and MAR was done by trial and error. All parameters are listed in Table 4.3. A data-window length of 10 seconds (600 samples) was chosen since shorter windows did not give stable results with any of the algorithms. For SSI, the most critical parameter seemed to be the covariance matrix update step, with results rapidly deteriorating if this step was increased from the minimum of 1 element. The order of the MAR algorithm had to be set as high as 30 to get fairly stable results. DMD seemed to perform best when the shift-stacking depth was set as low as 10 percent.

There is very good agreement between the mode frequency estimates of the three algorithms, as shown in Figure 4.6b. Note that the estimates start at $t = 10$ seconds, because the timestamp of an estimate is set as the final timestamp in the data window the estimate is based on. This preserves causality, as measurements from the future are not considered in an estimate made at the present time. An estimate taken at e.g. $t = 27$ seconds therefore considers data in the time span $t \in [17, 27]$ seconds. The mode frequency varies somewhat during the first 200 seconds, which is to be expected since the system is constantly perturbed. During the $t \in [200, 210]$ seconds there is some confusion as the data window glides over the transient. For $t > 210$ seconds the mode frequency estimates become very stable, settling around 0.63 Hz. One reason for the increased stability of estimates is that oscillations are more powerful than the perturbations here so, that it is likely that the frequency actually is more stable. Also, the signal-to-noise ratio is much smaller here than before the disturbance, so the estimates are likely also more accurate.

The three algorithms also agree on the damping most of the time, as shown in Figure 4.6c, though the great variation in all the damping estimates raises questions about the accuracy. Nevertheless, the variations seem to reflect what is in the data: Comparing with Figure 4.6d, it can be seen that the oscillation amplitude grows when the damping is negative, and vice versa, for example in the estimates for $t \in [80, 110]$ seconds. The estimates diverge in the segment $t \in [150, 200]$ seconds, where the oscillation amplitude is

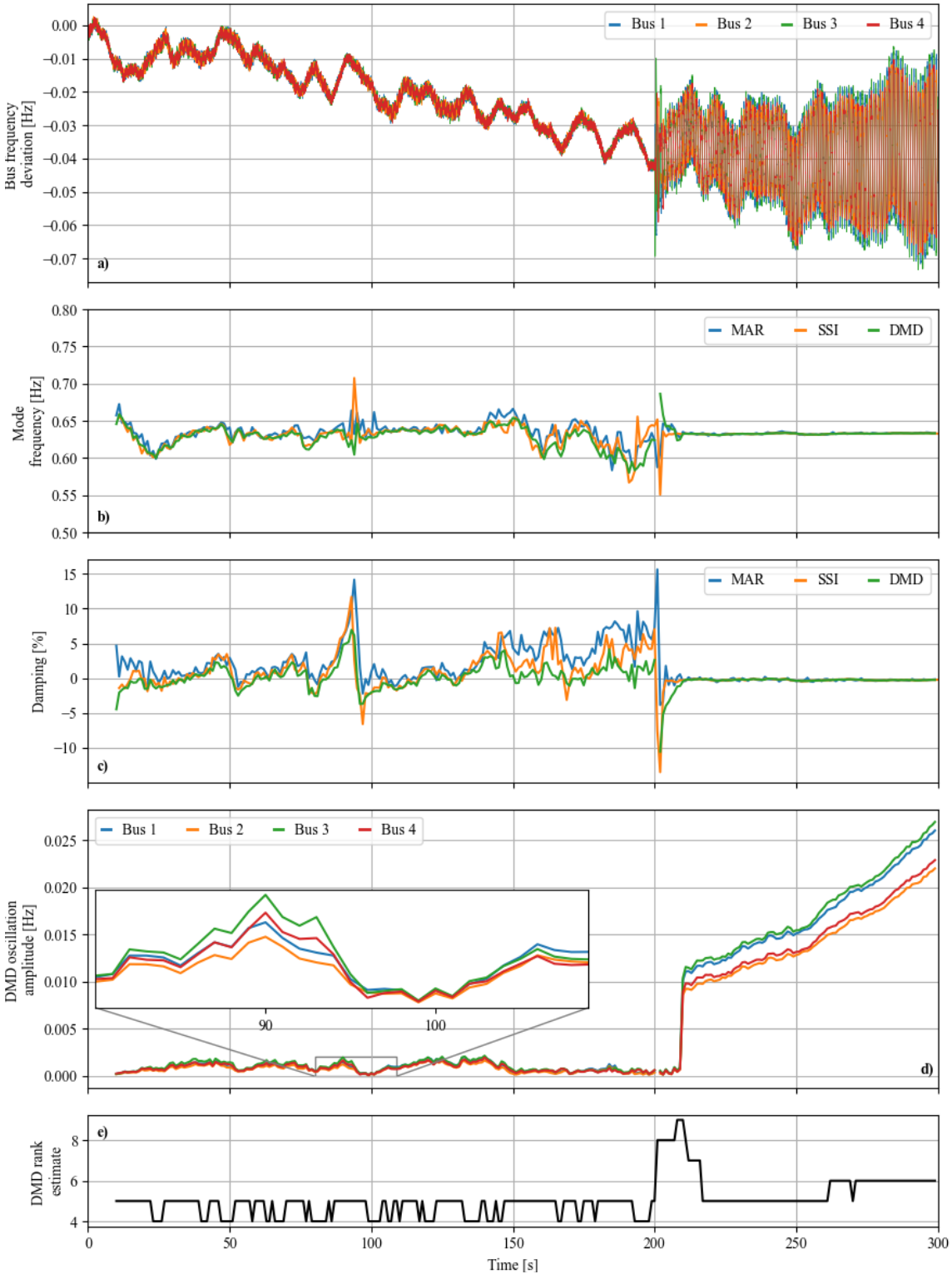


Figure 4.6: Tracking the inter-area mode of the Two-Area system. Load perturbations are implemented as a random walk where the steps are normally distributed with a mean of 0 and a standard deviation of 0.0005 per unit. Simultaneous voltage reference steps on all four generators are applied at $t = 200$ seconds. a) Bus frequency deviation signal at generator buses. b) Mode frequency estimates from MAR, SSI, DMD. c) Damping ratio estimates from MAR, SSI, DMD. d) DMD oscillation amplitude estimates for generator buses. e) DMD rank estimates.

consistently very low and mode frequency measurements also have more variation, indicating that the oscillations here are close to the limit of what these algorithms can handle. About 20 seconds after the disturbance, there is again very good agreement between the algorithms. The estimated damping here is on average approximately -0.25 percent, corresponding to what was found for the unperturbed system in Section 3.2.2.

Figure 4.6d also shows that the relationship between the oscillation amplitudes at the different generator buses is fairly stable, indicating that the same mode is tracked throughout the time series, since the oscillation amplitudes and the DMD mode amplitudes only differ by a constant. Tracking mode shapes will be further explored in Section 5. It can also be visually verified that the estimated oscillation amplitudes are consistent with the oscillations in Figure 4.6a, further confirming that the DMD results are reasonable.

DMD rank estimates in Figure 4.6 e) are fairly consistent through the ambient part of the data, with a peak during the transient when the local modes are excited for a short while, and another stable region as the offset and the inter-area oscillations dominate the signal.

Run times for the algorithms varied by two orders of magnitude, as shown in Table 4.3. It should be noted that these times are approximate best performances from a few runs and not proper benchmarks. The speed of MAR is very sensitive to the selected system order, and the speed of SSI depends much on the covariance matrix step rate. For both these algorithms, stable estimation performance was prioritised over computational speed here. Comparisons are not completely fair because tests were performed in two different programming environments (Matlab and Python), and the received code may not have been optimised for speed. The DMD code was not optimised for speed either, beyond standard programming practices such as using vectorised operations (Numpy arrays) instead of for loops where possible without sacrificing the readability of the code. Much of the speed advantage of DMD is due to the randomised sampling routine described in Section 2.2.3, which improved the total computation time from 92 seconds to 6.5 seconds in this case.

5 Testing on ambient real-world data

This section explores the possibility of using DMD for power-system oscillation monitoring under ambient conditions. Tests are performed on real-world PMU data from the Nordic power system.

5.1 PMU data from five universities

The PMU data used in this section comes from a network of PMUs at four universities in Sweden and one in Finland: Lund University, Chalmers University of Technology in Gothenburg, Luleå University of Technology, KTH Royal Institute of Technology in Stockholm, and Tampere University. Figure 5.1 shows the geographical placement of the PMUs.

The frequency deviation signals from all five PMUs were used. The data was available as PMU byte strings in the 16-bit integer data type [10] with a 50 Hz sampling rate. While the 16-bit integer data type gives shorter byte strings for streaming or storage, it also requires that frequency deviation data is rounded to the nearest 1 mHz. This rounding results in significant distortion in the form of both loss of information and numerical noise, since much of the ambient inter-area oscillation activity in the frequency deviation signal takes place at this order of magnitude or below. Another source of distortion is that these PMUs are situated in the distribution grid, plugged into wall sockets. The data set had some stretches of missing data and also contained some corrupted or duplicate timestamps. Data points with corrupted timestamps were removed. Duplicates were also deleted. Since a thorough investigation of the nature and distribution of missing data was outside the scope of this project, and since the data was too noisy to consider more advanced methods of handling missing data, linear interpolation was used to fill in the missing data points. This gives some added distortion, but filling in missing data points is necessary to preserve the temporal integrity of the time series. In a spectrogram, sections

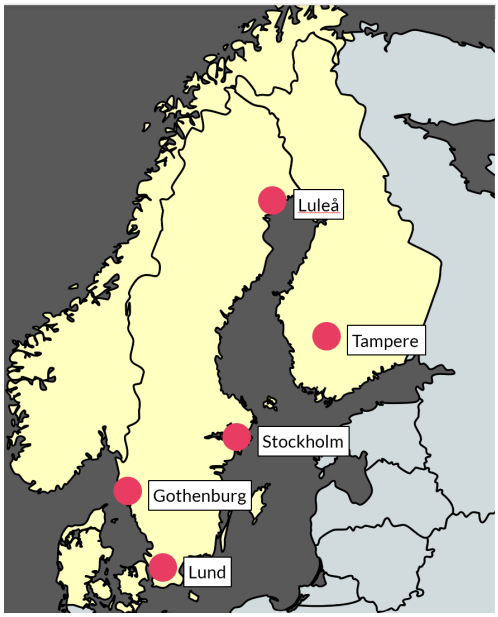


Figure 5.1: Locations of university PMUs. Map ©www.mapchart.net

of missing data longer than the time-window used will appear as an empty band, since the linear interpolation introduces no periodic content.

Keeping the full time-resolution did not seem to give any benefit for mode-tracking due to the information loss mentioned above, so to reduce the distortion somewhat, the data was downsampled to a 10 Hz sampling rate after applying a 5 Hz low-pass filter to prevent aliasing [9].

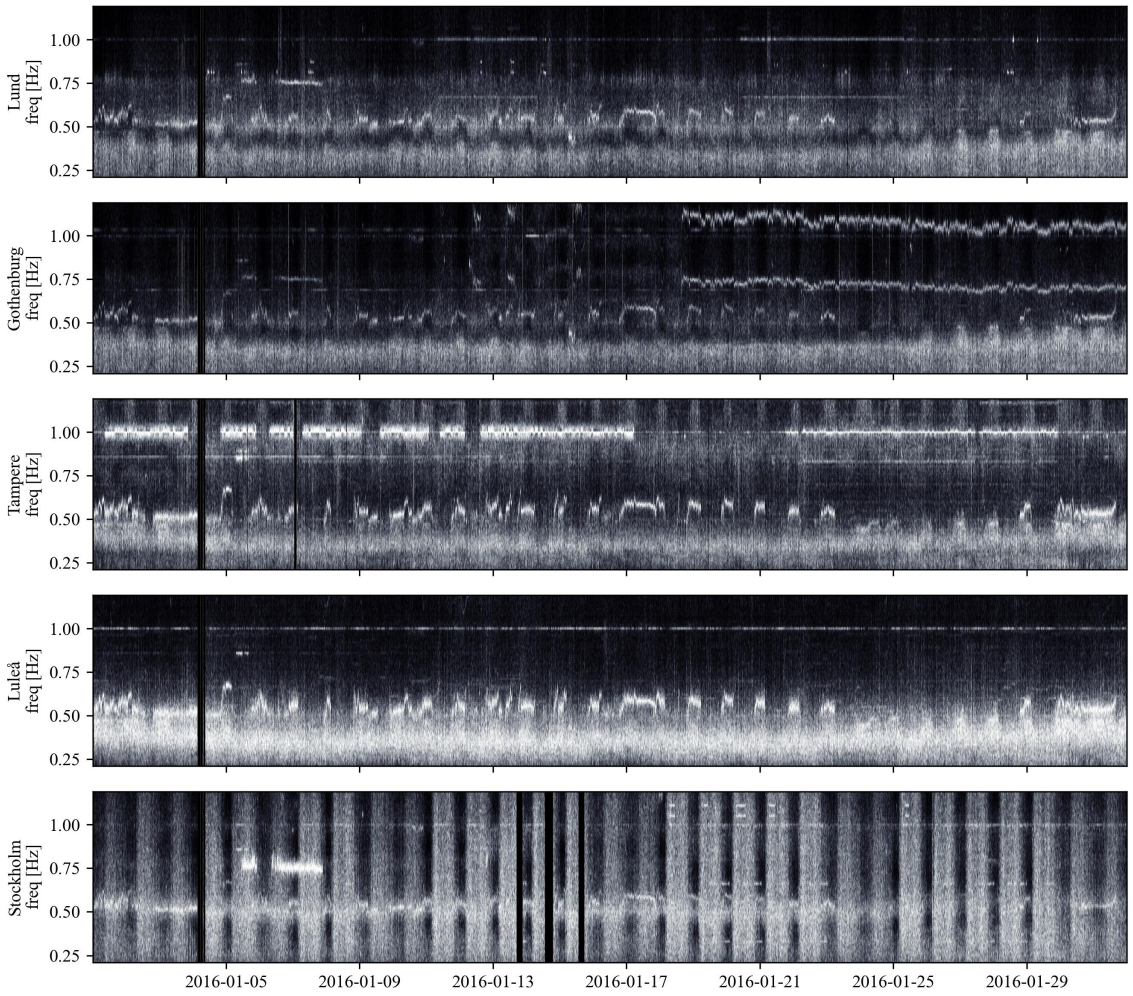
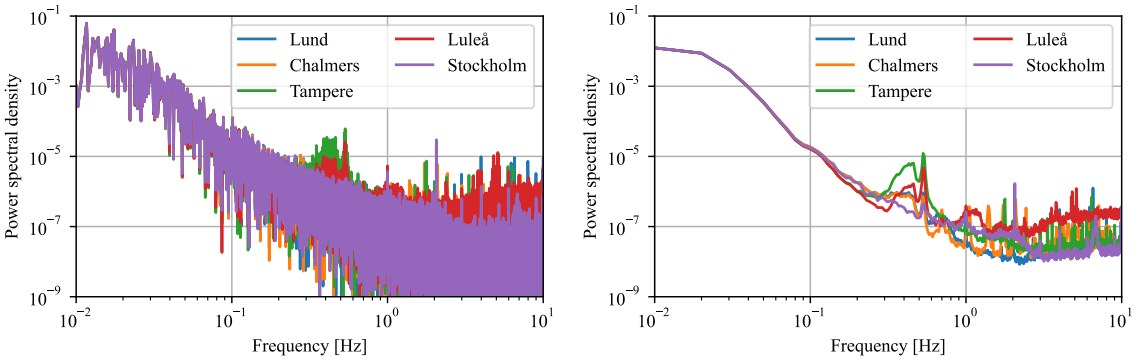


Figure 5.2: Spectrograms from five university PMUs in Sweden and Finland, January 2016.

Spectrograms of the frequency deviation channels from all five university PMUs during the whole month of January 2016 are shown in Figure 5.2. A high-pass filter was used to remove frequency content below 0.2 Hz for the spectrograms, which were computed with a 100 second time-window. A wide band, at first centred around 0.4 Hz and then dipping to 0.35 Hz before rising to 0.4 Hz again toward the end of the month, is prominent in the plots from Lund, Chalmers, Tampere and Luleå but is almost invisible at KTH. This is in the range of the lowest of the Nordic inter-area modes.

There is clear diurnal variation in the data from KTH, where the noise over the whole included frequency spectrum is strong during daytime and fades at night, and this pattern is stronger on weekdays than in the weekends. Daily patterns are also visible to a lesser extent at the other four locations. There is a rather sharp structure with a frequency



(a) Regular periodogram. **(b)** Welch periodogram (window length 100 s).

Figure 5.3: Power spectral density of the five university PMUs in the time interval 00:00–01:00 on 31 January 2016.

mostly around 0.5-0.6 Hz which is present during a large part of the month, mostly during night. Since it is visible in all five plots, it is possible that it is an inter-area mode. A faint, but sharp line at 1 Hz is also visible at 1.0 Hz in all the spectrograms. There are several other patterns of a more local nature. Transient disturbances are not visible at this time scale.

5.2 Tracking ambient power-system oscillations

DMD, SSI and MAR were used to track the possible inter-area mode in the hour between 00:00 and 01:00 on the 31 January 2016. The periodogram in Figure 5.3a gives an impression of how much noise is present in the signals, while the Welch periodogram Figure 5.3 draws a clearer picture of what periodic content is present on average during the hour.

No further preprocessing or filtering of the PMU data was done beyond the down-sampling step described above. A 120 second time-window was used since SSI and MAR did not give stable results for shorter time-windows. A full list of parameters is given in Table 5.1. During the hour which was analysed, a pattern was clearly present at all five locations according to Figure 5.2 with a frequency of approximately 0.53 Hz. The time-series plot of the signal from Chalmers in Figure 5.4a shows that there is a frequency dip during the first couple of minutes which likely is due to a minor disturbance. For the rest of the hour, the power system seems to be operating under ambient conditions, since the signal mainly consists of the ultra-low common-mode oscillations. In the zoomed-in spectrogram from Chalmers in Figure 5.4b, the 0.53 oscillation is still visible, but it is not as clear anymore. It varies both in frequency and in strength, seemingly disappearing at times, and partly melts into the lower-frequency content.

Nevertheless, DMD, SSI and MAR all manage to track the mode throughout the hour, as shown in Figure 5.4 c). All the three algorithms agree fairly well with the mode frequency indicated by the spectrogram. The most consistent frequency estimates come from MAR. However, results from MAR were very dependent on the selected order, which casts some doubt about this result. MAR Frequency estimates tended toward a somewhat higher value than SSI and DMD frequency estimates if the model order was increased beyond 40, which was used for the results shown here.

In Figure 5.4d, Damping estimates from DMD and SSI both have an average of between 0 and 1 percent, although DMD gives much more stable results. MAR gives damping estimates of around 4 percent most of the time for this particular choice of model order,

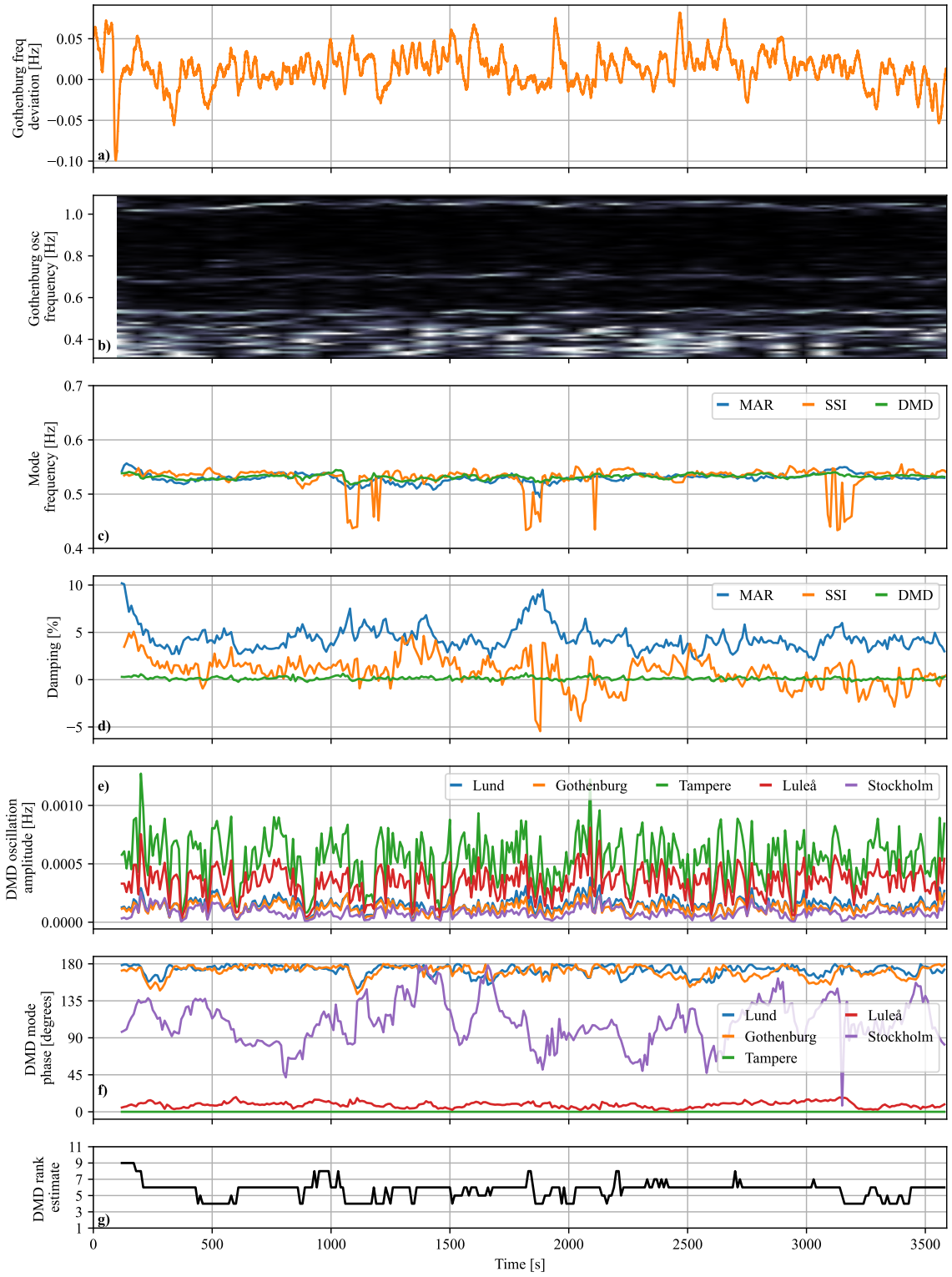


Figure 5.4: Tracking the a mode in PMU data. a) Bus frequency deviation signal at Chalmers. b) Spectrogram for the Chalmers frequency deviation signal. c) MAR, SSI and DMD mode frequency estimates d) MAR, SSI and DMD damping ratio estimates. e) DMD oscillation amplitude estimates. f) DMD mode angle estimates. g) DMD rank estimates.

Table 5.1: Parameters and run times for DMD, SSI and MAR analysis of PMU data.

DMD		**SSI**		**MAR**	
Target freq.	0.53 Hz	*Target freq.*	0.53 Hz	*Freq. low*	0.43 Hz
Tolerance	±0.1 Hz	*Tolerance*	±0.1 Hz	*Freq. high*	0.63 Hz
Max. rank	25	*System order*	12	*AR order*	40
Sampling rate	10 Hz	*Sampling rate*	10 Hz	*Sampling rate*	10 Hz
Data window	120 s	*Data window*	120 s	*Data window*	120 s
Update step	10 s	*Update step*	10 s	*Update step*	10 s
Stacking depth	10 %	*Covariance matrix window*	10 s		
		Cov. matrix update step	0.1 s		
Run time (Python 3.10)	6.5 s	Run time (Matlab 2021b)	29 s	Run time (Matlab 2021b)	41 s

but lower model orders can give an average damping of up to 10 percent, and increasing the model order beyond 40 gives average damping estimates tending toward those of DMD and SSI.

While the oscillation amplitudes in Figure 5.4 f) change a lot during the hour, the ratio between them stays fairly constant. Also, the DMD mode phase estimates in Figures 5.4 g) show a persisting pattern where the oscillations in Gothenburg and Lund are in counter-phase with the oscillations at Luleå and Tampere. This means that there is a consistent mode shape, which further indicates that it is an actual mode that is being tracked. The mode shape matches fairly well with the 0.5 Hz mode mentioned in Section 1.1, where southern Sweden swings against Norway and Finland, while it is perhaps a bit surprising that the mode phase at Luleå should be so close to the mode phase at Tampere 800 kilometers away. The mode phase at KTH fluctuates a lot, which may be due to lower accuracy of estimates due to more noise and a weaker signal there. Note also the increased mode oscillation amplitude after the disturbance in the beginning of the hour. Also note that oscillation amplitudes most of the time are in the range of 1 mHz or below, i.e. in the same order of magnitude as the rounding caused by the 16-bit integer data type. Mode shape estimates from SSI are similar to those of DMD. As mentioned, MAR does not give mode shape estimates.

DMD rank estimates in Figure 5.4 g) are higher than in the case of the simulated data in Section 4.3. This is likely because the real-world PMU data has more periodic content over the whole frequency spectrum and therefore a slower decaying singular value curve.

When applied to the real-world data, the randomised sampling routine not only increased computational speed, it also affected the DMD results. Randomised sampling gave somewhat more variation in the frequency estimate, but the damping, oscillation amplitude and mode phase estimates were significantly more stable. While the damping estimates did become more stable, there is still some uncertainty about the accuracy of the damping estimates, and how it is affected by the randomised sampling. This will be discussed further in Section 6.

6 Discussion and future work

The results in Sections 4.3 and 5.2 show that DMD gives results comparable to those of the established methods SSI and MAR when used for mode-tracking on both simulated data and on real-life, very noisy PMU data. The steps taken to validate the the DMD algorithm in Section 3 indicate that its ability to identify known structures in the data could be trusted. This provides a proof of concept and confirms correct implementation.

The usefulness of a mode detection algorithm must be judged by its ability to glean desired information from data corrupted by noise. Tests in Section 4.1 show that DMD can give good estimates even when the signal-to-noise ratio is very low, but that there are some limitations, not least when it comes to damping. The results in this project give no reason to doubt that that DMD can identify an undamped mode with steadily growing amplitude, such as the inter-area mode after the disturbance at $t = 200$ in Figure 4.6 c), where there is good agreement between DMD, SSI and MAR.

However, there is significant spread between the three algorithms' damping estimates for 0.53 Hz mode under ambient conditions in the PMU data. As mentioned in Section 5.2, the MAR damping estimate cannot be trusted here since it varies so much depending on the chosen system order. While SSI and DMD agree fairly well on the average damping of this mode, this is not evidence that they are correct. First, it is hard to know what the true noise level is since the frequency profiles are so different for the five PMUs in Figure 5.3, and therefore also hard to estimate the range of error from the noise sensitivity tests in Section 4.1. Another confounding factor is that the noise is not white and therefore might affect the accuracy in unpredictable ways. What is safe to conclude from the noise sensitivity tests is that a low SNR gives large errors. It follows that the uncertainty of the damping estimates in this case is large. Another thing is the actual value of the SSI and DMD damping estimates for the 0.53 Hz mode, which are below 1 percent on average. This means that the mode is very close to being undamped. As mentioned in [4], anything below 3 percent is considered low. Yet this mode persists throughout the month, seemingly without ever causing any catastrophe, and it decays nicely with a damping ratio of over 6 percent during several ringdowns of unknown cause occurring on January 3 (estimated by band-pass filtering and curve fitting). Of course, it is unreasonable to make any assumptions about the damping of this mode on January 31 by extrapolation from an earlier date, but at least the mode seems to have been well damped during the afternoon of January 3. Given the uncertainties described above, the damping results for the PMU data should be considered inconclusive.

The uncertain damping estimates are troubling since the damping is perhaps the most important quantity for early-warning purposes. A low SNR must be expected when the intention is to track ambient power-system oscillations before they become so big that they limit transmission capacity or become an imminent threat to power-system stability. Ambient power-system oscillations can be in the mHz range and smaller, as was shown in Figure 5.4 e). However, the SNR may not be as low as in the data used in this study if measurements from PMUs located at transmission-level substations is used. First, transmission-level PMU measurements will likely not be as contamination by distribution-level noise will likely not be as measurements from wall-socket PMUs. Also, most PMUs today likely stream out measurements using the 32-bit floating-point data type, which will reduce numerical noise and information loss. As indicated by the tests in Section 4.1, a decrease in SNR can give a substantial increase in the accuracy of estimates, especially for the damping. With the considerable noise and the rounding in this data set, and the

low number of channels available, there may just not be enough information present in the data to give reliable damping estimates.

6.1 An important limitation

Even if the damping estimates are unreliable, they hint to a possible limitation with DMD which has not been mentioned in the literature on DMD for power-system monitoring. This limitation has to do with how DMD estimates are affected if a mode's amplitude is perturbed during the time-window of analysis, either by a disturbance or by stochastic load variations. The oscillation amplitude is then expected to decay or grow corresponding to the mode's natural damping until another random perturbation occurs. It has been observed in this project that DMD can have problems estimating the natural damping in time-windows covering perturbations of the oscillation amplitude, which is especially an issue when using long time-windows, as was necessary in Section 5.

It is noted in [12] that MAR gives good damping estimates when used with long time-windows, and similar results are claimed for SSI in [13]. There is a trade-off between system identification and warning purposes: A short time-window will yield estimates of the local tendency in the mode amplitude, allowing events to be will be quickly observed, but will be less useful for system identification. This can be seen in the damping estimates in Figure 4.6 c), which correspond to the changes in oscillation amplitude. With a long time-window, any event will be noticed long after it happened, but a long time-window contains a lot of information and can therefore give more precise information about the natural modes of the system. The difference between the damping estimates of these algorithms may have to do with how they approach correlations between data points. As a statistical time-series model, MAR explicitly considers local temporal relationships, while SSI includes an average of local correlations. DMD, on the other hand, extracts spatiotemporal patterns in the whole time-window. This may be part of the explanation.

A hypothesis is that shift-stacking may be an important part of the problem, by making singular events appear periodic. Consider a time-window containing a disturbance, such as a 10-second window covering $t \in [195, 205]$ second in Figure 4.6 a). The initial frequency response to the $V_{\mathbf{ref}}$ step at $t = 200$ seconds will be visible in the measurements taken 5 seconds into the time window, i.e. in snapshot $\mathbf{x_{300}}$, or column 300 in the data matrix Each row in the original data matrix represents a measurements of a system state at a physical location in the power system. After shift-stacking, the disturbance will be visible in snapshot $\mathbf{x_{301}}$ for the new second row, in snapshot $\mathbf{x_{302}}$ for the new third row and so on. This may trick DMD into interpreting the disturbance as a periodic structure, peaking at different times in different «locations» because of a difference in phase. The word «locations» is in quotation marks here because the added rows are just time-shifted copies of the original rows and therefore do not represent actual physical locations. In the original data matrix, an overwhelming amount of correlations may express the mode's natural damping, so that a discontinuity or sharp change existing during only a few snapshots will «disappear» from the analysis due to averaging. Smeared out in time by shift-stacking, such events may artificially gain increased prominence, causing the damping in actual modes to be wrongfully attributed to spurious modes.

If shift-stacking has the effect hypothesised above, it would explain the tendency of DMD damping estimates to spend an amount of time corresponding to a whole time-window to settle down after a disturbance, as in Figure 4.6 c). It could also explain the difference between DMD and MAR damping estimates for the PMU data, if it is assumed that MAR better identified the natural damping of the mode. Experiments show

that this hypothesis could have some merit. However, the ability to estimate damping in time-windows containing mode amplitude perturbations also seems to be affected by randomised sampling, although it is not completely clear how. Results may be affected by the selected rank, the length of the time-window, and also buy the nature of the amplitude perturbations and at which points during the time-window they occur.

Shift-stacking is used to reinforce periodic structures in the data and limit the impact of noise. It is especially useful for this purpose when measurements are available from only a few locations, such as with both the simple power-system model and with the real-world PMU data used in this project. If shift-stacking should prove to be detrimental to the accuracy of damping estimates during mode amplitude perturbations, it must be avoided. A shift-stacking depth of 10 percent with a window-length of 120 seconds and a sampling rate of 10 Hz, as was used for DMD on the real-world data, means expanding the data matrix from 5 rows to 600 rows. Hence another 595 measurement locations would be needed for to expand the data matrix correspondingly. It would not be equivalent, of course, since the added measurements would be contributing new information, while shift-stacking only augments the data by repeating information which is already there. However, it means that if shift-stacking must be avoided, tracking modes in ambient data with DMD is likely only possible using a large number of channels, especially if a low signal-to-noise ratio is the reality. Testing on such a scale was beyond the scope of this project.

6.2 Further work

The limitation regarding the damping estimate as described above is problematic in the context of modal monitoring for an early warning system, since the damping of inter-area modes is a critical quantity with respect to power-system stability. The relatively low computational cost and otherwise good accuracy of DMD makes it attractive for online modal monitoring, so more research is needed to determine how serious this limitation is, and which solutions could be possible. In the case of transients, 4.6 e) shows the DMD rank estimate increasing a lot in the time-windows covering the transient. Further investigation into rank selection methods and how they react to changes in the singular value spectrum due to modal excitation is therefore needed.

The hypothesis that shift-stacking may cause disturbances and random perturbations to be misinterpreted as periodic content should be tested. This would require a much larger system model or a much bigger collection of PMU streams than what was used in this project, so that a small set of shift-stacked data set from a small selection of locations could be compared with a set of comparable size with measurements from separate locations. Comprehensive tests covering mode amplitude perturbations of both disturbance type and ambient type would need to be performed. Sensitivity to changes in DMD parameters should also be investigated.

Tests on data sets with a high number of measurement channels are also necessary to test the full capability of DMD for power-system purposes. DMD as it was developed in the fluid-dynamics field was attractive because of its ability to make a low-rank approximations of very high-dimensional data. Studies on DMD for power-system modal analysis so far have used data with relatively few channels, meaning that what is made is a low-rank approximation of only slightly higher-dimensional data, or as in the case of this project, on data which may not have a high enough dimension to properly capture the dynamics of the system. Whereas a method such as MAR becomes very slow when the dimensionality of the data is increased, the rank-reduction capabilities of DMD means that it has better chances

of handling higher-dimensional data sets with acceptable computational speed. This is especially the case if it is found that random sampling can be used without detriment to damping estimates.

The geographical distribution of measurement points for the data in this project and in works cited is limited by availability. As indicated by Figure 4.3, increasing the number of channels may increase or decrease the accuracy depending on the observability of a mode at the locations the added channels are sourced from. Optimal PMU placement for system observability is a field of research in itself, and the knowledge gained there should be used in further studies on DMD used on bigger data sets.

All tests of DMD, SSI and MAR in this project were done on data which had undergone no preprocessing except the downsampling of the PMU data which included a low-pass filter applied at 5 Hz. Filters can be problematic for online monitoring because they increase the delay from an event occurs until it can be observed, can introduce oscillations into the data after transients, and have nonlinear phase response. Studies on DMD for power-system monitoring generally do not use any filtering of data, because DMD can handle offsets in the data and performs its own noise-filtering by using SVD and rank-truncation. Section 4.2 showed that shift-stacking greatly reduces the noise-sensitivity of DMD damping estimates – but as discussed above, this probably has a serious limitation. If shift-stacking cannot be used in power-system monitoring for this reason, further research into filtering signals for DMD is likely needed, with the goal of achieving equivalent robustness against noise without the same sensitivity to perturbations in mode amplitudes.

An option which was briefly, but unsuccessfully explored during this project was to include more than one system state in the modal analysis. The goal is to increase the dimensionality of the data. Inter-area oscillations are visible not only in the in the frequency, but also in the voltage angle and magnitude, and of course in power flows, which can be computed when current phasors are available in addition to voltage phasors. Initial tests indicate that different offsets and trends existing in different states are a challenge, and that the scaling of signals should also be considered. Further investigations into normalisation and filtering of signals could allow further progress here.

Stochastic input to dynamic power-system simulations are another possible avenue of research, since output from carefully selected buses in a large system model could also be used to test DMDs capabilities. In this project stochastic input was limited to loads, but could be extended to intermittent renewable generation such as solar and wind power. A random walk such as that used in this project is practical since it is easily scalable, but it is unsuitable for longer simulations since it does not have finite variance, as is evident by (2.65). With access to fine-resolution load data (and data from wind and solar generation), more advanced time-series models such as ARIMA processes could be fit to the data to achieve statistically motivated and stationary stochastic inputs.

7 Conclusion

In this project, the DMD algorithm was implemented in Python and applied for oscillation tracking on real-life PMU measurements from the Nordic power system under ambient operation.

The implementation of DMD and the quality of DMD estimates were validated by comparing its results with absolutely known input parameters of synthetic data, and by comparing with structures demonstrably existing in power-system simulation data. The algorithm was found to be correctly implemented and to accurately identify mode frequency, damping and mode shapes. Statistical tests yielded new information about the sensitivity of DMD estimates to changes in sampling rate, data shift-stacking, the number of data channels, and the length of the analysis window under various levels of noise.

A user-defined load model for PSS/E was written in Fortran to introduce stochastic load variations in dynamic simulations in the form of a random walk. DMD was applied to track an inter-area mode in PSS/E output from a dynamic simulation with the IEEE Two-Area benchmark system and the random walk load model. It gave frequency and damping estimates similar to the results from stochastic subspace identification (SSI) and the multivariate autoregressive method (MAR), both for ambient oscillations and an undamped oscillation after a disturbance.

Finally, DMD was shown to give results comparable to those from SSI and MAR when used for mode-tracking on noisy data from a set of five university PMUs located at the distribution level in the Nordic power system.

Novel work on the DMD algorithm includes combining a hard-thresholding rank selection method with a randomised sampling technique, and the application of both of these to mode-tracking over time. These techniques improve both the stability of most estimates and reduce the computational cost of the algorithm. However, a limitation was discovered regarding DMD's ability to estimate damping correctly when the mode amplitude is perturbed during the time-window analysed. The data augmentation method of shift-stacking may be one of the sources of this issue, causing singular events to be misinterpreted as periodic content. Avoiding shift-stacking means that including measurements from PMUs at a large number of locations may be necessary to get good modal estimates with DMD.

References

[1] IPCC. "IPCC, 2023: Summary for Policymakers". In: *Climate Change 2023: Synthesis Report. A Report of the Intergovernmental Panel on Climate Change. Contribution of Working Groups I, II and III to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*. Ed. by Hoesung Lee and José Romero. Geneva: IPCC, 2023.

[2] Svenska Kraftnät. *Systemutvecklingsplan 2022-2031. Vägen mot en dubblerad elanvändning*. Tech. rep. 2021.

[3] Jim Follum. *Power System Oscillatory Behaviors: Sources, Characteristics, & Analyses*. Tech. rep. 2017.

[4] Janne Seppänen et al. "Resonance of Forcing Oscillations and Inter-Area Modes in the Nordic Power System". In: *2018 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*. 2018, pp. 1–6. DOI: 10.1109/ISGTEurope.2018.8571482.

[5] Jukka Turunen et al. *Using Spectral Analysis and Modal Estimation for Identifying Electromechanical Oscillations A Case Study of the Power System in Northern Norway and Northern Finland*. Tech. rep.

[6] Janne Seppanen et al. "Operational experiences of WAMS-based damping estimation methods". In: *2017 IEEE Manchester PowerTech, Powertech 2017*. Institute of Electrical and Electronics Engineers Inc., July 2017. ISBN: 9781509042371. DOI: 10.1109/PTC.2017.7980999.

[7] Kjetil Uhlen et al. "Application of Linear Analysis for Stability Improvements in the Nordic Power Transmission System". In: *2003 IEEE Power Engineering Society General Meeting, Conference Proceedings*. Vol. 4. 2003. DOI: 10.1109/pes.2003.1270938.

[8] P Kundur and N J Balu. *Power System Stability and Control*. EPRI power system engineering series. McGraw-Hill, 1994. ISBN: 9780780334632.

[9] Richard G. Lyons. *Understanding Digital Signal Processing*. Third Edition. Prentice Hall, 2011.

[10] IEEE Power & Energy Society. *IEEE Standard for Synchrophasor Data Transfer for Power Systems C37.118.2*. Tech. rep. 2011.

[11] J.F Hauer, C. J. Demeure, and L. L. Scharf. "Initial results in prony analysis of power system response signals". In: *IEEE Transactions on Power Systems* 5.1 (1990), pp. 80–89. ISSN: 15580679. DOI: 10.1109/59.49090.

[12] Janne M. Seppanen, Liisa C. Haarla, and Jukka Turunen. "Modal analysis of power systems with eigendecomposition of multivariate autoregressive models". In: *2013 IEEE Grenoble Conference PowerTech, POWERTECH 2013*. 2013. ISBN: 9781467356695. DOI: 10.1109/PTC.2013.6652230.

[13] S. A. Nezam Sarmadi and Vaithianathan Venkatasubramanian. "Electromechanical mode estimation using recursive adaptive stochastic subspace identification". In: *IEEE Transactions on Power Systems* 29.1 (Jan. 2014), pp. 349–358. ISSN: 08858950. DOI: 10.1109/TPWRS.2013.2281004.

[14] Jose Nathan Kutz et al. *Dynamic mode decomposition : data-driven modeling of complex systems*. Society for Industrial and Applied Mathematics, 2016. ISBN: 9781611974492.

[15] Emilio Barocio et al. “A Dynamic Mode Decomposition Framework for Global Power System Oscillation Analysis”. In: *IEEE Transactions on Power Systems* 30.6 (Nov. 2015), pp. 2902–2912. ISSN: 08858950. DOI: 10.1109/TPWRS.2014.2368078.

[16] Marcelo De Castro et al. “Experiences with Dynamical Mode Decomposition for Wide-Area Mode Estimation”. In: *2022 10th Workshop on Modelling and Simulation of Cyber-Physical Energy Systems, MSCPES 2022*. Institute of Electrical and Electronics Engineers Inc., 2022. ISBN: 9781665468657. DOI: 10.1109/MSCPES55116.2022.9770157.

[17] Annalisa Liccardo et al. “Detection and Analysis of Inter-Area Oscillations Through a Dynamic-Order DMD Approach”. In: *IEEE Transactions on Instrumentation and Measurement* 71 (2022). ISSN: 15579662. DOI: 10.1109/TIM.2022.3186371.

[18] Abdullah Alassaf and Lingling Fan. “Randomized Dynamic Mode Decomposition for Oscillation Modal Analysis”. In: *IEEE Transactions on Power Systems* 36.2 (Mar. 2021), pp. 1399–1408. ISSN: 15580679. DOI: 10.1109/TPWRS.2020.3012419.

[19] Alberto Berizzi et al. “Real-Time identification of electromechanical oscillations through dynamic mode decomposition”. In: *IET Generation, Transmission and Distribution* 14.19 (Oct. 2020), pp. 3992–3999. ISSN: 17518687. DOI: 10.1049/iet-gtd.2020.0202.

[20] N Halko, P G Martinsson, and J A Tropp. “Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions”. In: *SIAM Review* 53.2 (2011), pp. 217–288. DOI: 10.1137/090771806. URL: https://doi.org/10.1137/090771806.

[21] N Benjamin Erichson et al. “Randomized Dynamic Mode Decomposition”. In: *SIAM Journal on Applied Dynamical Systems* 18.4 (2019), pp. 1867–1891. DOI: 10.1137/18M1215013. URL: https://doi.org/10.1137/18M1215013.

[22] Orlando Delgado Fernández, Sini Tiistola, and Azwirman Gusrialdi. “Real-Time Data-Driven Electromechanical Oscillation Monitoring using Dynamic Mode Decomposition with Sliding Window”. In: *IFAC-PapersOnLine*. Vol. 55. 9. Elsevier B.V., 2022, pp. 158–163. DOI: 10.1016/j.ifacol.2022.07.028.

[23] Gene H Golub and Charles F Van Loan. *Matrix computations*. Johns Hopkins series in the mathematical sciences. Johns Hopkins Univ. Press, 1996. ISBN: 0801854148.

[24] Jonathan H. Tu et al. “On dynamic mode decomposition: Theory and applications”. In: *Journal of Computational Dynamics* 1.2 (Dec. 2014), pp. 391–421. ISSN: 21582505. DOI: 10.3934/jcd.2014.1.391.

[25] Matan Gavish and David L. Donoho. “The optimal hard threshold for singular values is 4/sqrt(3)”. In: *IEEE Transactions on Information Theory* 60.8 (2014), pp. 5040–5053. ISSN: 00189448. DOI: 10.1109/TIT.2014.2323359.

[26] Charles R. Harris et al. *Array programming with NumPy*. Sept. 2020. DOI: 10.1038/s41586-020-2649-2.

[27] Wes McKinney. “Data Structures for Statistical Computing in Python”. In: *Proceedings of the 9th Python in Science Conference*. 2010. DOI: 10.25080/majora-92bf1922-00a.

[28] Pauli Virtanen et al. "SciPy 1.0: fundamental algorithms for scientific computing in Python". In: *Nature Methods* 17.3 (Mar. 2020), pp. 261–272. ISSN: 15487105. DOI: 10.1038/s41592-019-0686-2.

[29] John D. Hunter. "Matplotlib: A 2D graphics environment". In: *Computing in Science and Engineering* 9.3 (2007). ISSN: 15219615. DOI: 10.1109/MCSE.2007.55.

[30] Petre Stoica and Randolph Moses. *Spectral Analysis of Signals*. Prentice Hall, 2005.

[31] Alan V Oppenheim, Ronald W Schafer, and John R Buck. *Discrete-Time Signal Processing*. Prentice Hall, 1999.

[32] Tapio Schneider and Arnold Neumaier. "Algorithm 808: ARFIT - A Matlab package for the estimation of parameters and eigenmodes of multivariate autoregressive models". In: *ACM Transactions on Mathematical Software* 27.1 (Mar. 2001), pp. 58–65. ISSN: 00983500. DOI: 10.1145/382043.382316.

[33] Yoshihiko Susuki et al. "Online Koopman Mode Decomposition for Power System Synchrophasor Data". In: *IFAC-PapersOnLine*. Vol. 55. 9. Elsevier B.V., 2022, pp. 54–58. DOI: 10.1016/j.ifacol.2022.07.010.

[34] Leonardo Lima. *IEEE PES Task Force on Benchmark Systems for Stability Controls Report on the 2-area, 4-generator system*. Tech. rep.

[35] Michael Brown et al. "Characterizing and quantifying noise in PMU data". In: *IEEE Power and Energy Society General Meeting*. Vol. 2016-November. 2016. DOI: 10.1109/PESGM.2016.7741972.

[36] IEEE Power & Energy Society. *IEEE Standard for Synchrophasor Measurements for Power Systems C37.118.1*. Tech. rep. 2011.

[37] G. Ledwich and E. Palmer. "Modal estimates from normal operation of power systems". In: *2000 IEEE Power Engineering Society, Conference Proceedings*. Vol. 2. Institute of Electrical and Electronics Engineers Inc., 2000, pp. 1527–1531. ISBN: 0780359356. DOI: 10.1109/PESW.2000.850208.

[38] Hwei P. Hsu. *Probability, Random Variables, and Random Processes, Second Edition*. McGraw-Hill Education, 2011.

[39] Siemens PTI. *PSS/E Program Operation Manual*. Tech. rep. 2022.

A Python implementation of DMD

Below is included a Python function with the DMD implementation used in Section 5. Please refer to Section 2.2 for a detailed description and sources. A short example using the function to analyse a simple signal is also included.

A.1 Python DMD function

```
def DMD(X, dt, max_rank, stack_depth_pct, q=2, p=10):
    # Shift-stacking
    n, m = X.shape
    if stack_depth_pct<=0:
        Xs, orig_signals_idx = X, np.arange(X.shape[0])
        n_new, m_new = n, m
    else:
        stack_depth = int(0.01 * stack_depth_pct * m)
        n_new = n * stack_depth
        m_new = m - stack_depth + 1
        Xs = np.empty((n_new, m_new))
        for j in range(0, stack_depth):
            Xs[j:n_new:stack_depth,:] = X[:, j:m-stack_depth+1+j]
        orig_signals_idx = np.arange(0, n * stack_depth, stack_depth)
    k = min(n_new, m_new, max_rank)

    # Randomisation
    l = k + p
    G = np.random.random((m_new, l))
    Y = Xs @ G
    for ix in range(q):
        Q, _ = np.linalg.qr(Y)
        Z, _ = np.linalg.qr(Xs.conj().T @ Q)
        Y = Xs @ Z
    Q, _ = np.linalg.qr(Y)
    H = Q.conj().T @ Xs

    # DMD data matrices; SVD
    H1, H2 = H[:, :-1], H[:, 1:]
    U, s, Vh = np.linalg.svd(H1, full_matrices=False)

    # Hard-thresholding for rank
    beta = l/m_new  # rows/cols
    wbeta = 0.56*beta**3 - 0.95*beta**2 + 1.82*beta + 1.43
    tau = wbeta * np.median(s)
    idx_threshold = (s > tau).sum()
    k = min(idx_threshold, k)

    # Rank truncation; projection onto POD subspace
    U, s, V = U[:, :k], s[:k], Vh[:k, :].T
    Atilde = U.conj().T @ H2 @ V @ np.diag(1 / s)

    # Eigenanalysis, mode frequencies, damping
    Lambda, W = np.linalg.eig(Atilde)
    Phi = Q @ H2 @ V @ np.diag(1 / s) @ W
    Omega = np.log(Lambda) / dt
    Zeta = -np.cos(np.angle(Omega))

    # Initial DMD mode amplitudes
    h1 = H1[:, 0]
    b = np.linalg.lstsq(Phi, Q @ h1, rcond=None)[0]

    return Phi[orig_signals_idx], np.diag(Lambda), Omega, Zeta, b, s
```

A.2 Minimal DMD example

```
Minimal example of using DMD to analyse a static window of data.
"""
from matplotlib import pyplot as plt
from function_files.DMD import DMD
import numpy as np

# Signal with two channels, one mode & random phase
t = np.arange(250)*0.02
theta = np.random.uniform(-np.pi, np.pi, (2, 1))
X = np.exp(-0.1*t) * np.cos(2*np.pi*0.36*t + theta)

# Do DMD
Phi, Lambda, Omega, Zeta, b, s = DMD(X, 0.02, 4, 20)

# Pick out oscillatory frequencies
osc_idx = np.where(np.logical_and(Omega.imag>0, Omega.imag<2*np.pi))[0]
osc_idx = osc_idx[np.argsort(Omega.imag[osc_idx])]

print(f"True frequency:            0.36 Hz")
print(f"Estimated frequency:       {(Omega.imag[osc_idx]/np.pi/2).round(3)} Hz")
print(f"True damping ratios:       4.4 %")
print(f"Estimated damping ratios:  {(Zeta[osc_idx]*100).round(1)} %")
print(f"True mode amplitudes:      [1.0, 1.0] pu")
print(f"Estimated mode amplitudes: {np.abs(Phi*b).sum(axis=1)} pu")
print(f"True mode phases:          {(theta*180/np.pi).round(1).ravel()} degrees")
print(f"Estimated mode phases:     "
      f"{(np.angle(Phi*b)[:,0]*180/np.pi).round(1)} degrees")

# DMD signal reconstruction
t = np.arange(X.shape[1]) * 0.02
Xrec = np.real(Phi @ (b.reshape(-1, 1)
                      * np.exp(Omega.reshape(-1, 1) * t)))
plt.figure()
plt.plot(t, X.T, ":", linewidth=4)
plt.plot(t, Xrec.T)
plt.legend(["Original signal"]*2 + ["DMD reconstruction"]*2)
plt.xlabel("Time [s]")
plt.title("Comparing original signals with"
          "\nreconstructions based on DMD estimates")
plt.show()
```

B Stochastic load perturbation in PSS/E

This appendix provides details for the user-defined PSS/E load model used in Section 4.3 to introduce stochastic load variations into a PSS/E dynamic simulation. The load model was designed to read load variations from a file, and input files with random walks were produced by a Python script.

Included below are the Fortran code for the load model, and Python code to produce a random walk and save it to file, and a Python script for a PSS/E dynamic simulation. The dynamic simulation example uses the IEEE Two-Area benchmark system [8, 34] and is intended to be used with the case file included with [34], which can be downloaded at `https://cmte.ieee.org/pes-psdp/benchmark-systems-2/`.

A modified dynamics data (.dyr) file with added load models and governors is included below. Included in this file are load model specifications, the original contents of the IEEE Two-Area benchmark case with exciters adjusted to represent the «no TGR» case [8, 34], and added governors. The governors were only included to give some frequency regulation with a droop of 5%, and all other parameters are generic.

Please refer to the PSS/E documentation [39] for further details on user-defined models and PSS/E-specific Fortran functions.

B.1 Description of the user-defined load model FIL_BL

The FIL_BL model is designed to read a time series from file and introduce it as a load variation. File contents are assumed to be in per unit and can be scaled to PSS/E system base per unit with the factor CON(J+1). Each load model is assigned with a unique 4-digit identifier ICON(M+1), and reads from a file named by the convention «DATA<filenum>.DAT». All CONS and ICONS for the FIL_BL model are listed in Tables B.1 and B.2.

The input file is opened during initialisation of a PSS/E dynamic simulation, and a new value is read from the file for each simulation step and added to any existing load specified in the load-flow case. The voltage characteristic of the existing load, as specified during conversion with the CONL() activity, will be kept. Additionally, frequency dependence can be specified with CON(J) if desired. The load variation has a constant power characteristic.

The FIL_BL load model can also be used independently, in which case it should be assigned to a zero-valued load in the load-flow case. Then only the load variation read from the file will be considered.

Table B.1: CONs for the PSS/E stochastic load model

Index	Name	Description
J	K_{freq}	Frequency sensitivity factor [0–1]
J+1	σ_{file}	Gain / standard deviation of file contents [pu Sbase]
J+2	$\cos\varphi$	Power factor of the load deviation [0–1]
J+3	T_{activate}	Time of activation [s]

Table B.2: ICONs for the PSS/E stochastic load model

Index	Name	Description
M	D_{trip}	Trip dual time step? (1 = yes)
M+1	filenum	Integer identifier of input file

Dual time steps will be triggered for all time steps after $t > T_{\text{activation}}$ if D_{trip} is set to 1. The dual time step is a feature of PSS/E where two outputs are given for a single time step in certain cases, notably if the simulation is stopped in order to apply a disturbance, but dual time steps can also be triggered manually. Enforcing dual time steps for every simulation step will increase the simulation time. It also affects the simulation: Setting $D_{\text{trip}} = 1$ will result in smoother frequency curves, but the voltage curves will evolve in a step-like fashion. Conversely, $D_{\text{trip}} = 0$ will give choppy frequency curves but smoother voltage curves. An exploration into the effects which forced dual time steps with the FIL_BL load model have on system states was outside the scope of this study, but may be necessary if such a load model is to be used in a different context. In this project, $D_{\text{trip}} = 1$ was used. To minimise the impact of the forced dual time steps, the simulation period was set to 1/300 s and the output data was downsampled to 60 Hz. This makes the load variation per time step smaller since the random walk can be scaled using (2.65).

The Fortran code below was compiled using the PSS/E environment manager with the Fortran compiler preprocessor (\fpp) included. The compiled model was tested with PSS/E version 35.5 with the PSS/E runfile and the dyr file included below, with the unmodified raw case file from [34]. Section B.3 gives an example of how an input file with a random walk can be constructed.

B.2 Fortran code for PSS/E file-reading load model

```
#define Dtrip     ICON(M)       ! Flag to use dual time step at load step
#define filenum   ICON(M+1)     ! Integer identifier of input file
#define Kfreq     CON(J)        ! Frequency deviation
#define sigma     CON(J+1)      ! Stoch. process std dev / file input gain
#define cosPhi    CON(J+2)      ! Load deviation power factor
#define Tactivate CON(J+3)      ! Time of activation (s)
#define deltaI    VAR(L)        ! Load current deviation magnitude
#define Z         VAR(L+1)      ! Load deviation (from file input)
#define sinPhi    VAR(L+2)      ! Coefficient for load dev reactive part
#define fileunit  ICON(N)       ! Unit number for the input file

      subroutine FIL_BL(LD, SLOT, SLOT2)

      include 'COMON4.INS'
      implicit none
      integer ibus        ! Index of the bus where the load is connected
      integer J, L, M, N     ! Starting CON, VAR, ICON, reserved-ICON index
      integer LD          ! Load sequence number
      integer SLOT, SLOT2 ! Array allocation index for shared, private data
      real Vabs           ! Bus voltage magnitude
      real X              ! File input
      integer ioflag                    ! Flag for file IO
      integer already_open_fileunit    ! Unit number for file to be closed
      complex Sp, Si, Sy, Spiy  ! Constant power, current, adm. & total load
      complex S, deltaS         ! Corrected load, load dev. from init. value
      character(len= 4) filenumchar ! Char array for file identifier
      character(len=12) filename    ! Character array for the whole filename
      real CNSTCR, CNSTPQ
      external CNSTCR, CNSTPQ   ! functions defined in PSSE
      external SHOW_MODEL_INDICIES
      intrinsic ABS, AIMAG, CONJG, REAL, CMPLX, SQRT
      real, parameter :: pi = 3.1415927

C MODE 8

      if (MODE == 8) then
         J=1
         M=1
         ICON_DSCRPT(M)    ='Force dual timestep? (1=Yes)'
         ICON_DSCRPT(M+1)  ='Data file identifier number'
         CON_DSCRPT(J)     ='K, Frequency Sensitivity factor'
         CON_DSCRPT(J+1)   ='sigma, Stoch. process std dev./ file input gain'
         CON_DSCRPT(J+2)   ='cosPhi, Load deviation power factor'
         CON_DSCRPT(J+3)   ='Time of activation (s)'
         return
      endif

C MODES 1-7
      if (MODE <= 7) return


C SECONDARY ENTRY POINT  (T-ENTRY)
      entry TIL_BL(LD, SLOT, SLOT2)

      J = LDSTRT(1, SLOT)    ! starting CON index
      M = LDSTRT(2, SLOT)    ! starting ICON index
      L = LDSTR2(2, SLOT2)   ! starting VAR index
      N = LDSTR2(3, SLOT2)   ! starting reserved ICON index

      ibus=NUMLOD(LD)            ! Bus number of the load
```

```
      if (ibus <= 0) return      ! If load out of service, do nothing

C MODE 1 (initialization)

      if (MODE == 1) then
         deltaI = 0.0
         Z = CMPLX(0.0)               ! Initialize the input VAR
         fileunit = filenum+10000     ! File unit number
         sinPhi = SQRT(1-cosPhi**2)   ! Reactive part of load dev.
         if (Dtrip .NE. 1) Dtrip = 0  ! No forced dual timestep unless stated
         if (filenum > 9999) then     ! File ids > four digits not accepted.
             filenum = -1
             return

         else
             write(filenumchar,'(i4.4)') filenum    ! convert the file id
             filename = 'DATA'//filenumchar//'.DAT' ! whole filename

             ! Check if the file's already open and close it if it is
             inquire(file=filename, number=already_open_fileunit, opened=ioflag)
             if (ioflag > 0) close(already_open_fileunit)

             ! Open the file and store the unit number in the VAR array
             open(unit=fileunit, file=filename, status='old', action='read',
iostat=ioflag)

             ! Flag if unable to open file for some reason.
             if (ioflag > 0) fileunit = -1

         endif
      else

C MODE 3 (load current injection)
         Vabs = ABS(VOLT(ibus))
         ! If load not set to 0 in the load flow, use characteristic from CONL()
         ! activity. PQBRAQ describes how low the bus voltage can go while
         ! keeping the constant P, Q characteristic.
         if (Vabs < PQBRAK) then
            Sp = CNSTPQ(Vabs)*CLODFR(1,LD)
         else
            Sp = CLODFR(1,LD)
         endif

         ! Constant I characteristic is kept until bus voltage goes below 0.5 pu
         if (Vabs < 0.5) then
            Si = CNSTCR(Vabs)*Vabs*CLODFR(2,LD)
         else
            Si = Vabs*CLODFR(2,LD)
         endif

         ! Constant admittance part of the load
         Sy = Vabs*Vabs*CONJG(CLODFR(3,LD))
         ! Load before applying freq dependence and load dev
         Spiy = Sp + Si + Sy
         S = Spiy

         ! Load deviation
         if (TIME >= Tactivate) then
            if (ITER == 0)  then     ! Update only once per voltage solution
              if ((Dtrip == 1) .AND. (KPAUSE == 0)) then ! dual time step?
                  KPAUSE = 1
```

```
            KTRIP = 1
          endif

          if (Kpause .NE. 1) then   ! No update at first of dual time steps
          ! Read new load deviation from file. Skip if error or finished
              read(fileunit, *, iostat=ioflag) X
              if (ioflag == 0) Z = X * sigma

          endif
        endif
      endif
      deltaS = CMPLX(cosPhi, sinPhi) * Z
      S = S*(1.0 + Kfreq*BSFREQ(ibus)) + deltaS    ! Freq dep

      ! Store final values
      TPLOAD(LD) = REAL (S)
      TQLOAD(LD) = AIMAG(S)
      deltaI = ABS((S-Spiy)/VOLT(ibus))
      CURNT(ibus) = CURNT(ibus)-CONJG((S-Spiy)/VOLT(ibus)) ! Current inj
    endif
    return
    end subroutine FIL_BL
```

B.3 Construction of random walk

```
"""
Create a random walk & save it to file
"""

import numpy as np
id = 12     # File ID for the PSSE file-reading load model
fs = 300    # simulation step
sl = 60*15  # seconds Simulation length
n = fs * sl # Number of steps in walk
filename = 'DATA'+str(id).zfill(4)+'.DAT'
z = np.cumsum(np.random.randn(n).astype(np.float32))
z.tofile(filename, sep="\n", format='%.8g')
print("Created file ", filename, "with a", len(z), "step random walk")
```

B.4 PSS/E dynamics file

```
//  DYR file for IEEE Two-area model. Stochastic load model +
//    some governors added.

    7 'USRMDL' 1 'FIL_BL'   12  1    2   4   0   3   1
       0    12    0.00    0.0005    0.995    1.0
    / Dtrip datafile_number Kfreq sigma cosPhi Tactivate

    9 'USRMDL' 1 'FIL_BL'   12  1    2   4   0   3   1
       0    13    0.00    0.0005    0.995    1.0

// --- Contents of Benchmark_4ger_ESST1A_noTGR_2015.dyr -------
      1 'GENROE' 1     8.0000      0.30000E-01  0.40000      0.50000E-01
          6.5000       0.0000       1.8000       1.7000      0.30000
         0.55000      0.25000      0.20000      0.39200E-01  0.26720    /
      1 'ESST1A' 1          1            1      0.10000E-01   99.000
         -99.000       1.0000       1.000       1.0000       1.0000
          200.00       0.0000       4.0000      -4.0000       4.0000
         -4.0000       0.0000       0.0000       1.0000       0.0000
          3.0000    /
      2 'GENROE' 1     8.0000      0.30000E-01  0.40000      0.50000E-01
          6.5000       0.0000       1.8000       1.7000      0.30000
         0.55000      0.25000      0.20000      0.39200E-01  0.26720    /
      2 'ESST1A' 1          1            1      0.10000E-01   99.000
         -99.000       1.0000       1.000       1.0000       1.0000
          200.00       0.0000       4.0000      -4.0000       4.0000
         -4.0000       0.0000       0.0000       1.0000       0.0000
          3.0000    /
      3 'GENROE' 1     8.0000      0.30000E-01  0.40000      0.50000E-01
          6.1750       0.0000       1.8000       1.7000      0.30000
         0.55000      0.25000      0.20000      0.39200E-01  0.26720    /
      3 'ESST1A' 1          1            1      0.10000E-01   99.000
         -99.000       1.0000       1.000       1.0000       1.0000
          200.00       0.0000       4.0000      -4.0000       4.0000
         -4.0000       0.0000       0.0000       1.0000       0.0000
          3.0000    /
      4 'GENROE' 1     8.0000      0.30000E-01  0.40000      0.50000E-01
          6.1750       0.0000       1.8000       1.7000      0.30000
         0.55000      0.25000      0.20000      0.39200E-01  0.26720    /
      4 'ESST1A' 1          1            1      0.10000E-01   99.000
         -99.000       1.0000       1.000       1.0000       1.0000
          200.00       0.0000       4.0000      -4.0000       4.0000
         -4.0000       0.0000       0.0000       1.0000       0.0000
          3.0000    /

// --- Added governors (Not inlcuded in the IEEE case) ------------------
      1 'IEESGO' 1    0.10000E-01   0.0000      0.15000      0.30000
          8.0000      0.40000       20.000      0.70000      0.43000
          1.0000       0.0000      /
      2 'IEESGO' 1    0.10000E-01   0.0000      0.15000      0.30000
          8.0000      0.40000       20.000      0.70000      0.43000
          1.0000       0.0000      /
      3 'IEESGO' 1    0.10000E-01   0.0000      0.15000      0.30000
          8.0000      0.40000       20.000      0.70000      0.43000
          1.0000       0.0000      /
      4 'IEESGO' 1    0.10000E-01   0.0000      0.15000      0.30000
          8.0000      0.40000       20.000      0.70000      0.43000
          1.0000       0.0000      /
```

B.5 PSS/E run file

```
"""
Example PSSE runfile using the stochastic load model
"""

import os, sys
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt


PSSPY_LOCATION = r'C:\Program Files\PTI\PSSE35\35.5\PSSPY39'
sys.path.append(PSSPY_LOCATION)
PSSE_LOCATION = r'C:\Program Files\PTI\PSSE35\35.5\PSSBIN'
os.environ['PATH'] = os.environ['PATH'] + ';' + PSSE_LOCATION
import psse3505
import psspy
import dyntools
import redirect
redirect.psse2py()  # Redirect output to Python

# Setup
Fs = 60  # Hz target sampling rate
Simul_time_step = 1./300
Vstep_time = 900.
Simul_stop_time = 610.
Vstep_size = 1.

psspy.psseinit()
psspy.read(0, r"""Benchmark_4ger_33_2015.raw""")
psspy.dyre_new([1, 1, 1, 1], r"""Benchmark_4ger_noTGR_stoch_loads_fcr.dyr""", ""
, "", "")
psspy.addmodellibrary("""loadmodels.dll""")
psspy.chsb(0, 1, [-1, -1, -1, 1, 12, 0]) # Add Bsfreq channel for all buses
psspy.fnsl([0, 0, 0, 1, 1, 0, 99, 0])
psspy.cong(0)
psspy.conl(0, 1, 1, [0, 0], [100.0, 0.0, 0.0, 100.0])
psspy.conl(0, 1, 2, [0, 0], [100.0, 0.0, 0.0, 100.0])
psspy.conl(0, 1, 3, [0, 0], [100.0, 0.0, 0.0, 100.0])
psspy.ordr(0)
psspy.fact()
psspy.tysl(0)

outfile_title = "outp.out"
try:
    os.remove(outfile_title)
except OSError:
    pass

## PSSE dynamic simulation
psspy.dynamics_solution_param_2(intgar1=99,realar=[1.0,0.0001,Simul_time_step,0.
033333,0.05,0.11667,1.0,0.0005])
ierr = psspy.strt(0, outfile_title)
psspy.run(0,Vstep_time+10,0,1,0)
psspy.increment_vref(1, r"""1""", 0.03 * Vstep_size) # Steps in original
direction
psspy.increment_vref(2, r"""1""", -0.01 * Vstep_size)
psspy.increment_vref(3, r"""1""", -0.03 * Vstep_size)
psspy.increment_vref(4, r"""1""", 0.01 * Vstep_size)
psspy.run(0,Simul_stop_time+10,0,1,0)
```

```
def make_DataFrame(outfile_title):
    outfile = dyntools.CHNF(outfile_title)
    short_title, chanid_dict, chandata_dict = outfile.get_data()
    df = pd.DataFrame(chandata_dict)
    df.columns = chanid_dict.values()
    return df
df = make_DataFrame(outfile_title)

# Handle dual time steps by making the first one the average of them and then
removing the second one.
time_col = df.columns[["Time" in x for x in df.columns]]
dual_step2 = np.where(df.duplicated(subset=time_col))[0]
new_step1_values = (df.loc[dual_step2,:].to_numpy()+df.loc[dual_step2-1,:].
to_numpy())/2
df.loc[dual_step2-1, :] = new_step1_values
df = df.drop(dual_step2)

# Downsample & make time vector
q=int(1/Simul_time_step/Fs)
df = df.iloc[::q, :]
df.iloc[:,0] = np.arange(len(df))/Fs

plt.figure(); plt.plot(df.iloc[:,0], df.iloc[:, 1:5]);plt.show()

# Save the data
df.to_parquet("PSSE_output.parquet")



psspy.pssehalt_2()
```