Engineering Learning Initiatives Research Report

By Nathaniel Rogalskyj

Advisor: Professor David Bindel

Sponsor: John A. Swanson '61, MBE '62, MME '63

Abstract

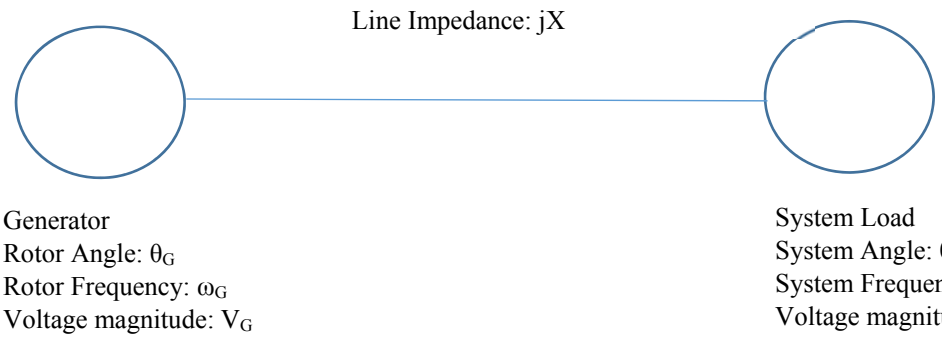
In the report, I summarize the research I did during the Engineering Learning Initiatives Summer Research program under Professor Bindel. The goal of the research was to correctly identify line failures in the electrical power grid in real-time given PMU (Phasor Measurement Units) sensor data. In the report, I discuss in order: (1) a model for the electrical power grid, (2) a general method for detecting a line failure using system identification, (3) different algorithms for the system identification and (4) my results regarding the comparison of these algorithms.

Introduction:

Section 1: Derivation of the Swing Equation

The electrical power grid can be modeled as a graph G=(V,E) where V are the vertices and E are the edges. V is the set of power producers/generators. They are also known as busses. E is the set of lines that connect the busses. Power is carried along the lines from generators to loads present in the line.

As an example of the power grids dynamics, consider a single bus, single load system.



A generator is characterized by its rotor angle θ_G and its rotor frequency ω_G. The system load is characterized by the system angle θ_{sys} and system frequency ω_{sys}. The system frequency in electrical power grids is maintained at a constant 50-60Hz. The line is characterized by the imaginary part of its impedance Im(Z). The following is the development of the swing equation as presented in [3].

Define the difference between the rotor angle and system angle:

$$\delta = \theta_G - \theta_{Sys}$$

Define the difference between the rotor frequency and the system frequency:

$$\omega_{diff} = \omega_G - \omega_{sys}$$

We assume that $\omega_{sys}$is constant, so upon taking a derivative with respect to time of both sides:

$$\frac{d}{dt}\omega_{\text{diff}} = \frac{d}{dt}\omega_G$$

It is important to model the generator mechanics. The governing equation for rotor mechanics relates the change in the frequency to damping proportional B to the frequency (rotor speed) ω_G and the net torque applied to the rotor,

which is the difference between the mechanical torque τ_m and the electrical counter torque τ_e, where J is the rotational inertia and B is the damping coefficient:

$$J\frac{d}{dt}\omega_G = -B\,\omega_G + (\tau m - \tau e)$$

Substitute for ω_G and $\frac{d}{dt}\omega_G$ to get:

$$J\frac{d}{dt}\omega_{diff} = -B\,(\omega_{diff} + \omega_{sys}) + \tau_m - \tau_e$$

Furthermore, substituting again yields

$$J\frac{d^2\delta}{dt^2} = -B(\frac{d\delta}{dt} + \omega_{sys}) + \tau_m - \tau_e$$

Multiplying both sides by ω_G, we get:

$$\omega_G J\frac{d^2\delta}{dt^2} = -\omega_G B\frac{d\delta}{dt} - B\,\omega_G\,\omega_{sys} + \tau_m\,\omega_G - \tau_e\,\omega_G$$

The mechanical power is $P_m = -B\,\omega_G\,\omega_{sys} + \tau_m\,\omega_G$ and the electrical power $P_e = -\tau_m\,\omega_G$. The electrical power is dependent on whether the system angle and rotor angle are in phase. Thereby $P_e = P_{max}\cos(\delta)$. We then get:

$$\omega_G J\frac{d^2\delta}{dt^2} = -\omega_G B\frac{d\delta}{dt} + P_m - P_{max}cos(\delta)$$

where $P_{max} = V_G V_{Sys}/X$. Let $M = \omega_G J$ and let $C = \omega_G B$. Then the above becomes:

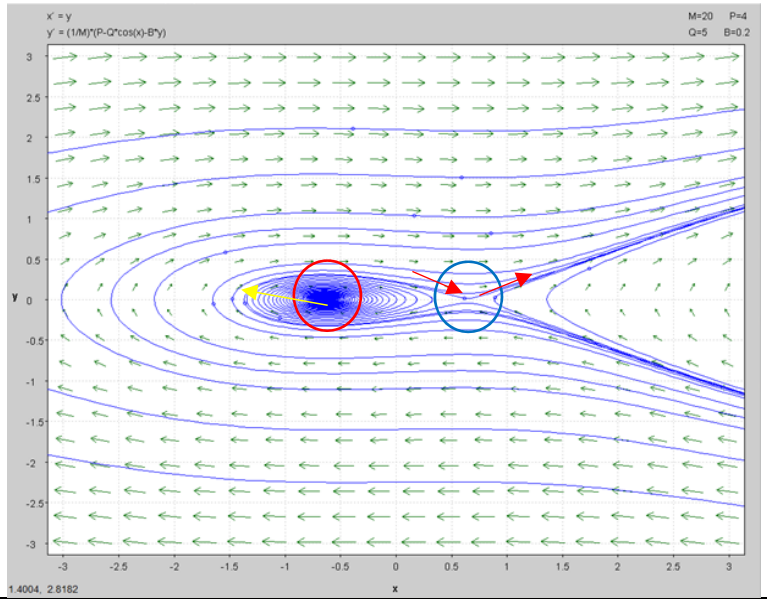
$$M\frac{d^2\delta}{dt^2} + C\frac{d\delta}{dt} = P_m - P_{max}cos(\delta)$$

Section 2: Analysis of Swing Equation and Generalization to Larger Networks

The system defined by the differential equation above can be looked geometrically in the phase plane.

Consider the (x,y) plane. At the point (x,y), assign the vector (x', y'). Given a point (x_0, y_0), representing an initial condition, the solution to the system can be found by tracing out a trajectory in the vector field. Tracing the trajectory out represents the evolution of the state with time. The geometric way of looking at a differential equation gives intuition into how a solution looks:

Figure 1



Phase Portrait of the one-bus, one load system. To find the solution to the differential equation given initial condition (x_0, y_0). Trace the curve outlined by the vectors starting at (x_0, y_0). In the neighborhood of a fixed point, the linearization of the fixed point gives the stability of the fixed point and approximates the non-linear vector field. Red circle represents neighborhood where linearization around stable fixed point is valid. Blue circle represents neighborhood where linearization around unstable fixed point is valid. Red arrows represent directions(eigenvectors) of linearization. Yellow arrow represents disruption of power grid from fixed point by line failure.

The system of differential equations defined by the swing equation is:

$$x' = y = f(x, y)$$
$$y' = \left(\frac{1}{M}\right)(P_m - P_{max}\cos(\mathrm{x}) - \mathrm{By}) = \mathrm{g(x,y)}$$

where $x = \delta$ and $y = \frac{d\delta}{dt}$. We solve for the equilibrium or the fixed point by setting the $x' = 0$ and $y' = 0$. In the figure above the fixed points are circled. If the system starts at a fixed point, the system stays at the fixed point. A fixed point can be stable or unstable. At a stable fixed point, if the initial condition is sufficiently close then the system evolves to the fixed. At an unstable fixed, if the initial condition is sufficiently close then the system moves away from the fixed point. The vector field above yields two fixed points given by (x,y): $(Arccos(\frac{P_m}{P_{max}}, 0)$ and $(-Arccos(\frac{P_m}{P_{max}}, 0)$. The stable and unstable fixed points are shown above. One can linearize at a fix point in order to acquire information about the stability of the fixed point and its directions.

It is important to delve into some theory.

In the n dimensional case, the vector field defined by a differential equation is written:

$$\boldsymbol{x}' = f(\boldsymbol{x})$$

where $f: R^n \to R^n$. The function f can be expanded at a using the Taylor series to get:

$$f(\boldsymbol{x}) \approx f(\mathbf{a}) + \mathrm{D}f(\boldsymbol{a})(\boldsymbol{x} - \boldsymbol{a}) + O((\boldsymbol{x} - \boldsymbol{a})^2)$$

where $\mathrm{D}f(\boldsymbol{a})$ is the Jacobian. Let f be of the form:

$$f(\boldsymbol{x}) = \begin{pmatrix} f_1(\boldsymbol{x}) \\ f_n(\boldsymbol{x}) \end{pmatrix} \text{ then the Jacobian is } \mathrm{D}f(\boldsymbol{a}) = \begin{bmatrix} \frac{\delta f_1(\boldsymbol{x})}{\delta x_1} & \cdots & \frac{\delta f_1(\boldsymbol{x})}{\delta x_n} \\ \vdots & \ddots & \vdots \\ \frac{\delta f_n(\boldsymbol{x})}{\delta x_1} & \cdots & \frac{\delta f_n(\boldsymbol{x})}{\delta x_n} \end{bmatrix} | \boldsymbol{x} = \boldsymbol{a}$$

Now let there be a fixed point **x*** which requires that $f(\boldsymbol{x}^*) = \boldsymbol{0}$. Consider the expansion of f near **x*** and consider disturbances around **x*** where $w = \mathbf{x}^* + \boldsymbol{\varepsilon}$. The $O((\mathbf{x}^* - \boldsymbol{w})^2) = O((\boldsymbol{\varepsilon})^2)$ term is very small. Hence there is a good approximation given by:

$$f(\boldsymbol{x}) \approx \mathrm{D}f(x^*)(\boldsymbol{x} - x^*)$$

for x near x^*. The proper way to characterize a linear system of the form:

$$\boldsymbol{x}' = A\boldsymbol{x}$$

is to calculate the eigenvalues λ_i and eigenvectors $\boldsymbol{v_i}$ of A. It is given that the solution is of the form:

$$\boldsymbol{x}(t) = \sum_{i=1}^{n} c_i \boldsymbol{v_i} e^{\lambda_i t}$$

The method can be applied to the one-bus, one load system. The calculation of the Jacobian at the fixed points yields one stable fixed point and one unstable fixed point for $P_m < P_{max}$. A plot of it is show above in Figure 1. In a neighborhood of each fixed point the corresponding eigenvalues determine its stability and the eigenvectors determine the directions the linearization, as is detailed by the blue and red arrows calculated from the linearization. Actual electrical power grids, however, are more complicated. Instead of a single generator and a single load, actual power grids have multiple generators and loads. For a power grid with n generators and loads, there will be 2n state variables since each generator has a rotor angle and rotor frequency associated with it. Furthermore, the electrical power P_e on each generator is determined by the network properties and the current rotor angle. The generalization of the simple one-buss, one-load system leads to a system of Differential Algebraic Equations (DAE). For the purpose of this paper, it is important to know that linearization around a fixed point remains a valid technique. For completeness though, the equations are detailed below:

The classical generator model is a second-order differential equation. The following is presented in [7]. For each of the m generators the equations are:

$$\delta_i = \omega_s \omega_i$$

$$\omega_{diff\,i} = \left(\frac{1}{2H_i}\right)(P_{mi} - P_{ei}(\delta_i, G) - B_i\omega_i)$$

where δ_i is the difference between the ith generator's rotor angle, ω_s is the system frequency, $\omega_{diff\,i}$ is the difference in generator-system frequency, H_i is the ith rotor inertia, P_{mi} is the mechanical power of the ith generator, $P_{ei}(\delta_i, G)$ is the electrical power exerted by the ith generator, a function of δ_i and the network G, B_i is the damping coefficient. The expression for P_{ei} is

$$\begin{cases} \begin{bmatrix} e'_{xi} \\ e'_{yi} \end{bmatrix} = \begin{bmatrix} \sin(\delta_i) & \cos(\delta_i) \\ -\cos(\delta_i) & \sin(\delta_i) \end{bmatrix} \begin{bmatrix} 0 \\ e'_{qi} \end{bmatrix} \\ I_t = Y\left[e'_{x1}, e'_{y1} \ldots e'_{xm}, e'_{ym}\right] \\ Yi \text{ is the ith row of admitance matrix of } G \\ \text{Denote } I_t = [i_{x1},\ i_{y1} \ldots i_{xm},\ i_{ym}]^T \\ \begin{bmatrix} e_{xi} \\ e_{yi} \end{bmatrix} = \begin{bmatrix} e'_{qi}\cos(\delta_i) \\ e'_{qi}\sin(\delta_i) \end{bmatrix} - \begin{bmatrix} \mathrm{R}_{ai} & -\mathrm{X}_{di}' \\ X' & \mathrm{R}_{ai} \end{bmatrix} \begin{bmatrix} i_{xi} \\ i_{yi} \end{bmatrix} \\ P_{ei}(\delta_i, G) = e_{xi}i_{xi} + e_{yi}i_{yi} \end{cases}$$

where e'_{qi} is the field voltage complex magnitude of generator i which is determined by initial conditions and is given; e'_{xi} and e'_{yi} are the internal bus voltages along the x (real) and y (imaginary) axes of the generator i; Y_i is the ith row of the admittance matrix Y. The nodes of the admittance matrix are the generators. Constant impedance loads are incorporated into the admittances. For an example and setup of a problem, see [7]. I_{ti} is the terminal current of generator i. R_{ai} and X_{di} are the source impedance of generator I; e_{xi} and e_{yi} are the terminal voltages of the generator i along the x and y axes.

Unlike in the single generator, single bus case, there are multiple stable equilibrium. The phase space with 2m dimensions is significantly more complex and harder to visualize. However, the method of linearization continues to work and give us information about the dynamics in the neighborhood of a fixed point.

Problem Statement:

The goal of the project is to determine line failures in the electrical power grid. More formally, given prior knowledge of the electrical grid modeled as a graph G=(V,E), where V are busses and E are lines, and a stream of output data $y(t) \in R^m$ from m PMUs that provide the phasor- phase and voltage magnitude- of a nearby generator, we hope to detect line failures on the grid.

When there is a line failure on the grid, it is equivalent to a disruption of the grid from its steady state or fixed point. The yellow arrow in Figure 1 describes the disruption. Transient signals provide information about the line failure as the system settles to some stable fixed point. The assumption is that the disruption of the grid from its fixed point is not far from some other fixed point. As discussed, the dynamics in the neighborhood of the fixed point are approximated linearly. The transients that can be read from the PMUs take the form given above:

$$\boldsymbol{y}(t) \approx \sum_{i=1}^{d} c_i \boldsymbol{v_i} e^{\lambda_i t}$$

Given prior knowledge of the network, the modes λ_i and mode shapes $c_i \boldsymbol{v_i}$ of each specific line failure can be determined. The only thing necessary then is to find a way of extracting the modes and mode shapes from noisy $\boldsymbol{y}(t)$ and comparing them against predetermined modes from each specific line failure.

However, the method must also be robust to noise. The final statement is then that we would like to extract the modes given an observation

$$\hat{\boldsymbol{y}}(t) = \boldsymbol{y}(t) + \boldsymbol{e}_t$$

where e_t is an error vector:

$$e_t = \begin{cases} p, & white\ noise\ std\ \sigma \\ 1-p, & gross\ outlier, norm\ distribution\ std\ \alpha \end{cases}$$

e_t is modeled in this manner recognizing that under normal operation the PMU error is white noise and then in other cases the PMU breaks and the error is normally distributed except with a different, larger standard deviation α. So p is a measure of how often it breaks, and α is a measure of how much it breaks.

Mode Extraction Algorithms:

As stated above, the goal is to extract the modes and mode shapes from the data $\hat{y}(t)$. There are two families of algorithms: Non-Subspace Methods and Subspace Methods. Both are discussed below:

Non-Subspace Methods:

We looked at a number of Non-Subspace based methods. The major method was Prony and the others were studied in a paper [Non Subspace] based around evaluation of an output-output transfer function at an already known mode λ_i

Prony:

The main non-subspace method that we examined is Vector Prony with Scalar fitting. Vector Prony with Scalar Fitting is presented below:

Given data $\{y_0, y_2, \ldots y_{w-1}\}$ where $y_i \in R^p$ sampled at rate Δt nd a guess at number of modes n, extract the modes.

Simplifying notation, allow $\boldsymbol{v_i}$ to designate the mode shape described above, previously designated $c_i\boldsymbol{v_i}$. Assume that:

$$\hat{\boldsymbol{y}}(t) = \sum_{i=1}^{n} \boldsymbol{v_i} e^{\lambda_i t}$$

The above equation is the solution to a linear difference equation where:

$$\hat{\boldsymbol{y}}_{\boldsymbol{t}} = a_1\hat{\boldsymbol{y}}_{\boldsymbol{t-1}} + a_2\hat{\boldsymbol{y}}_{\boldsymbol{t-2}} + \cdots + a_n\hat{\boldsymbol{y}}_{\boldsymbol{t-n}}$$

If we are given w data points, there becomes a system of w-n linear equations and n unknowns.

If w-n > n, we can solve for the coefficients $\{a_1, a_2 \ldots a_n\}$ using overdetermined/determined least squares. Having $\{a_1, a_2 \ldots a_n\}$ allows for the solving of the modes to the differential equation by solving for the roots of the characteristic polynomial.

$$P(z) = \boldsymbol{z^n} - [a_1\boldsymbol{z^{n-1}} + \cdots + a_{n-1}\boldsymbol{z} + a_n] = 0$$

The solved z_i are the modes to the difference equation. To recover the modes to the continuous equation, it is necessary to take the natural log of z so:

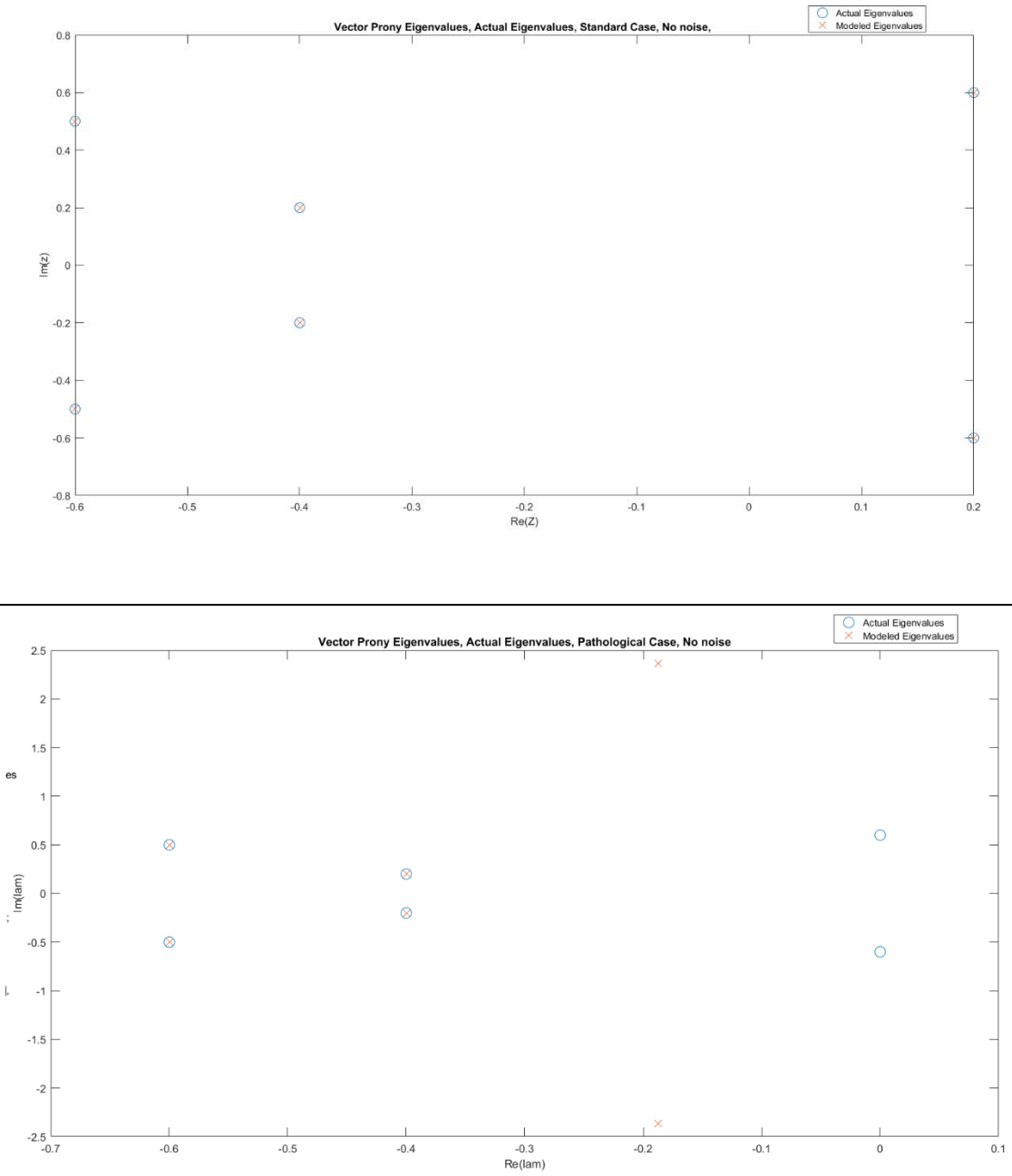
$$\lambda_i = \frac{\ln(z_i)}{\Delta t}$$

To recover the vectors $\boldsymbol{v_i}$, use the above equation to fit the modes. Solve the following using least squares.

$$\begin{bmatrix} {z_1}^1 & \cdots & {z_n}^1 \\ \vdots & \ddots & \vdots \\ {z_1}^w & \cdots & {z_1}^w \end{bmatrix} \begin{bmatrix} \boldsymbol{v_1}^T \\ \vdots \\ \boldsymbol{v_n}^T \end{bmatrix} = \begin{bmatrix} \boldsymbol{y_1}^T \\ \vdots \\ \boldsymbol{y_w}^T \end{bmatrix}$$

One measure of the success of Prony that is easy to visualize is to plot the eigenvalues recovered by Vector Prony against the actual eigenvalues. For arbitrary eigenvalues and limited noise, Prony works well as shown in Figure 2. The analysis also gives insight into instances in when Prony is unable to distinguish modes in the data. Such occurs when the modes have little weight in the finite amount of data that is recorded. One such case is when $\mathrm{Re}(\lambda) \approx 0$ and $||\boldsymbol{v_i}|| \approx 0$. Such a case is of little importance if $\mathrm{Re}(\lambda) < 0$ implying that the mode is damped out. But if $\mathrm{Re}(\lambda) > 0$ then over time the mode will have further weight in the data, yet could be unpredicted if the number of data points is small. Not detecting undamped modes is dangerous in Power Engineering, since undamped modes challenge stability. Consult Figure 2 for the pathological case, when this occurs with no noise.

Figure 2



(TOP) All eigenvalue-eigenvector pairs have weight in the data, namely the norm of the eigenvectors are not sufficiently close to zero or eigenvalues have Re(λ) not close to zero. (BOTTOM) Not all eigenvalues-eigenvectors pairs have weight in the data. The eigenvector-eigenvalue pair corresponding to the eigenvalue close to zero has eigenvector norm close to zero, and Re(λ) close to zero

With increasing levels of noise, Prony is less likely to accurately distinguish pathological eigenvalue-eigenvector pairs. Our ability to distinguish such modes is a function of the number of samples taken, w, and the noise level across all the channels $\boldsymbol{e}_t$. It is important then to derive an estimate for the number of samples needed to accurately determine an undamped mode. Consider a simplified input to Prony, where the order guess is n=1 and the data is generated by one mode, hence:

$$\hat{\boldsymbol{y}}(t) = \boldsymbol{v}e^{\lambda_i t} + \boldsymbol{e}_t$$

From the algorithm above this reduces to a scalar fit, where A=e^{λ_i}

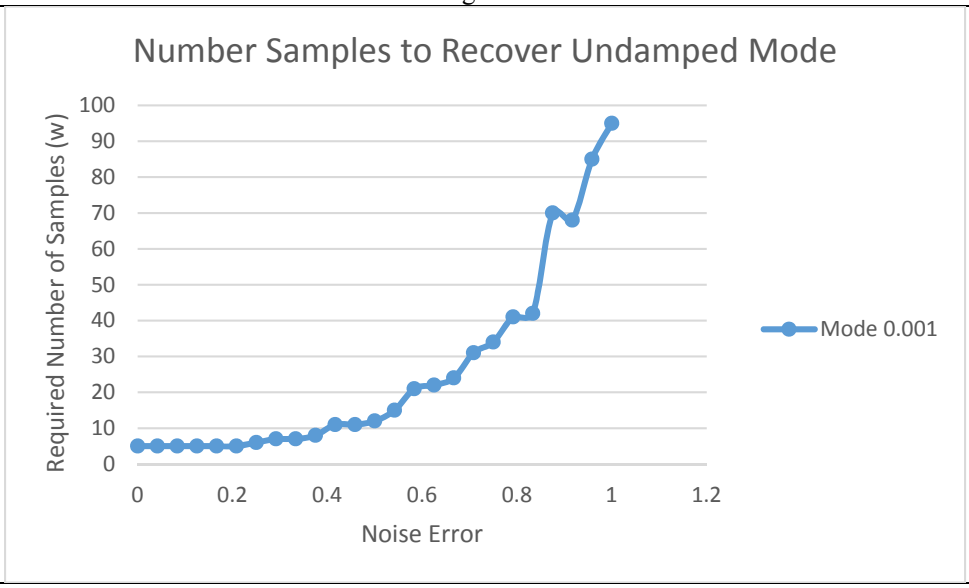
$$A(w) = \frac{\sum_{t=1}^{w}\sum_{i=1}^{n}[v_i e^{\lambda(t-1)} + \boldsymbol{e}_{t-1,\ i}][v_i e^{\lambda(t)} + \boldsymbol{e}_{t-1,\ i}]}{\sum_{t=1}^{w}\sum_{i=1}^{n}[v_i e^{\lambda(t-1)} + \boldsymbol{e}_{t-1,\ i}]^2}$$

Acquiring a bound on the variance would provide information about the w that would provide enough samples to ensure a good approximation with an eigenvalue of λ. So we look for a function for the variance of A in terms of w:

$$(Var(A))\ (w) = var(\frac{\sum_{t=1}^{w}\sum_{i=1}^{n}[v_i e^{\lambda(t-1)} + \boldsymbol{e}_{t-1,\ i}][v_i e^{\lambda(t)} + \boldsymbol{e}_{t-1,\ i}]}{\sum_{t=1}^{w}\sum_{i=1}^{n}[v_i e^{\lambda(t-1)} + \boldsymbol{e}_{t-1,\ i}]^2})$$

A numerical result is provided below

Figure 3



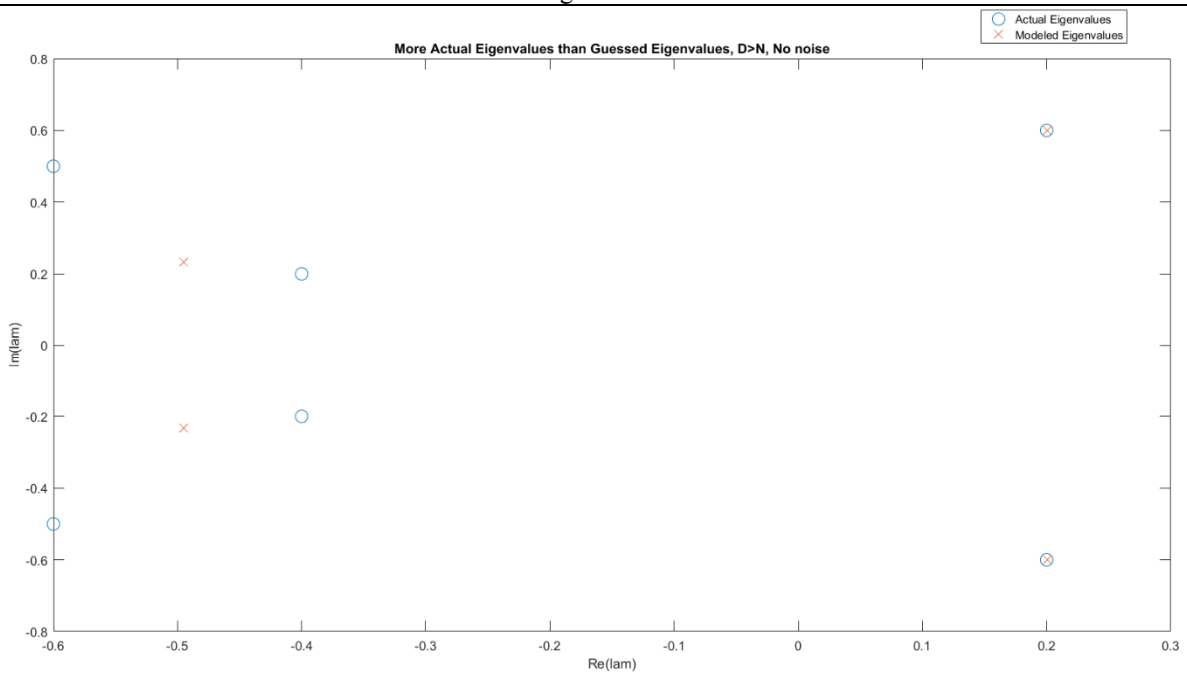
The figure presents how many samples w is needed to reduce the variance of A, Mode with $\lambda_i = 0.001$, to a tolerance of tol =0.01

Such analysis can be done in further work.

The fitting for the simple example above is a lower bound for the variance of a given mode in a muli-mode signal. To uncover the mode, the guessed order must be greater than the single mode example above. That means that there are more parameters to fit in the linear difference equations step and hence each individual mode has greater variance error. Therefore, the sample bound w for a given variance tolerance is a lower bound for more complicated signals.

Another interesting property is when the number of modes in the signal d is greater than the input parameter n.

Figure 3



Prony spaces the modeled eigenvalues relative to the weight each mode contributes to the data. As one can see, the eigenvalues with positive real components are fitted while there is a weighting between the eigenvalues with negative real components.

One other alternative Non-subspace Prony-based method that was attempted included fitting p x p square matrices $a_1, a_2 \ldots a_n$ instead of scalars $a_1, a_2 \ldots a_n$ and then solving the polynomial eigenvalue problem. First with the matrices $a_1, a_2 \ldots a_n$ obtained by solving least squares, form the same polynomial above P(z) and search for z such that P(z) is singular. The polynomial eigenvalue problem is able to return p*n eigenvalues. The method allows for better fitting due to the larger number of parameters. However, the extra fitting parameters caused fitting of noise and skewed eigenvalues. In system identification, such would be considered variance error.

Output-Output Transfer Function:

Given a mode λ_i estimated using a modal estimation technique, [2] provides a general method to estimate the mode shape. These methods rely on the following Output-to-Output Transfer algorithm. First, given a pair of measured outputs $y_k(t)$ and $y_l(t)$ construct a transfer function $G_{kl}(s)$. The transfer function $G_{kl}(s)$ is the linear mapping between the Laplace transform of $y_k(t)$ and $y_l(t)$ given by $G_{kl}(s)$=L($y_k(t)$)/L($y_l(t)$). The relative mode shape for mode λ_i can be found by evaluating $G_{kl}(\lambda_i)$.

A number of different approximations of the algorithm namely: Spectral Method, Frequency Domain Decomposition and the Channel Matching Method rely on the assumption that $\lambda_i = \sigma_i + j\omega \approx j\omega$, which as the

paper concludes is especially bad for high damping and leads to bias. The paper concludes that n4sid, presented below, is best.

Subspace Based Methods:

The other family of algorithms that were studied are Subspace Based Methods. Our results show that these methods are more robust than Prony or Output-Output Transfer Function methods.

Subspace Based Methods rely on the fact that all linear systems can be placed in the following form:

$$x_{t+1} = Ax_t + Bu_t + Ke_t$$
$$y_t = Cx_t + Du_t + e_t$$

where $x \in R^n$ is the state vector, A is the state propagation matrix, B is input matrix, K is the noise matrix, $y \in R^p$, is the observed output vector, and $u \in R^n$ is input vector. C is observation matrix and D is the feedthrough matrix. The matrices are defined up to a similarity transformation since one can pick the basis for x. More concretely if A, B, K, C are matrices of a linear system and P is an invertible change of basis matrix, then

$$A' = P^{-1}AP$$
$$B' = P^{-1}B$$
$$K' = P^{-1}K$$
$$C' = CP$$

are matrices that describe the same linear system. Furthermore, if one were given the matrices A, C, then one could find the modes and eigenvectors by the following:

$$\boldsymbol{v_i} = C\boldsymbol{v}_{A\_i}$$
$$\lambda_i = ln(\lambda_{A\_i})$$

Where λ_{A_i} , $\boldsymbol{v}_{A\_i}$ is the ith eigenvalue-eigenvector pair of A. The goal then is to find A, C of the linear system. In the current setup in the problem statement, note that there is no input $u \in R^n$ so the system to be found is:

$$x_{t+1} = Ax_t + Ke_t$$
$$y_t = Cx_t + e_t$$

There are a number of different algorithms that find A, C. We studied two: n4sid "Numerical Algorithms for Subspace State Space System Identification" and SSARX MLR "State-Space ARX with Multivariate Linear Regression".

Notation and Setup:

Given a list of vectors $x_1, x_2, \ldots . x_n$ the Hankel matrix is defined as:

$$X_{s|t,r} = \begin{bmatrix} x_s & x_{s+1} & x_{s+2} & \cdots & x_{s+r-1} \\ x_{s+1} & x_{s+2} & x_{s+3} & \cdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{t-1} & \cdots & \cdots & \cdots & x_{t+r-2} \end{bmatrix}$$

Define the observability matrix as:

$$\Gamma_i = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{i-1} \end{bmatrix}$$

Define the state matrix to be:

$$X_{r,s} = [x_r \quad x_{r+1} \quad \dots \quad x_{r+s-1}]$$

N4SID:

N4SID has three inputs: i and j and the data $\{y_{0,}\, y_{2,}\, \dots y_{w-1}\}$. The input i is the number of Hankel rows being used in the fit and should be selected to half as much useful data as possible taking in apriori knowledge of what types of modes to expect and the noise level. The parameter j should be an overestimate of the model order equivalent to the number of rows in the state vector. The presentation of the following is from: [5]

First, N4SID splits the deterministic-stochastic system above into two sub-systems: deterministic and stochastic. The deterministic system is:

$$x^d{}_{t+1} = Ax^d{}_t$$
$$y^d{}_t = Cx^d{}_t$$

The stochastic subsystem is:

$$x^s{}_{t+1} = Ax^s{}_t + Ke_t$$
$$y^s{}_t = Cx^s{}_t + e_t$$

By recursive substitution, we get:

$$Y_{0|i-1,j} = \Gamma_i X^d{}_{0,j} + Y^s{}_{0|i-1,j}$$
$$Y_{i|2i-1,j} = \Gamma_i X^d{}_{i,j} + Y^s{}_{i|2i-1,j}$$

Define the projection of future outputs onto the row space of past outputs to be:

$$Z = Y_{i|2i-1,j}/Y_{0|i-1,j}$$

where $A/B = AB^T(BB^T)^{-1}B$. Effectively,

$$Z = LY_{0|i-1,j}$$

where

$$L = min_R||Y_{i|2i-1,j} - RY_{0|i-1,j}||_2$$

which looks like the following:

$$\begin{bmatrix} y_i & y_{i+1} & y_{i+2} & \cdots & y_{i+j-1} \\ y_{i+1} & y_{i+2} & y_{i+3} & \cdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_{2i-1} & \cdots & \cdots & \cdots & y_{2i+j-2} \end{bmatrix} = R \begin{bmatrix} y_0 & y_1 & y_2 & \cdots & y_{j-1} \\ y_1 & y_2 & y_3 & \cdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_{i-1} & \cdots & \cdots & \cdots & y_{i+j-2} \end{bmatrix}$$

If the deterministic state $x^d{}_k$ and the $y^s{}_k$ are uncorrelated then

$$\lim_{j\to\infty} Y^s{}_{0|w,j}(X_{0,w})^T = 0$$

Furthermore, that implies that as $j \to \infty$:

$$Z = \Gamma_i X^d{}_{0,j}$$

Z then represents the row space of Γ_i. N4SID then takes a weighted SVD of Z to determine the model order. The weighting changes the basis for Γ_i, ideally such that effects of noise are minimized. W_1 must be full rank and $\text{rank}(Y_{0|i-1,j}) = rank(Y_{0|i-1,j}W_2)$

$$Z = SVD(W_1 Z W_2) = USV = [U_+ U_0] \begin{bmatrix} \Sigma_+ & 0 \\ 0 & \Sigma_0 \end{bmatrix} \begin{bmatrix} V_+ \\ V_0 \end{bmatrix}^T$$

Σ_+ represents the part where the singular values are non to close to zero, U_+, V_+ following accordingly. One can then take the observability matrix to be:

$$\Gamma_i = U_+ \Sigma_+^{1/2}$$

The matrix C can be read off from the first entry and by the shifted nature of Γ_i, one can acquire A by doing least squares:

$$\begin{bmatrix} CA \\ CA^2 \\ \dots \\ CA^{i-1} \end{bmatrix} A = \begin{bmatrix} C \\ CA \\ \dots \\ CA^{i-2} \end{bmatrix} A$$

It is shown that the standard deviation of A goes by:

$$\sigma \propto 1/j$$

and increasing i leads to better estimation of the stochastic subsystem.

SSARX MLR:

The other algorithm that was studied is State-Space Identification Using Multivariate Linear Regression. The algorithm was provided in [1] The input parameters are p and f.

There is additional notation for the algorithm:

$$\bar{A} = A - KC$$
$$\varphi_i = C\bar{A}^{i-1}K$$

First, we have that:

$$y_k = \sum_{i=1}^{\infty} \varphi_i \, y_{k-i} + e_k$$

If the modes of A are stable, namely all eigenvalues lie within the unit disk, then for some p, if i>p then

$$\bar{A}^i \approx 0$$

Therefore

$$y_k = \sum_{i=1}^{p} \varphi_i \, y_{k-i} + e_k$$

Then form the following:

$$\begin{bmatrix} {y_p}^T \\ {y_{p+1}}^T \\ \dots \\ {y_{N-1}}^T \end{bmatrix} = \begin{bmatrix} {y_0}^T & {y_1}^T & \dots & {y_{p-1}}^T \\ {y_1}^T & {y_2}^T & \dots & {y_p}^T \\ \vdots & \vdots & \ddots & \vdots \\ {y_{N-p-1}}^T & {y_{N-p}}^T & \dots & {y_{N-2}}^T \end{bmatrix} \begin{bmatrix} {\varphi_p}^T \\ {\varphi_{p-1}}^T \\ \dots \\ {\varphi_1}^T \end{bmatrix} + \begin{bmatrix} {e_p}^T \\ {e_{p+1}}^T \\ \dots \\ {e_{N-1}}^T \end{bmatrix}$$

The $\varphi_1, \varphi_2 \dots \varphi_p$ can be estimated and the $e_p, e_{p+1} \dots e_{N-1}$ can be estimated as the residuals of the Multivariate Linear Regression or a robust least squares procedure that negates the influence of outliers.

The algorithm then aims to calculate:

$$Z = \Gamma_f {X^d}_{p,n-f}$$

A formula derived by recursive substitution gives:

$$C\bar{A}^r x_s = y_{s+r} - e_{s+r} - \sum_{i=1}^{r} \varphi_i y_{s+r-i}$$

The formula can be put into matrix form in order to calculate Z, where r is chosen to be f. It is derived in the following manner:

$$Z = \begin{bmatrix} C \\ C\bar{A} \\ \dots \\ C\bar{A}^{f-1} \end{bmatrix} [x_p \quad x_{p+1} \quad \dots \quad x_{N-f}] = \begin{bmatrix} I & 0 & 0 & 0 \\ -\varphi_1 & I & 0 & 0 \\ \vdots & \ddots & \ddots & 0 \\ -\varphi_f & \dots & -\varphi_1 & I \end{bmatrix} \begin{bmatrix} y_p & y_{p+1} & \dots & y_{N-f} \\ y_{p+1} & y_{p+2} & \dots & y_{N-f+1} \\ \vdots & \ddots & \ddots & \vdots \\ y_{p+f-1} & y_{p+f} & \dots & y_{N-1} \end{bmatrix} - \begin{bmatrix} e_p & e_{p+1} & \dots & e_{N-f} \\ e_{p+1} & e_{p+2} & \dots & e_{N-f+1} \\ \vdots & \ddots & \ddots & \vdots \\ e_{p+f-1} & e_{p+f} & \dots & e_{N-1} \end{bmatrix}$$

As in N4SID, we take the SVD of Z in order to obtain Γ_f and $[x_p \quad x_{p+1} \quad \dots \quad x_{N-f}]$

$$Z = SVD(Z) = [U_+ U_0] \begin{bmatrix} \Sigma_+ & 0 \\ 0 & \Sigma_0 \end{bmatrix} \begin{bmatrix} V_+ \\ V_0 \end{bmatrix}^T$$

Find the observed states:

$$[\hat{x}_p \quad \hat{x}_{p+1} \quad \dots \quad \hat{x}_{p+1}] = \Sigma V_+$$

Calculate by using least squares on the following:

$$[\hat{x}_{p+1} \quad \hat{x}_{p+2} \quad \dots \quad \hat{x}_{N-f}] \approx [\bar{A} \;\; K] \begin{bmatrix} \hat{x}_p & \hat{x}_{p+1} & \dots & \hat{x}_{N-f-1} \\ \hat{y}_p & \hat{y}_{p+1} & \dots & \hat{y}_{N-f-1} \end{bmatrix}$$

$$[y_p - \hat{e}_p \quad y_{p+1} - \hat{e}_{p+1} \quad \dots \quad y_{N-f} - \hat{e}_{N-f}] \approx C\,[\hat{x}_p \quad \hat{x}_{p+1} \quad \dots \quad \hat{x}_{N-f}]$$

Simulation results provided in [] show that SSARX MLR outperforms N4SID. One thing that was studied was the difference between N4SID projection and the SSARX determination of the Markov parameters. One thing that I found was that N4SID seems to also estimate the same Markov parameters and it does so unintelligently.

As stated above for N4SID, the goal is to find the best fit R:

$$\begin{bmatrix} y_i & y_{i+1} & y_{i+2} & \dots & y_{i+j-1} \\ y_{i+1} & y_{i+2} & y_{i+3} & \dots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_{2i-1} & \dots & \dots & \dots & y_{2i+j-2} \end{bmatrix} = R \begin{bmatrix} y_0 & y_1 & y_2 & \dots & y_{j-1} \\ y_1 & y_2 & y_3 & \dots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_{i-1} & \dots & \dots & \dots & y_{i+j-2} \end{bmatrix}$$

In the terminology of SSARX MLR, R is the following matrix:

$$\begin{bmatrix} \varphi_i & \varphi_{i-1} & \varphi_{i-2} & \cdots & \varphi_1 \\ \varphi_{i+1} & \varphi_i & \cdots & \cdots & \varphi_2 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \varphi_{2i} & \varphi_{2i-1} & \cdots & \cdots & \varphi_i \end{bmatrix}$$

It shows that N4SID fits the same Markov parameters as SSARX MLR multiple times. Such extra degrees of freedom without any new model flexibility suggests that N4SID may be overfitting compared to SSARX MLR.

In addition to that analysis, it was also seen that SSARX MLR allows for easier use of standard regularization techniques in the estimation of the Markov Parameters

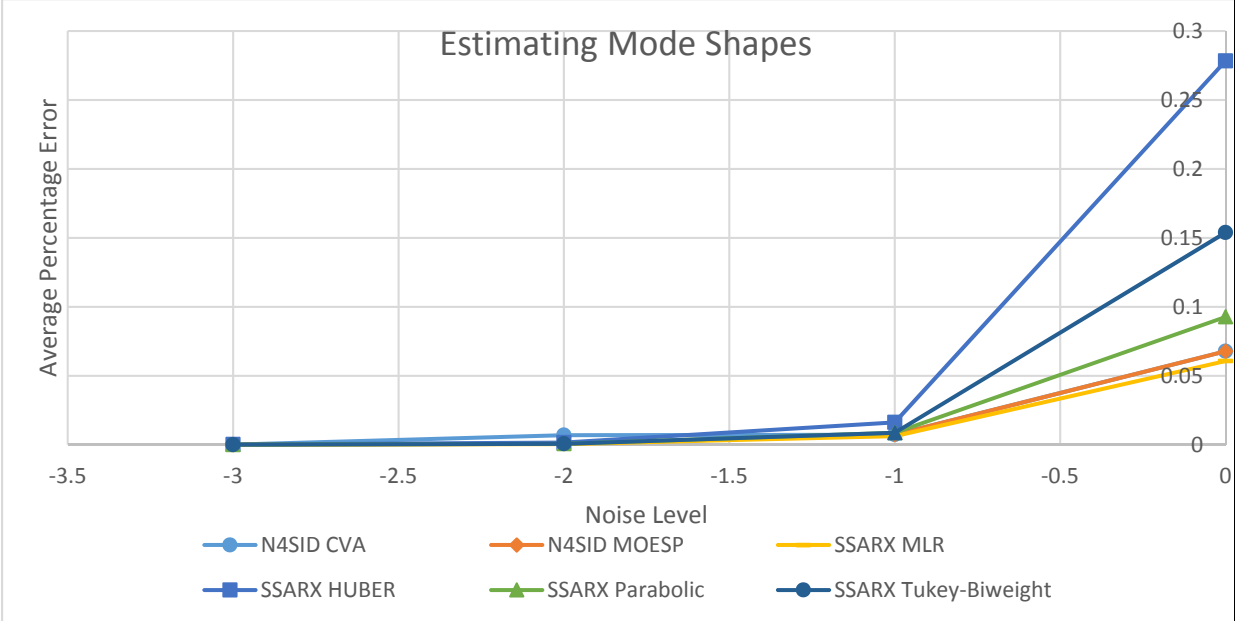
Results:

During the research, Vector Prony with Scalar Fitting, Vector Prony with Matrix Fitting, N4SID and SSARX MLR were implemented. The algorithms were tested on data generated by:

$$\hat{\boldsymbol{y}}(t) = \sum_{i=1}^{n} \boldsymbol{v_i} e^{\lambda_i t} + e_t$$

The best algorithms' determination of the modeshape and eigenvalue error were calculated. The results are provided below:

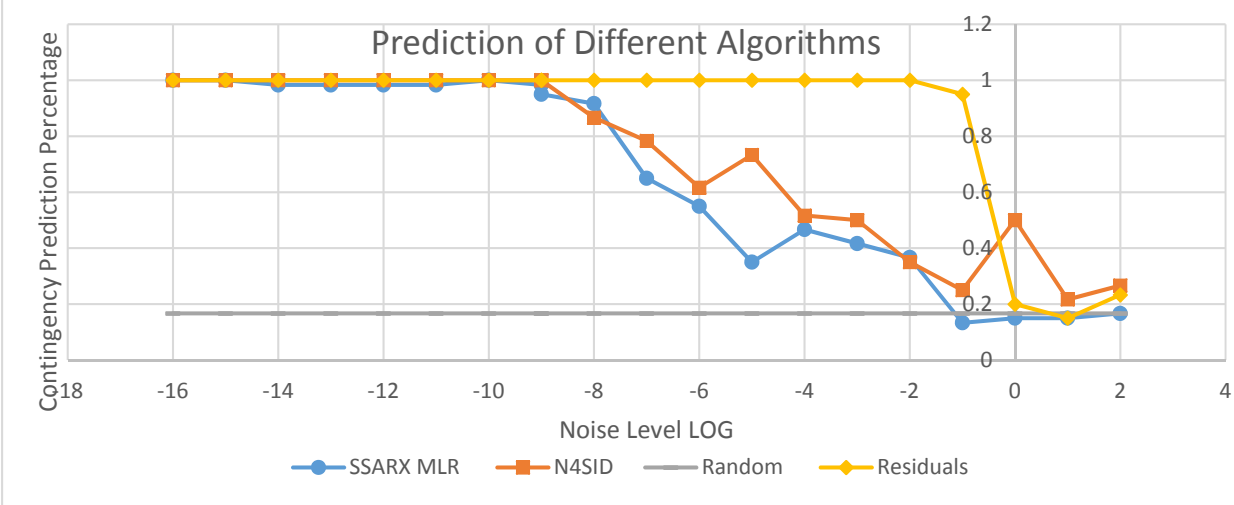
Figure 4



The below provides a comparison of the different implemented algorithms on data generated by modes. It is seen that N4SID MOESP, N4SID CVA and SSARX MLR are the best performers. The others likely suffer from the weighting of different data points that is not then accounted for when calculating the row space of the observability matrix. The algorithms are not representation independent.

The other natural goal would be to examine the method of calculating the modes of simulated data and matching them to noisy input data from a line contingency. Data from six different contingencies was simulated on a nine-bus network. Modes and mode shapes were calculated from each contingency and stored. Modes and modeshapes were then extracted from noisy data from a contingency. The mode shapes were then matched to the closest stored contingency mode shapes. The rate of accuracy was measured over many trials. Another algorithm was also tried, called the residuals method. In the method, noisy data from a contingency was compared directly to simulated contingency data. The contingency with the smallest noisy contingency-contingency data residual matrix norm was selected. That algorithm was also tested at different noise levels. Results provided below:

Figure 5



The figure shows that contingency identification based off comparison of mode shapes by both SSARX MLR and N4SID leave much to be desired. The methodology is highly sensitive to noise and is likely a result of close mode shapes for different contingencies. However, the simple comparison of minimum norm residuals yields an algorithm that is computationally easier and more predictive.

The algorithm that worked the best for the problem is the residuals method. Upon further thought, it makes sense that this is the case since the residuals method relies on all possible information, while the mode/modeshape extraction methods rely only on the linear part of the data. In addition, singularly matching mode-shapes or singularly matching modes without appropriate consideration for "other part" of the data can lead to poor results. For instance, suppose that the algorithm calculates a modeshape of $\boldsymbol{c_1 v_1} = \begin{bmatrix} 1 \\ 100 \end{bmatrix}$ with mode $\lambda_i = -100$ on the normal data such that the mode-shape has little weight in the data, and on the noisy data a small perturbation due to noise causes a new modeshape calculation of $\begin{bmatrix} 1 \\ 30 \end{bmatrix}$ with similar highly damped mode $\lambda_i = -100$. Without taking into consideration the damping of the mode and its implication of the modeshapes weight on the data, these modes which have little support in the data- highly susceptible to perturbation by noise- could lead to more varied results. An algorithm could be devised such that the modes susceptibility to noise was taken into account. However, this seems very similar to reconstructing the linear part of the data, which is entirely unnecessary since the data is given and can be used directly as in the residuals method.

Conclusion:

It seems that we can conclude a number of different things.

First, the Residuals method is decently accurate under noise and may be suitable for practical use in line contingency detection. The proposed method of matching modes/mode shapes between simulated data and input noisy data for using either SSARX MLR or N4SID is not practical and shows high sensitivity to noise.

In terms of extracting modes from data, results show that SSARX MLR may outperform N4SID. Interpretation of N4SID using the Markov parameters found in SSARX MLR shows that N4SID fits redundant Markov parameters, which may allow for greater variance error when estimating. Such may explain the results of [1] and my findings.

Further areas of study would be the following: (1) Why is matching mode shapes so sensitive to noise? (2) Characterizing more formally in one language and notation the various mode and mode shape extraction algorithms. (3) The required number of samples needed to get an accurate estimate for an undamped mode and mode shape under certain noise conditions.

References:

[1] Chen, Y. A Predictor From State Space Identification Algorithm Using Multivariate Linear Regression. Proceedings of the 31st Chinese Control Conference, 2012.

[2] Dosiek L, and Zhou N, Mode Shape Estimation Algorithms Under Ambient Conditions: A Comparative Review, IEEE Transactions on Power Systems, Vol. 28, NO 2, 2013

[3] Law, J.D. Development of the Swing Equation, http://www.ee.uidaho.edu/ee/power/jlaw/COURSES/ECE504PSS/HANDOUTS/Out-of-Step/s33f.pdf

[4] L. Ljung, System Identification: Theory for the User, 2nd Edition, Prentice Hall, 1999.

[5] Overschee and Moor, N4SID: Subspace Algorithms for the Identification of Combined Deterministic- Stochastic Systems, Automation, Vol. 311. No. I. pp. 75-93, 1994

[6] Trnka, P. Thesis Proposal, Czech Technical University in Prague, 2005

[7] Wang B, and Sun K, Power System Differential Algebraic Equations, arXiv, https://arxiv.org/vc/arxiv/papers/1512/1512.05185v1.pdf

[8] Zhou N, and Pierre J, Some Considerations in Using Prony Analysis to Estimate Electromechanical Modes