Universidad de la República
Facultad de Ingeniería

Algoritmos de estimación del estado de la red de potencia para su uso con sincrofasores.

Memoria de proyecto presentada a la Facultad de Ingeniería de la Universidad de la República por

Gonzalo Aristoy, Emilio Font, Sebastián Isaurralde

en cumplimiento parcial de los requerimientos para la obtención del título de Ingeniero Electricista.

Tutor

Álvaro Giusto..... Facultad Ingeniería - UDELAR

Tribunal

Álvaro Giusto..... Facultad Ingeniería - UDELAR
Ricardo Franco ..... Facultad Ingeniería - UDELAR
Ignacio Ramírez ..... Facultad Ingeniería - UDELAR

Montevideo
30/09/2014

Algoritmos de estimación del estado de la red de potencia para su uso con sincrofasores., Gonzalo Aristoy, Emilio Font, Sebastián Isaurralde.

Esta tesis fue preparada en LaTeX usando la clase iietesis (v1.1).
Contiene un total de 95 páginas.
Compilada el jueves 30 octubre, 2014.
http://iie.fing.edu.uy/

Resumen

El proyecto consistió en la búsqueda bibliográfica e implementación de un algoritmo de estimación de estado para las redes eléctricas de potencia utilizando la información que proporcionan los sistemas de medida y monitoreo más actuales instalados en las redes. Para resumir el aporte del proyecto, es necesario explicar previamente que los sistemas de medida y monitoreo se pueden agrupar en dos, primero se tienen los sistemas más antiguos llamados *Supervisory Control And Data Acquisition* (SCADA). En segundo lugar se tienen las redes de *Phase Measurement Unit* (PMU), estos son modernos equipos que incorporan medidas distintas y más rápidas que los SCADA. Este último tipo de medidas son el punto de partida del proyecto, que consistió en la incorporación de esta nueva tecnología en el arte de la estimación de estado.

El proyecto se dividió en tres grandes etapas, la primera consistió en la búsqueda de algoritmos que cumplieran con los requisitos buscados. La siguiente etapa consistió en implementar en Matlab uno de los algoritmos encontrados. En la última etapa se migró parte del código escrito en Matlab al lenguaje C++, con el objetivo de poder cuantificar los tiempos de ejecución.

El documento incluye una reseña histórica y una descripción de la estimación de estado clásica, basada únicamente en datos aportados por SCADA. Se realiza una introducción a los distintos tipos de algoritmos encontrados en la bibliografía, y se profundiza en el algoritmo implementado. Por último se presentan los resultados obtenidos para algunos casos de prueba.

Dichos resultados son satisfactorios y el algoritmo logra tener un seguimiento aceptable a las formas de onda en casos donde ocurren cortos circuitos trifásicos o cambios drásticos de carga. Es importante aclarar que la *performance* de la estimación depende de tres grandes factores, primero el tipo de falta, segundo la cantidad de PMU instalados, y tercero la posición relativa entre el lugar de la falta y los PMUs. Los tiempos de ejecución logrados en C++ son satisfactorios y fue posible aproximarse al valor ideal con ordenadores comerciales y sin optimizar el código.

Esta página ha sido intencionalmente dejada en blanco.

Convenciones Notacionales

- X:Vector de estado.
- $h(X)$: Función de estado.
- Z: Vector de medidas.
- V_i: Módulo de la tensión en la barra i.
- θ_i: Fase de la tensión en la barra i.
- P_i: Potencia activa inyectada en la barra i.
- Q_i: Potencia reactiva inyectada en la barra i.
- $X_f(t)$: Función sinusoidal para definición de fasor.
- X_{fm}: Amplitud de $X_f(t)$.
- ω: Frecuencia angulár.
- ϕ: Fase inicial.
- $\vec{X_f}$: Fasor que representa $X_f(t)$.
- X_{fr}: Parte real de X_f.
- X_{fi}: Parte imaginaria de X_f.
- $X_s(t)$: Función sinusoidal para definición de sincrofasor.
- X_{sm}: Amplitud de $X_s(t)$.
- f_0: Frecuencia nominal del sistema.
- ψ: Argumento de una sinusoide en función del tiempo.
- Y_{ik}: Admitancia de la línea i-k.
- Y_{ik}^S: Admitancia shunt a tierra de la línea i-k.

Esta página ha sido intencionalmente dejada en blanco.

Glosario

algoritmo Un algoritmo es un conjunto prescrito de instrucciones o reglas bien definidas, ordenadas y finitas que permite realizar una actividad mediante pasos sucesivos que no generen dudas a quien deba realizar dicha actividad. Dados un estado inicial y una entrada, siguiendo los pasos sucesivos se llega a un estado final y se obtiene una solución.

C++ Es un lenguaje de programación diseñado a mediados de los años 1980 por Bjarne Stroustrup. La intención de su creación fue el extender al exitoso lenguaje de programación C con mecanismos que permitan la manipulación de objetos. En ese sentido, desde el punto de vista de los lenguajes orientados a objetos, el C++ es un lenguaje híbrido [23].

Matlab MATLAB (abreviatura de MATrix LABoratory, "laboratorio de matrices") es una herramienta de software matemático que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio (lenguaje M) y servicio de especie [15].

pu El sistema por unidad (pu, per-unit en inglés) de cualquier cantidad se define como la relación entre esta cantidad y la cantidad base y se expresa como un decimal.

topología Conocer la topología de la red significa conocer: 1. que barras están en servicio, 2. que características tienen las líneas que unen dichas barras, 3. las características eléctricas de los transformadores , 4. la posición de los taps de los transformadores, 5. que generadores están en servicio , 6. conocer en que lugares y en que estado se encuentran los compensadores de reactiva, 7. demás características que hacen al conocimiento de la red.

UTE Empresa estatal uruguaya que administra y regula la generación, transmisión y distribución de energía eléctrica.

Esta página ha sido intencionalmente dejada en blanco.

Siglas

FE *Frecuency Error.*

PLC *Programmable Logic Controller.*

PMU *Phase Measurement Unit.*

RFE *Rate of Change of Frecuency Error.*

ROCOF *Rate Of Change Of Frecuency.*

RTU *Remote Terminal Unit.*

SCADA *Supervisory Control And Data Acquisition.*

TVE *Total Vector Error.*

UTC *Coordinated Universal Time.*

WLS *weighted least squares.*

Esta página ha sido intencionalmente dejada en blanco.

Tabla de contenidos

Tabla de contenidos

Capítulo 1

Introducción

Debido a las dimensiones, la variabilidad de las cargas y a la generación distribuida, las redes de potencia son sistemas interconectados muy complejos. La necesidad de un servicio ininterrumpido, de buena calidad, robusto y con una óptima gestión ha hecho que el monitoreo y control en tiempo real de los sistemas de potencia sea extremadamente importante y necesario. Dentro de las herramientas que se disponen para cumplir con los objetivos se encuentra *la estimación de estado*. La misma es una herramienta que permite estimar los módulos y fases de las tensiones en todas las barras del sistema partiendo de mediciones realizadas sobre la red. Actualmente la estimación de estado utilizada en los sistemas de potencia es la denominada *estimación de estado clásica*, que está desarrollada para resolver problemas estáticos.

1.1. Conceptos básicos

Una red de potencia está compuesta básicamente por generadores, transformadores y cargas, los cuales se conectan entre ellos mediante líneas de potencia. Se denominan *barras* a los nodos de este sistema interconectado.

El resultado de la estimación de estado técnicamente es un vector que contiene los módulos y fases de las tensiones en cada una de las barras de la red, al cual se le llama *vector de estado (X)*. Dicho vector es clave para luego obtener más información acerca del estado de la red.

1.1.1. Datos de entrada para la estimación de estados

Los datos de entrada para la estimación de estado son:

- Las características eléctricas y los valores numéricos de los modelos de los distintos elementos instalados en la red, como son: líneas de transmisión, transformadores, generadores, compensadores de energía reactiva, etc.

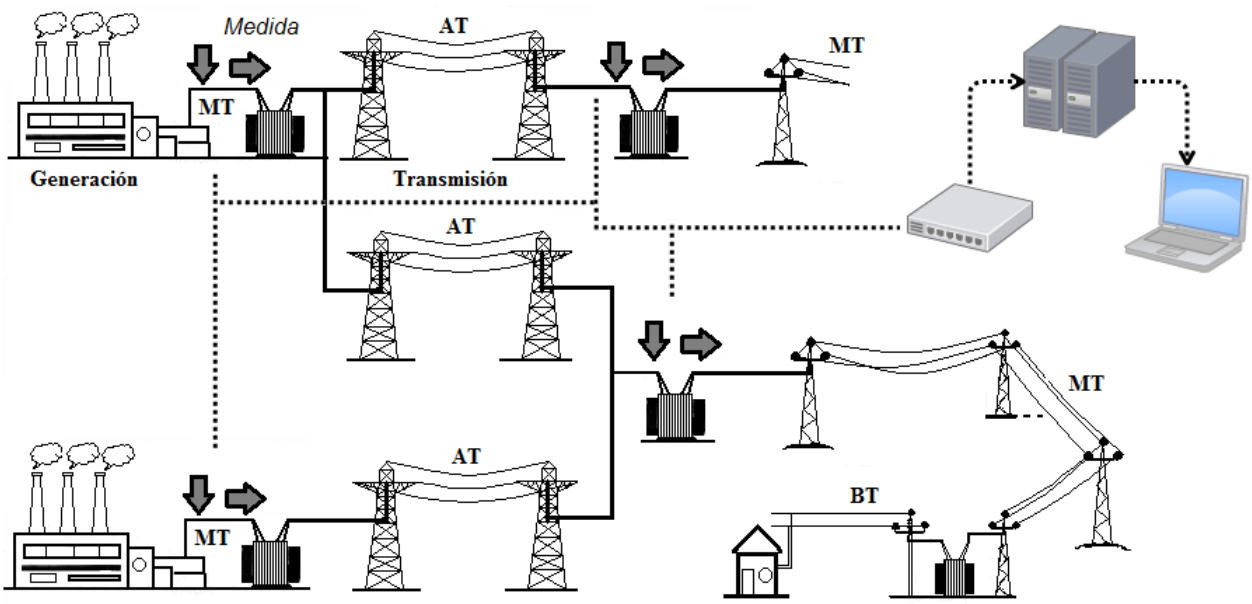


Figura 1.1: Esquema de sistema

- El estado de las conexiones en la red, por ejemplo conocer el estado de interruptores, seccionadores, taps de los transformadores, etc.
- Las medidas tomadas en la red, que pueden ser tensiones, corrientes y potencias, tanto en módulo como en fase.

Se dice que se conoce la *Topología de la Red* si se cumplen los dos primeros ítems. Con esta información se elabora la *función de estado* $h(X)$, que está formada por las ecuaciones de vínculo entre las medidas y el vector de estado.

Las medidas realizadas en las redes de potencia son el resultado de complejas redes de equipos, transductores y sensores instalados a lo largo de todo el sistema de potencia. En la figura 1.1 se puede observar un esquema simplificado de los puntos principales de medida. Aquí se ve una red de potencia típica, donde además de las líneas de potencia, tenemos una red datos que recolecta las medidas tomadas y las envía a una central de procesamiento.

Los sistemas de medida que aportan información y que serán analizados en el trabajo son los siguientes:

- **SCADA**: Sistema de medida y control utilizado desde mediados de los años 70 en las redes eléctricas de potencia [5]. Este sistema recolecta y centraliza una amplia cantidad de datos, de los cuales se destacan: de cada barra del sistema módulos de tensiones (V), potencias activas (P) y potencias reactivas (Q). Además proporciona información acerca de la topología de la red. Dichas medidas son reportadas cada algunos segundos, este lapso

depende caso a caso y de los eventos que ocurran en la red [20]. En caso de tener medidas en todas las barras de la red, se dice que el sistema de medida tiene una cobertura total de la red. En Uruguay la cantidad de barras de alta tensión (150kV y 500 kV) monitoreadas por SCADA se encuentra en el orden de las 120. A la estimación de estado en base al SCADA se le llamara *Estimación de Estado Clásica*, y esta sirve para resolver problemas estáticos.

- ***Redes de PMUs***: Sistema de medida moderno introducido comercialmente en el año 1992 en las redes de potencia [13]. Estos dispositivos son capaces de medir varios parámetros, de los cuales se destacan desde el punto de vista de la estimación de estado los módulos de la tensión y la corriente, y las fases de la tensión y la corriente. Estos equipos logran medir fase debido a que comparten una única referencia temporal, aunque dichos equipos estén separados por cientos de kilómetros. Combinando magnitud y fase de una misma variable nace el concepto de *Sincrofasor* que será profundizada en la sección 1.4.2. Por otra parte, las *PMUs* tienen la ventaja de reportar sus datos con tasas de velocidad muy rápidas comparadas con la del SCADA, pero tienen la desventaja de necesitar de una infraestructura muy costosa comparada con la del SCADA convencional. En Uruguay se están instalando 31 unidades y se prevé a corto plazo alcanzar las 100 unidades instaladas con esta funcionalidad. Es importante aclarar que el término *datos provenientes de una PMU* se utilizará equivalentemente como *llegada de sincrofasor* y *llegada de PMU* de aquí en adelante.

En el texto la nomenclatura que tendrán las variables que contienen las medidas, tanto sean del SCADA como de las PMU, será la letra Z, donde el subíndice indicará de donde provienen los datos.
Ambos sistemas de medida tienen conceptos en común y comparten equipos básicos de las redes, el siguiente punteo pretende resumir algunos de los elementos en común:

- Transformadores para la medida de tensión.
- Transformadores para la medida de corriente.
- Equipos que miden las salidas de los transformadores antes mencionados.
- Sistemas de comunicaciones. Los mismos pueden operar sobre diversas tecnologías y coexistir todos en la misma red.

1.1.2. Proceso de estimación

Una vez obtenidas las medidas realizadas en la red, la estimación de estado se resume en:

1. Una central que recolecta y procesa los datos.

2. Un algoritmo para calcular la estimación de estado de la red basado en los datos recolectados. Este tipo de algoritmos internamente deben resolver sistemas de ecuaciones no lineales (asociados a la función de estado *h(X)* no lineal) y para ello recurren a herramientas matemáticas como ser el método de *Gauss-Newton*. La salida de dicho tipo de algoritmos es el ya mencionado vector de estado *X*.

3. Algoritmos que utilizan la estimación de estado para ofrecer más datos útiles a los operadores de la red.

1.1.3. Aplicación de los resultados

Los resultados de la estimación de estado clásica les permite a los operadores de red tener información muy valiosa acerca de múltiples variables y procesos. A continuación se mencionan los principales resultados:

- Detectar diferencias en la topología de la red que fue extraída directamente del SCADA [27]. Esto puede ser debido a cambios en la red en zonas no observables, o por una posible falla en el sistema SCADA.
- Estimar las fases de las tensiones en todos los barras del sistema. Con el SCADA no es posible medir dichas fases directamente, es necesario algoritmos de estimación de estado para obtener dicha información [27].
- Detectar errores en las medidas. Al tener redundancia de información se pueden detectar inconsistencias y ubicar qué puntos de medida son erróneos [27]. Este tipo de datos erróneos son denominados como *Bad Data*.
- Mejorar las incertidumbres de las medidas debido a la redundancia de información [27].
- Conocer el despacho o flujo de potencia en la red en tiempo real [27].
- Minimizar la función de costo de generación [27].
- Maximizar el uso de compensadores de reactiva instalados en la red con el fin de liberar a los generadores de dicha tarea [27].

1.1.4. Incorporación de nuevas tecnologías a la estimación de estado

La estimación de estado hasta ahora fue descrita de forma independiente del sistema de medida, pero es hora de realizar una introducción detallada de los cambios y aportes que supone la incorporación de los PMU en la estimación de estado. Como ya fue mencionado, los datos aportados por las PMU son módulos de tensiones y corrientes con sus correspondientes fases respecto a una única referencia temporal. Se hace hincapié en que el SCADA no tiene la posibilidad de medir las fases de la tensión o la corriente directamente. Por otra parte los nuevos datos

recolectados por las PMU suponen nuevas ecuaciones de vínculo entre las medidas Z y el vector de estado X. Esto significa que la función de estado $h(X)$ será alterada y supone cambios en los algoritmos de estimación de estado desde el punto de vista algebraico y diferencial. Pero ese no es el único problema que se tiene por el hecho de incorporar los sincrofasores en la estimación de estado, el otro problema son los tiempos de ejecución requeridos cuando se intenta realizar estimación de estado sin perder datos de las PMUs.

Los datos provenientes de los PMUs tienen tasas de refresco del orden de los milisegundos, 20 ms en el caso de *UTE*, y se recuerda que este mismo lapso en los SCADA puede ser de varios segundos. Esto significa que los tiempos de ejecución de los algoritmos de estimación de estado clásica tienen como límite máximo tiempos del entorno del segundo. En cambio los sincrofasores se actualizan en decenas de milisegundos. Intentar que se ejecuten algoritmos de estimación de estado en tiempos menores a los mencionados es uno de los principales desafíos para el desarrollo de los mismos y la viabilidad de su aplicación. En la figura 1.2 se esquematiza que dentro de los lapsos de actualización del SCADA pueden entrar varios refrescos de datos provenientes de los PMU. Es claro que las exigencias de tiempos para los algoritmos de estimación de estado clásica son más holgados que un algoritmo que aspire a poder alimentarse de PMUs. Cabe aclarar que la figura es meramente ilustrativa y no significa que los algoritmos de estimación de estado clásica no se logren ejecutar en tiempos menores a los esbozados. Esto significa que la estimación de estado deberá superar los siguientes desafíos:

1. Modificar o crear algoritmos para incorporar los nuevos datos.

2. Lograr que dichos algoritmos se ejecuten en lapsos menores a algunas decenas de milisegundos.

Por último se menciona el hecho de que la observabilidad que actualmente tienen las PMU de las redes de potencia no es completa. Esto significa que no todas las barras de las redes son medidas directa o indirectamente por una PMU. Esto se debe a los elevados costos de infraestructura que supondría tener una red totalmente observable.

Por lo tanto en la actualidad se tienen dos sistemas de medida que tienen sus ventajas y desventajas muy marcadas y las mismas son:

- **SCADA** El mismo reporta gran cantidad de datos que hacen posible una observabilidad completa de la red. En cambio estos datos son reportados con tiempos en el orden del segundo.

- **PMU** Los mismos reportan una cantidad limitada de datos y no es posible una observabilidad completa de la red. En cambio estos datos son reportados de forma muy rápida. Es importante aclarar que estos dispositivos son capaces de aportar las fases de las magnitudes medidas, datos que antes eran imposibles de conocer.

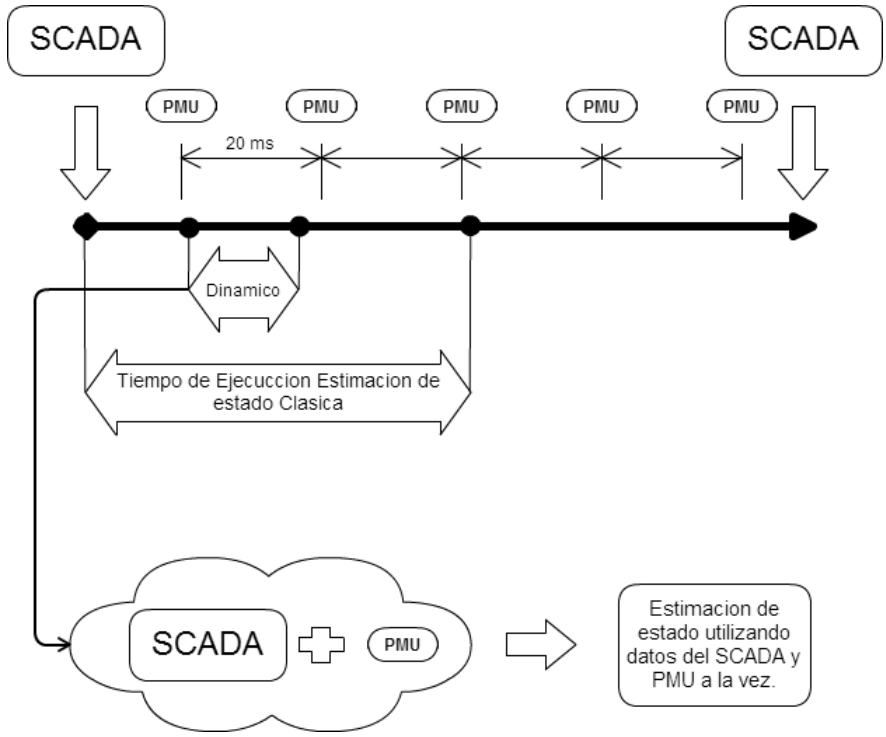


Figura 1.2: Esquema Conceptual de tiempos entre SCADA y PMU

1.2. Motivación

Como ya se mencionó en la introducción, las redes eléctricas están sufriendo un gran cambio desde el punto de vista de monitoreo, debido a la entrada en servicio de nuevos equipos (PMU) capaces no solo de medir desfasajes entre barras distantes geográficamente, sino también de realizar otras medidas y cumplir funcionalidades de protección. Antiguamente las únicas lecturas disponibles de las redes eran módulos de tensiones y corrientes, y flujos de potencia activa y reactiva. Con estos datos y un conocimiento en la topología de la red, se pueden emplear diversas técnicas matemáticas con el fin de conocer completamente el estado de la misma. Este tipo de algoritmos son los que fueron definidos como *estimación de estado clásica*.

Esta clase de algoritmos de estimación no tienen la capacidad de incorporar los datos que proporcionan las PMU. Es decir que sin el desarrollo de nuevos algoritmos la estimación de estado estaría desaprovechando la información de los sincrofasores, y perdiendo una oportunidad de desarrollar estimaciones en tiempo real. Esto motiva a la creación de nuevas herramientas que sean capaces de realizar una estimación en base a ambas fuentes de información, y fuerza a distintos actores de la sociedad a involucrarse, investigar y desarrollar en el área.

En el caso de Uruguay, está planificada la incorporación de PMUs en la red de Transmisión y esto motiva aún más la necesidad de generar experiencia en este tema.

1.3. Objetivos

Los objetivos del proyecto son:

- Estudio exhaustivo de algoritmos de estimación de estados. Esto comprende la búsqueda en revistas, tesis, artículos y bibliografía en general.
- Seleccionar un algoritmo que pueda estimar transitorios producidos por cambios leves en la red. Implementarlo en Matlab y ponerlo a prueba mediante un simulador de red eléctrica.
- Implementar el programa objetivo que ejecute dicho algoritmo bajo el lenguaje de programación C++.

Y los criterios de éxito son:

- Obtener resultados satisfactorios desde el punto de vista del cliente, de las simulaciones mediante MATLAB de los algoritmos seleccionados. Definimos por necesidades del cliente la capacidad de seguimiento de los estados de forma tal de poder percibir ciertas fallas simples.
- Migrar el código del algoritmo escrito en Matlab a un programa ejecutable capaz de realizar seguimiento a una tasa de refresco mayor a 1Hz.

Cambios en los objetivos:
A lo largo del proceso de desarrollo del proyecto, ciertos objetivos secundarios fueron desplazados. Más allá de que el plan original era estudiar entre 2 y 4 algoritmos de estimación previamente en Matlab para luego seleccionar uno de ellos para ser implementado en lenguaje C, se decidió implementar sólo uno en Matlab. Esta decisión fue tomada en conjunto con el tutor del proyecto, luego de la etapa de estudio bibliográfico y de la presentación del seminario. Allí se notó que realmente había un algoritmo que destacaba entre los demás y que debía ser atendido en exclusividad. La causa es buscar optimizar el tiempo del proyecto y los resultados esperados, focalizando toda la atención en el estudio y desarrollo de un solo algoritmo de estimación.

Por otra parte, se pretendía generar un sistema de estimación que presente un error menor al 1 % a lo largo del seguimiento en el tiempo. Este criterio de éxito no era alcanzable ya que las características de los cambios de estados en las barras son muy amplias, existiendo varios tipos de fallas, algunas suficientemente abruptas que hacen muy difícil seguirlas en tiempo real con el error antes mencionado.

1.4. Antecedentes y descripción de la temática

1.4.1. SCADA

SCADA, por sus siglas en inglés *Supervisory Control And Data Acquisition*, es un sistema informático que permite controlar y supervisar procesos industriales a distancia [5].

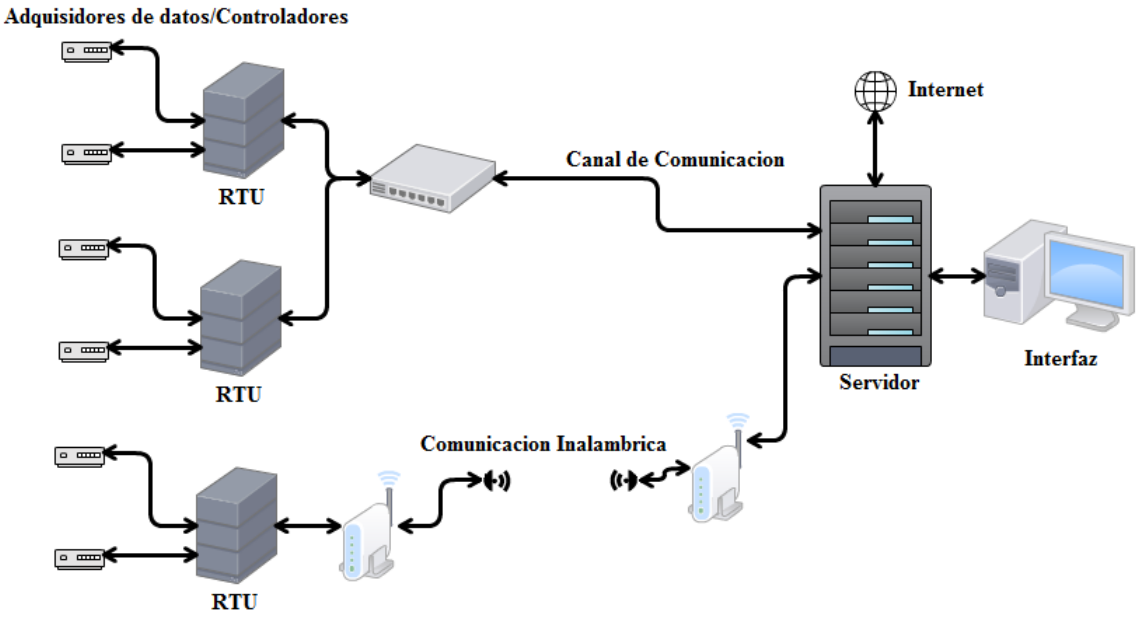


Figura 1.3: Esquema Conceptual de un SCADA

Técnicamente se le puede llamar SCADA a cualquier software que tenga acceso remoto a las medidas hechas de un proceso, al estado del mismo y que pueda intervenir en la planta [18]. El mismo realiza la tarea de interfaz entre equipos de control y medida y los de gestión y administración. La aplicabilidad del sistema SCADA no solo se limita a los sistemas eléctricos de potencia, también tiene gran penetración en otros procesos industriales [27].

En la figura 1.3 se puede observar un esquema básico de los elementos intervinientes en un sistema SCADA. Los principales elementos que se observan son, un servidor concentrador de datos, canales de comunicación y terminales remotas *Remote Terminal Unit* (RTU). Los adquisidores de datos y controladores son una representación de muchos tipos de dispositivos que tienen capacidad de realizar medidas y tareas de actuación en el sistema. Y estos a su vez se comunican con las RTU mediante algún método de comunicación local.

Alguna de las ventajas de los sistemas SCADA son:

- Las principales empresas encargadas de fabricar *hardware* y *software* tienen compatibilidad entre ellas, dando gran flexibilidad al sistema.
- Se concibe de manera modular, permitiendo que sea muy fácil la incorporación de nuevos elementos al sistema.
- Mediante la comunicación de los equipos *Programmable Logic Controller* (PLC) con el SCADA se logra tener control sobre todo tipo de actuadores y sensores.

- Los equipos remotos se pueden configurar para funcionar de manera independiente y hacer más robusto el sistema frente a desconexiones.

Como se ve en la figura 1.3 para que un sistema de control y automatización *SCADA* cobre vida son necesarios distintos elementos, de los cuales se destacan los equipos de medida, actuadores, equipos de comunicación, ordenadores para el procesamiento de datos y equipos para visualización e interfaz humana. Todos estos elementos están vinculados en menor o mayor medida por software para realizar las tareas de control, visualización, etc. [19].

La manera de gestionar estos procesos se puede dividir en dos categorías [18]:

- Sistemas Centralizados. Este tipo de sistemas son constituidos por unidades centrales que se encargan de tomar todas las decisiones y procesamientos a partir de los datos de las terminales. Dichas terminales no tienen capacidad de actuar por si solas, y su único cometido es interpretar órdenes y enviar las medidas a la central. A dichas terminales se les llama RTU.
- Sistemas Distribuidos. Este tipo de sistemas se constituye de sub-sistemas que tienen autonomía para procesar información y tomar decisiones. Las mismas pueden estar dentro de un sistema más grande y reportar sus medidas y estado a un sistema central. Este tipo de sistemas son más robustos y confiables.

En el caso de una red de potencia eléctrica, alguno de los equipos que componen la cadena de medida son:

- Transformadores de medida de Corriente.
- Transformadores de medida de Tensión.
- Medidores de potencia.
- Analizadores de red.
- Relés de protección y maniobra.
- Otros tipos de Relés.
- Otros tipos de sensores.
- PLC.

En particular las medidas que proporcionan los SCADA en las redes de potencia y son de interés en la estimación de estado son:

- Módulos de las tensiones en las barras $/\vec{V}/=V$.
- Potencias activas P.

- Potencias reactivas Q.
- Estado de la red en términos topológicos.

El problema de las medidas recolectadas por el SCADA es que tiene tiempos de actualización que están en el orden del segundo, y además la información no representa un mismo instante exactamente. Esto lo convierte en un sistema incapaz de seguir transitorios rápidos que pueda sufrir la red.

1.4.2. Sincrofasores

Estándar IEEE C37.118

La norma IEEE para sincrofasores es la C37.118 y su versión actual es la 2011. Esta norma consta de dos documentos, y a su vez en el 2014 surge una actualización.

- IEEE C37.118.1-2011: IEEE Standard for Synchrophasor Measurements for Power Systems [11].
- IEEE C37.118.2-2011: IEEE Standard for Synchrophasor Data Transfer for Power Systems [10].
- IEEE C37.118.1a-2014: Amendment 1: Modification of Selected Performance Requirements [12].

Para el desarrollo de esta tesis analizaremos el primer documento, el cual es el estándar para medidas de sincrofasores en sistemas de potencia. Aquí se definen los conceptos básicos de fasor sincronizado (sincrofasor), frecuencia y tasa de variación de la frecuencia (ROCOF por su sigla en inglés). Define también que es una unidad medidora de fase (PMU).

Definición de fasor

La representación de señales sinusoidales mediante el uso de fasores es común en el análisis de sistemas de potencia alterna. La onda sinusoidal es:

$$X_f(t) = X_{fm}\cos(wt + \phi) \ , \tag{1.1}$$

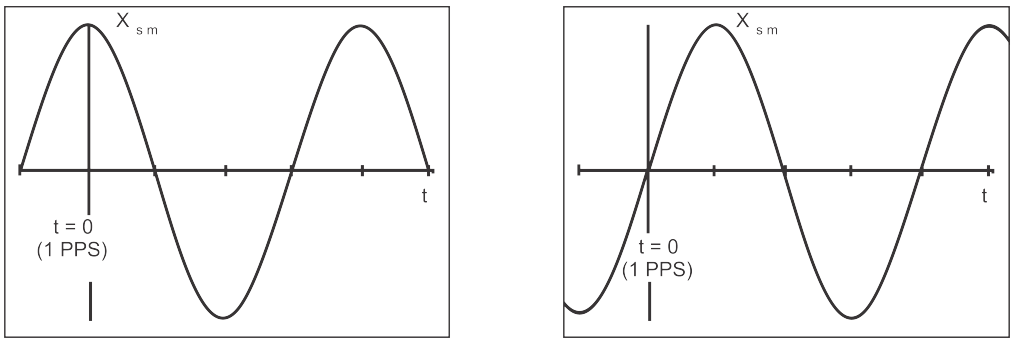
cuya representación fasorial es:

$$\begin{aligned} \vec{X_f} &= (X_{fm}/\sqrt{2})e^{j\phi} \\ &= (X_{fm}/\sqrt{2})(\cos(\phi + j\sin\phi)) \\ &= X_{fr} + jX_{fi} \ , \end{aligned} \tag{1.2}$$

donde la magnitud es el valor eficaz de la onda, y los subíndices r e i significan la parte real e imaginaria del valor complejo. El valor de ϕ depende de la escala de tiempo, particularmente de la ubicación de $t = 0$. Es importante notar que el fasor está definido bajo una frecuencia angular w, razón por la cual todos los fasores deben estar bajo la misma escala de tiempo y frecuencia.

Definición de sincrofasor

Se define como sincrofasor, o fasor sincronizado, al fasor calculado a partir de una señal que es muestreada utilizando una señal de tiempo estándar como referencia para las medidas.
La representación en sincrofasor de la señal $X_f(t)$ de la ecuación (1.1) es el valor de $\vec{X_f}$ en la ecuación (1.2), donde ϕ es el valor instantáneo del ángulo de fase relativo a una función coseno a la frecuencia nominal del sistema y sincronizado con *Coordinated Universal Time* (UTC). Esta sinusoide llamada"onda coseno universal" está sincronizada con la hora UTC de forma que su máximo coincide con el cambio de segundo, visto también como el comienzo de cada nuevo segundo, los cuales están determinados por un flanco ascendente de una señal llamada pulso por segundo (PPS).



(a) $X_s = X_{sm}/\sqrt{2}$, convención para 0 grados.

(b) $X_s = (X_{sm}/\sqrt{2})e^{-j\pi/2}$, convención para -90 grados.

Figura 1.4: Convención para sincrofasores. Las figuras muestran la onda $X_m(t)$.

Así se define esta convención, donde el ángulo de fase es 0 grados cuando el máximo de $X_f(t)$ ocurre junto con un PPS, y -90 grados cuando el cruce por cero con pendiente positiva coincide con un PPS. La figura (1.4) representa lo descrito. Esta sinusoide se describe en la ecuación (1.3):

$$X_s(t) = X_{sm}\cos(w_0 t + \phi) = X_{sm}\cos(2\pi f_0 t + \phi)\ , \tag{1.3}$$

donde f_0 es la frecuencia angular nominal del sistema (50Hz o 60Hz). En el caso general donde la amplitud es una función del tiempo $X_{sm}(t)$ y la frecuencia también lo es $f(t)$, se puede definir la función $g(t) = f(t) - f_0$, donde f_0 es la frecuencia nominal del sistema y g es la diferencia entre la frecuencia actual y la nominal. De esta forma se puede obtener la siguiente ecuación:

$$\begin{aligned} X_s(t) &= X_{sm}(t)\cos(2\pi \int f dt + \phi) \\ &= X_{sm}(t)\cos(2\pi \int (f_0 + g)dt + \phi) \\ &= X_{sm}(t)\cos(2\pi f_0 t + (2\pi \int g dt + \phi)) \end{aligned} \quad . \tag{1.4}$$

La representación en sincrofasor para esta onda es:

$$\vec{X_s}(t) = (X_{sm}(t)/\sqrt{2})e^{j(2\pi \int g dt + \phi)} \ . \tag{1.5}$$

Analizando el caso especial donde $X_{sm}(t) = X_{sm}$ constante y $g = \Delta f$ es un desplazamiento constante contra la frecuencia nominal, $\int g(t)dt = \int \Delta f dt = \Delta f \times t$, por lo que el sincrofasor queda representado como:

$$\vec{X_s}(t) = (X_{sm}/\sqrt{2})e^{j(2\pi \Delta f \times t + \phi)} \ , \tag{1.6}$$

quien va a rotar a una tasa constante Δf, la cual es la diferencia entre la frecuencia actual y la nominal del sistema.

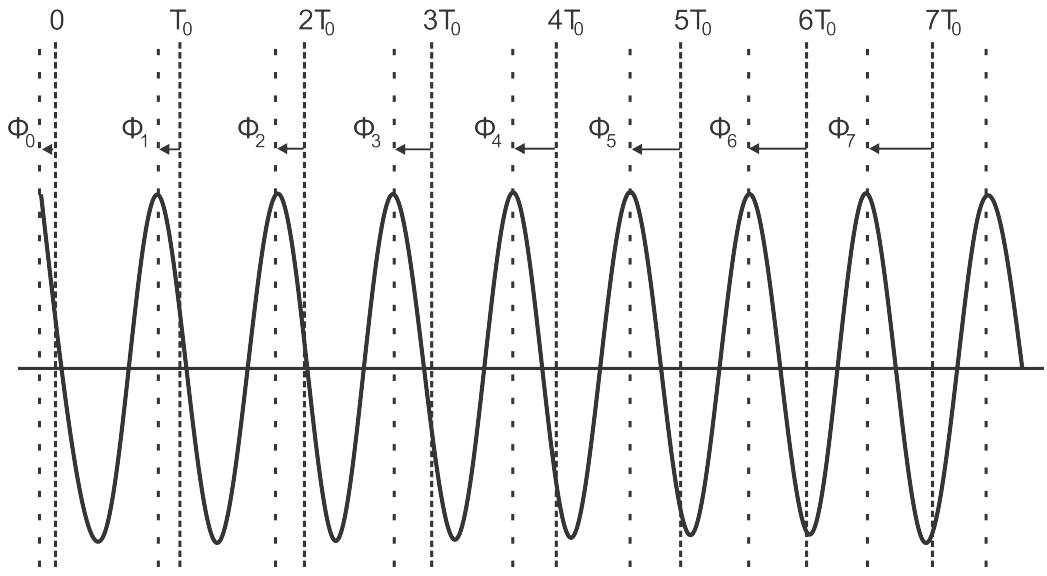


Figura 1.5: Sinusoide de frecuencia $f > f_0$ observada a períodos T_0. La fase ϕ crece uniformemente.

El concepto está ilustrado en la figura (1.5). Considere que una sinusoide a una frecuencia no-nominal está siendo muestreada a intervalos 0, T_0, $2T_0$, $3T_0$, ..., nT_0, ..., donde $T_0 = 1/f_0$ es el período nominal del sistema. Sus correspondientes representaciones fasoriales son $\vec{X_{s0}}$, $\vec{X_{s1}}$, $\vec{X_{s2}}$, $\vec{X_{s3}}$, ..., $\vec{X_{sn}}$. Si la frecuencia f de la sinusoide es distinta de f_0 y a su vez menor que $2f_0$, el fasor observado será de

magnitud constante, pero los ángulos de fase de la secuencia de fasores $\vec{X_{s0}}$, $\vec{X_{s1}}$, $\vec{X_{s2}}$, $\vec{X_{s3}}$, ..., $\vec{X_{sn}}$, ... cambiará de forma uniforme a una tasa $2\pi((f - f_0)T_0)$ como se ilustra.

Algunos detalles en la definición de sincrofasores deben ser resaltados:

- Las medidas de distintos sincrofasores están referidas a una base de tiempos común y relacionadas con una frecuencia común. Es importante para la comparación directa en los ángulos de fase ϕ entre distintos sincrofasores.
- La estimación de la fase ϕ incluye las diferencias en la frecuencia.
- La estimación de sincrofasores incluye los efectos de otras contribuciones como ser oscilaciones y variaciones locales de frecuencia.
- Los sincrofasores son funciones del tiempo, por lo que cambian su valor entre una observación y la siguiente, salvo que la señal se una sinusoide pura a la frecuencia nominal del sistema.
- Para determinar la fase ϕ es necesaria una referencia de reloj precisa que provea el tiempo *UTC*.

Frecuencia y estimación de la variación de la frecuencia

Un PMU debe ser capaz de calcular y reportar la frecuencia y tasa de cambio de frecuencia, o ROCOF por su sigla en inglés de *rate of change of frequency estimation*. Para esta medición se debe utilizar la siguiente definición. Dada una onda sinusoidal:

$$X_s(t) = X_{sm}(t)\cos[\psi(t)] \ , \tag{1.7}$$

la frecuencia se define como:

$$f(t) = \frac{1}{2\pi}\frac{d\psi(t)}{dt} \ , \tag{1.8}$$

y el ROCOF se define como:

$$ROCOF = \frac{df(t)}{dt} \ . \tag{1.9}$$

Los sincrofasores están siempre calculados en relación a la frecuencia nominal del sistema (f_0). Si el argumento del coseno está representado como $\psi(t) = w_0 t + \phi(t) = 2\pi[f_0 t + \phi(t)/2\pi]$, la fórmula para la frecuencia se torna como se muestra a continuación:

$$f(t) = f_0 + d[\phi(t)/2\pi]/dt = f_0 + \Delta f(t) \ , \tag{1.10}$$

donde $\Delta f(t)$ es la desviación en frecuencia contra la nominal y:

$$ROCOF(t) = d^2[\phi(t)/2\pi]/dt^2 = d(\Delta f(t))/dt \ . \tag{1.11}$$

La frecuencia en medidas de fasor puede ser reportada como la actual frecuencia $f(t)$ o como la desviación de la frecuencia nominal $\Delta f(t)$. En condiciones estacionarias, $\Delta f(t)$ puede ser representada como un número escalar Δf.

Evaluación en la medida de sincrofasores

Los valores teóricos de la representación en sincrofasor de una sinusoide, y los valores obtenidos mediante un PMU (ver 1.4.3) pueden incluir diferencias en amplitud y fase. Si bien se pueden especificar estas diferencias separadamente, la diferencia en amplitud y en fase se consideran juntas en una magnitud llamada *error total del vector* o *Total Vector Error* (TVE). TVE es la diferencia entre una muestra perfecta de un sincrofasor y el valor estimado por la unidad en el mismo instante de tiempo. Este valor es normalizado a por unidad según el valor teórico. TVE es definido en la siguiente ecuación:

$$TVE_s(n) == \sqrt{\frac{(\bar{X}_{sr}(n) - X_{sr}(n))^2 + (\bar{X}_{si}(n) - X_{si}(n))^2}{(X_{sr}(n))^2 + (X_{si}(n))^2}} \ , \tag{1.12}$$

donde $\bar{X}_{sr}(n)$ y $\bar{X}_{si}(n)$ son las secuencias de los valores estimados por la unidad bajo prueba, y $X_{sr}(n)$ y $X_{si}(n)$ son los valores teóricos de la señal de entrada en el instante n. Los valores de $X_{sr}(n)$ y $X_{si}(n)$ pueden ser determinados de forma muy precisa bajo ciertas condiciones bien definidas, como ser frecuencia o fase constante.
Las medidas de sincrofasores deben ser evaluadas utilizando el criterio de TVE de la ecuación (1.12). Los valores requeridos por la norma para TVE son de 1 % para una situación estática donde se mantienen constantes la amplitud X_{sm}, la frecuencia y ROCOF. Este 1 % también se aplica para una entrada de tipo rampa en frecuencia a una tasa de variación de 1Hz/s. Para un cambio de tipo escalón no se aplica exigir un máximo TVE, pero si se exige que el sobretiro en la medición no supere en 10 % al valor del escalón ingresado.

Evaluación en la medida de frecuencia y ROCOF

Las medidas de frecuencia y ROCOF deben ser evaluadas utilizando las siguientes definiciones (Ecuaciones 1.13 y 1.14). Con este criterio, los errores de frecuencia y ROCOF son el valor absoluto de la diferencia entre el valor teórico y el valor estimado, dados en Hz y Hz/s.

Error en medida de frecuencia - *Frecuency Error* (FE):

$$\text{FE} == |f_{\text{teórico}} - f_{\text{medida}}| = |\Delta f_{\text{teórico}} - \Delta f_{\text{medida}}| \ . \tag{1.13}$$

Error en medida de ROCOF - *Rate of Change of Frecuency Error* (RFE):

$$\text{RFE} == |(df/dt)_{\text{teórico}} - (df/dt)_{\text{medida}}| \ . \tag{1.14}$$

Los valores medidos y los valores teóricos son del mismo instante de tiempo, el cual está dado por la marca de tiempo insertada por el PMU en los valores

medidos. Los valores máximos requeridos por la norma para estado estacionario son FE $= 0,005$Hz y RFE $= 0,01$Hz/s.

Tiempo de respuesta en la medición y tiempo de retardo

El tiempo de respuesta es el tiempo de transición de la medida ante un escalón entre dos estados estacionarios aplicado en la entrada. Este tiempo de respuesta se determina como la diferencia de tiempos entre que la medida abandona cierto intervalo de precisión respecto del estado pre escalón y el tiempo en que entra y se mantiene dentro de cierto intervalo de precisión en el estado estacionario post escalón. Esto debe ser medido aplicando un escalón positivo o negativo tanto en fase como en módulo a la entrada del PMU, y esta debe ser la única magnitud que cambie durante la prueba. Los límites de precisión son TVE, FE y RFE, para el fasor, frecuencia y *Rate Of Change Of Frecuency* (ROCOF) respectivamente, y estos límites están determinados por la norma.

El tiempo de retardo en la muestra está definido como el intervalo de tiempo comprendido entre el instante en que es aplicado el escalón en la entrada el PMU y el tiempo en que la medida alcanza un valor que esté en la mitad entre el valor final y el valor inicial en los regímenes estacionarios. Estos tiempos se miden referidos a la hora UTC. Esta medida debe ser determinada aplicando un escalón positivo o negativo en fase o magnitud a la señal de entrada del PMU, donde esta señal de entrada debe estar en estado estacionario antes y después de aplicado el escalón, mientras todas los demás parámetros de la señal deben permanecer constantes. El propósito de evaluar el retardo en la medición es verificar que el tiempo marcado en la medida de sincrofasor ha sido debidamente compensado. Es de esperar que los filtros inserten retardo de grupo, por lo cual esta compensación debería llevar a cero este retardo.

Latencia en el reporte de medidas

Se le dice latencia en el reporte de medidas al tiempo de retardo entre que el evento ocurre en el sistema de potencia y el evento es reportado en los datos de salida. Este retardo incluye varios factores, como por ejemplo la ventana con la que se procesan las muestras para realizar la medida, el método empleado por la PMU para la estimación, los filtros, el tiempo de procesado de la PMU, y cuando ocurre el evento respecto al intervalo de reporte. Este estándar define la latencia de reporte del PMU como el máximo intervalo entre el instante del reporte del dato indicado en la *time stamp* del dato, y que el dato se torna accesible a la salida del PMU (identificado por la primera transición del primer bit del mensaje de salida en el punto de interfaz de la comunicación).
Si el PMU reporta 50 muestras por segundo, la máxima latencia permitida para funcionalidades de protección es de 40ms, y para funcionalidades de medida es de 100 ms.

Errores operacionales y de medida

El PMU debe asignar una bandera a cada medida para indicar la presencia o no de problemas internos encontrados durante el proceso de medición. Esta bandera debe incluir errores detectables por el PMU como por ejemplo errores en la conversión A/D, *overflow* en memoria o en cálculos, y cualquier otra condición que cause un error en la medición.

Reporte de medidas

Las medidas de sincrofasor, frecuencia y ROCOF deben ser realizadas de tal forma que puedan ser reportadas a una tasa constante F_s, el cual es un número entero cuando la tasa es mayor a una muestra por segundo, o un número entero de segundos entre medidas, cuando la tasa de medida es igual o más lenta que una por segundo. Todas las medidas deben ser realizadas y reportadas bajo la misma tasa. Los tiempos de reporte deben estar adecuadamente espaciados de tal forma que el intervalo entre reportes sea constante para todos.

Tasas de reporte

El PMU debe soportar tasas de reporte como sub-múltiplos de la frecuencia nominal del sistema. Las tasas requeridas para 50 Hz y 60 Hz se enumeran en la siguiente tabla:

Frecuencia nominal del sistema (Hz)	Tasa de reporte (F_s datos por segundo)
50	10, 25, 50
60	10, 12, 15, 20, 30, 60

La tasa actual a ser utilizada debe ser seleccionable por el usuario y es admitido y fomentado por la norma tener soporte para otras tasas de reporte más altas, tales como $100/s$ o $120/s$.

1.4.3. PMU

Definición de PMU

Del estándar C37.118.1:

> "Unidad medidora de fase (*phase measurement unit*, o PMU): En este estándar, es un dispositivo que genera estimaciones de fasores sincronizados, frecuencia y ROCOF, estimados mediante señales de voltaje y/o corriente y una señal sincronizada en el tiempo. Nótese que el mismo dispositivo puede realizar otras funciones y tener otro nombre funcional, por ejemplo un dispositivo que registre la forma de onda en el sistema de potencia, llamado DFR (*digital fault recorder*)."

Una PMU debe calcular los sincrofasores y reportarlos a una tasa constante, la cual debe ser seleccionada por el usuario entre las soportadas por la unidad. El sistema también puede incluir palabras digitales de estatus, como estatus del interruptor para los casos donde el PMU incluye un relé de protección numérica, umbrales de alarma o información de temperatura.

A diferencia de los métodos clásicos de recolección de datos, los PMU permiten medir y comparar las fases de las distintas barras en un sistema de potencia aunque estos puntos estén distantes geográficamente.
Algunas de las funciones típicas que cumplen las PMUs en las redes de potencia son:

- Medir la frecuencia de la red.
- Medir la variación de la frecuencia de la red.
- Medir desfasajes

Estas medidas son utilizadas directamente en algoritmos que monitorean la estabilidad de la red y rutinas orientadas a evitar el funcionamiento en isla o minimizar el impacto de ciertas faltas [16].
Adicionalmente los PMU aportan información muy valiosa en lo referente a estimación de estado, ya que miden directamente el estado de la red, es decir módulo y fase de las tensiones en donde se instalan. Esta tecnología permitiría en teoría conocer el estado de la red en tiempo real disponiendo de una PMU por cada barra del sistema. Esto no es posible aún debido al elevado coste de los equipos y de la infraestructura necesaria, lo que supondría realizar una gran inversión en la red. Por este motivo en las redes se instalan un número limitado de PMU lo que provoca que sigan siendo necesarios algoritmos de estimación de estado.
Esto motivó a la investigación de nuevos algoritmos capaces de fusionar los datos provenientes del SCADA y de los PMU para mejorar la estimación final. Esto involucra enfrentar problemas básicos como:

- El SCADA se actualiza cada algunos segundos, pero proporciona suficiente información para obtener una estimación de la red completa.
- Los PMU se actualizan cada algunos milisegundos, pero usualmente no se logra una cobertura total de la red.

Esto motiva a que los nuevos algoritmos sean capaces de arrojar una estimación de estado en lapsos muy cortos y que a su vez sean capaces de contemplar la información del SCADA.
Los algoritmos existentes que realizan esta fusión son variados, y no todos son capaces de arrojar una estimación en lapsos cortos. Más adelante se discutirán algunos de ellos.

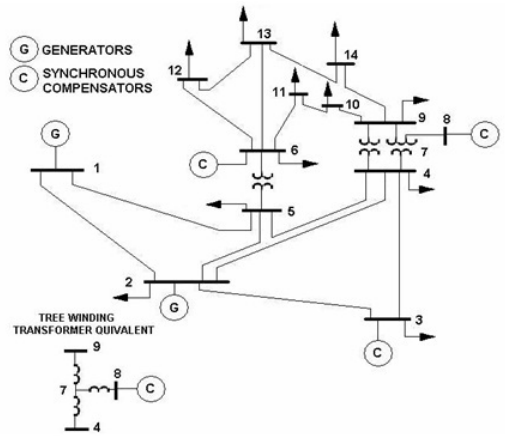


Figura 1.6: IEEE 14 Bus System

1.4.4. Introducción a la estimación de estado clásica

Para realizar una introducción a la estimación de estado clásica, en las siguientes líneas se realizará un repaso de todos los conceptos básicos detrás de las redes eléctricas de potencia. Esto incluye definir: como se nombran las partes de una red de potencia, como se modelan las líneas de transmisión, la topología de la red, las ecuaciones de vínculo entre estado y medidas, las medidas proporcionadas por el SCADA, y que herramientas matemáticas se utilizan para la resolución del sistema de ecuaciones (Mínimos cuadrados ponderados, etc.).

Nomenclatura de los elementos de la red

En la figura 1.6 se muestra un ejemplo de esquema unifilar típico con que se representan las redes de potencia, cada una de las barras negras horizontales son llamadas “barras” o “nodos” del sistema. Estas están interconectadas por líneas de transmisión o transformadores. En dicho esquema encontraremos otros elementos como pueden ser generadores, compensadores de reactiva y también cargas del sistema.

Si se conoce las características de las líneas de transmisión, transformadores, generadores y compensadoras de reactiva, y además se conoce el estado de las conexiones en la red, entonces se dice que hay un conocimiento de la topología de la red. Esto matemáticamente se refleja en la llamada *Matriz de Admitancia Nodal* o Ybus.

Modelo de las líneas de transmisión

Las líneas de transmisión se dividen en 3 modelos dependiendo del largo de las mismas: líneas cortas, medias o largas. El modelo básico para líneas de media y

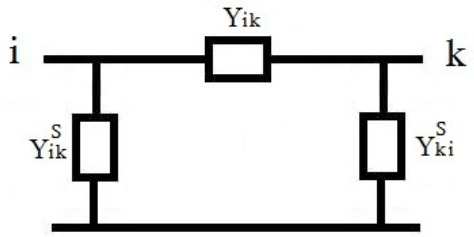


Figura 1.7: Modelo Pi para línea i-k

larga distancia es el modelo Pi que se visualiza en la figura 1.7,

donde los subíndices i y k son los números de barras. Los elementos Y_{ik}^S e Y_{ki}^S son las admitancias shunt a tierra, y el elemento Y_{ik} es la admitancia serie de la línea.

Partiendo de la siguiente lista de datos se pueden calcular los parámetros del modelo pi de la línea:

- L: largo de la línea (km)
- r: resistencia unitaria (Ω/km)
- l: auto inductancia unitaria (H/km)
- g: conductancia unitaria (Ω^{-1}/km)
- c: capacitancia unitaria (F/km)
- z: impedancia serie unitaria (Ω/km) : $z = r + jlw$
- ϕ: admitancia shunt unitaria (Ω^{-1}/km) : $\phi = g + jcw$
- Z: impedancia serie total (Ω) : $z.L$
- C: admitancia shunt total (Ω^{-1}) : $\phi.L$
- θ: constante total: (adimensionada) $\theta = (Z.C)^{1/2}$

Los parámetros del modelo pi se calculan como se indica en las siguientes expresiones [7]:

$$Línea \quad larga: \quad \begin{cases} \vec{Y_{ik}} = (\vec{Z}.\frac{sh(\theta)}{\theta})^{-1} \\ \vec{Y_{ik}^S} = \vec{Y_{ki}^S} = C.\frac{ch(\theta)-1}{\theta.sh(\theta)} \end{cases} \tag{1.15}$$

$$Línea \quad media: \quad \begin{cases} \vec{Y_{ik}} = \vec{Z}^{-1} \\ \vec{Y_{ik}^S} = \vec{Y_{ki}^S} = \frac{C}{2} \end{cases} \tag{1.16}$$

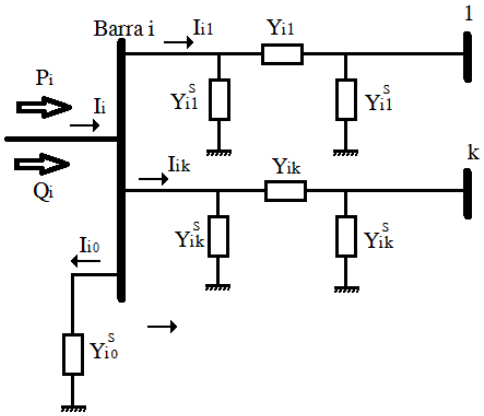


Figura 1.8: Esquema de red con modelos Pi

Matriz de admitancia nodal o Ybus

A partir de los datos de las líneas de transmisión, transformadores y compensadores *shunt* se calcula la matriz de admitancia nodal como se explica a continuación.

En la barra i se cumple que las corrientes en pu deben ser:

$$\vec{I_i} = \vec{I_{i0}} + \vec{I_{i1}} + \vec{I_{i2}} + .. + \vec{I_{in}} \ , \tag{1.17}$$

$$\vec{I_i} = \vec{V_i} \left\{ \vec{Y_{i0}^S} + \sum_{k=1,k\neq i}^{n} \vec{Y_{ik}^S} \right\} + (\vec{V_i} - \vec{V_1})\vec{Y_{i1}} + .. + (\vec{V_i} - \vec{V_n})\vec{Y_{in}} \ , \tag{1.18}$$

$$\vec{I_i} = \left\{ \vec{Y_{i0}^S} + \left\{ \sum_{k=1,k\neq i}^{n} \vec{Y_{ik}^S} \right\} + \vec{Yi1} + .. + \vec{Yin} \right\} \vec{V_i} - \vec{Y_{i1}}\vec{V_1} - .. - \vec{Y_{in}}\vec{V_n} \ . \tag{1.19}$$

De la última igualdad se desprende la forma matricial de escribir las corrientes en función de las tensiones:

$$\begin{pmatrix} \vec{I_1} \\ \vdots \\ \vec{I_n} \end{pmatrix} = \begin{pmatrix} m_{11} & \cdots & m_{1n} \\ \vdots & \ddots & \vdots \\ m_{n1} & \cdots & m_{nn} \end{pmatrix} \begin{pmatrix} \vec{V_1} \\ \vdots \\ \vec{V_n} \end{pmatrix} \ , \tag{1.20}$$

donde la entrada m_{ij} se vincula con los parámetros de la red como sigue:

$$m_{ij} = -\vec{Y_{ij}} \qquad i \neq j \ , \tag{1.21}$$

$$m_{ii} = \sum_{k=1}^{n} \vec{Y_{ik}} + \sum_{k=1}^{n} \vec{Y_{ik}^S} + \vec{Y_{i0}^S} \ . \tag{1.22}$$

Las características de la matriz de admitancia nodal son las siguientes:

- Es una matriz simétrica.

- Tamaño nxn compleja, siendo n el número de barras.
- Se calcula una única vez si la topología de la red no cambia.
- Matriz base para la construcción de los sistemas de ecuaciones que son utilizados en estimación de estado y flujo de carga.

Estimación de estado clásica

El objetivo de la estimación de estado consiste en hallar los módulos y fases de las tensiones en cada barra del sistema. A diferencia de lo que ocurre en flujo de carga, en la estimación de estado se cuenta con más ecuaciones que incógnitas. Esta redundancia tiene como beneficio que puedan existir errores en la mayoría de las medidas, pero igual así se pueda estimar el estado del sistema con buena precisión. Esto se cumple en la hipótesis de que cada medida tenga un error que responda a ruido blanco gaussiano y de media nula. En el caso de que no se cumplan estas características, existen herramientas estadísticas para detectar estas medidas erróneas o *Bad Data* en la jerga internacional. En la estimación de estado clásica la única información disponible proviene del SCADA (P_i, Q_i, $|V_i|$), donde la variable principal *vector de estado del sistema* se compone de los siguientes elementos:

$$X = [|V_1|, |V_2|, .., |V_n|, \theta_1, \theta_2, .., \theta_n] \ . \tag{1.23}$$

Dicho vector contiene los módulos en *pu* de cada barra del sistema y las fases de dichas tensiones con respecto a una barra de referencia en radianes o grados. La relación entre las medidas P_i, Q_i, V_i del SCADA y el vector de estado se detalla a continuación. Para ello recurrimos a las ecuaciones de flujo de carga:

$$P_i = |V_i| \sum_{j=1}^{N} |V_j| \left\{ g_{ij} cos(\theta(i) - \theta(j)) + b_{ij} sin(\theta(i) - \theta(j)) \right\} \ , \tag{1.24}$$

$$Q_i = |V_i| \sum_{j=1}^{N} |V_j| \left\{ g_{ij} sen(\theta(i) - \theta(j)) - b_{ij} cos(\theta(i) - \theta(j)) \right\} \ , \tag{1.25}$$

$$g_{ij} = Re\{m_{ij}\} \ , \tag{1.26}$$

$$b_{ij} = Im\{m_{ij}\} \ . \tag{1.27}$$

Notar que se utiliza la información de la matriz Ybus para la construcción de dichas ecuaciones. Por otra parte se define el vector Z como sigue a continuación:

$$Z_{SCADA} = h(X) + \epsilon \ , \tag{1.28}$$

que es un vector $3m \times 1$, donde $3m$ es la cantidad de medidas.

Se puede observar en la figura 1.9 el esquema del sistema. Se dice que la solución del sistema X_{sol} es aquella que minimiza la suma cuadrática de los errores, llamado *weighted least squares* (WLS) [1] que expresado matemáticamente se escribe como:

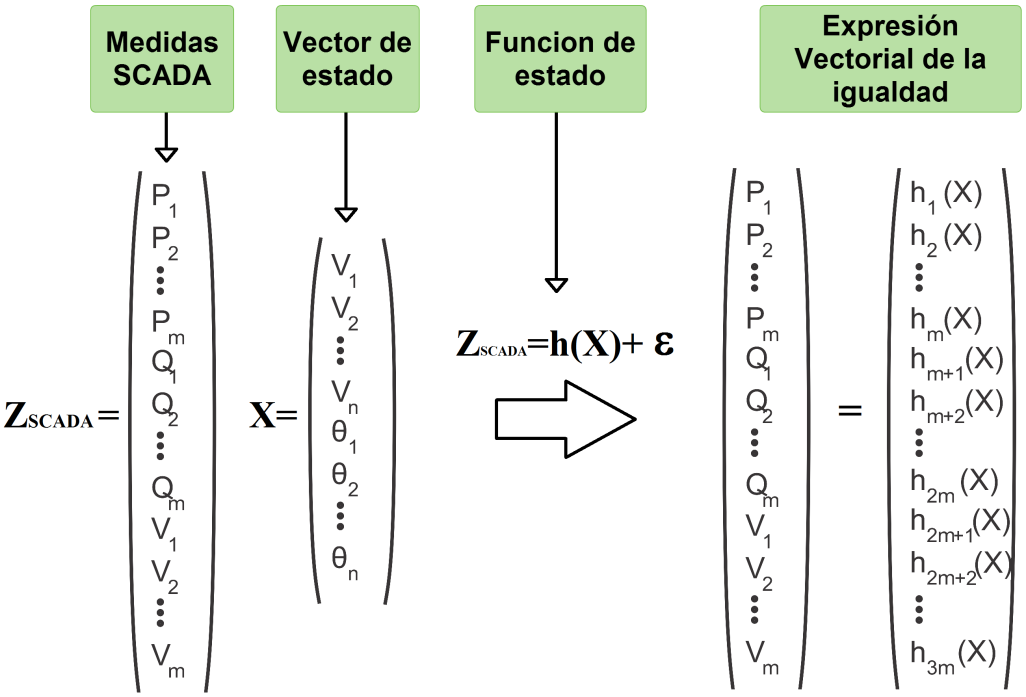


Figura 1.9: Sistema de estimación

$$min_X \quad \{J(X)\} = min_X \quad \{\sum_1^m e_i^2\} \ , \tag{1.29}$$

donde e_i es el elemento i-esimo del vector ϵ

$$e_i = [Z_{SCADA} - h(X)]_i \ . \tag{1.30}$$

El método se puede mejorar considerablemente si se pondera la suma según la confianza de la medida. Es decir si una medida tiene gran incertidumbre, el error e_i deberá tener menor influencia en la minimización de $J(X)$. Esto expresado en una ecuación matemática se escribe como:

$$J(X) = \sum_1^m (\frac{e_i}{\sigma_i})^2 \ , \tag{1.31}$$

donde σ_i es la incertidumbre de dicha medida. La función $J(X)$ se puede escribir matricialmente como:

$$J(X) = [Z_{SCADA} - h(X)]^T R^{-1} [Z_{SCADA} - h(X)] \ , \tag{1.32}$$

donde la matriz R es:

$$R^{-1} = diag[1/\sigma_i^2] \ , \tag{1.33}$$

Para minimizar $J(X)$ en función de X hacemos:

$$\frac{\partial J(X)}{\partial X} = P^T(X)R^{-1}[Z_{SCADA} - h(X)] = 0 \ , \tag{1.34}$$

donde P es la matriz Jacobiana de la función h respecto al vector X:

$$P(X) = \nabla h(X_1, .., X_n) \ . \tag{1.35}$$

Por último para encontrar la solución a la ecuación (1.34) se utiliza el método iterativo Gauss-Newton que se detalla a continuación [1]:

Se define la matriz de ganancia del sistema para el instante de iteración k como:

$$G(X^k) = P(X^k)^T R^{-1} P(X^k) \ . \tag{1.36}$$

Se tiene el vector de incremento o diferencias entre los pasos del método iterativo como:

$$\Delta X^k = X^{k+1} - X^k \ . \tag{1.37}$$

Por último se debe resolver el sistema lineal que sigue a continuación:

$$G(X^k)\Delta X^k = P^T(X^k)R^{-1}[Z - h(X^k)] \ , \tag{1.38}$$

$$X^{k+1} = \Delta X^k + X^k \ . \tag{1.39}$$

El método iterativo termina cuando el módulo del vector ΔX^k es menor a cierto criterio pre-definido.

1.4.5. Introducción a la estimación de estado utilizando PMUs

Con la aparición de los PMU, surgió el interés por aprovechar esa información extra, y se han elaborado varios métodos para involucrarla en la estimación de estados. Estos métodos los podemos separar en 2 grandes grupos:

- -Implementación directa en la estimación de estados.
- -Implementación no invasiva, desacoplada de la estimación de estados clásica.

En la implementación directa, las medidas provenientes de los PMU son tratadas de igual forma que las medidas tradicionales, pero considerando su mayor precisión para la estimación. Esto logra mejorar el desempeño de los estimadores clásicos, logrando errores más pequeños y una mayor robustez. Para ello es necesario cambiar el software dedicado a la estimación de estados, de forma de incorporar también estas nuevas medidas. La implementación no invasiva no requiere cambiar los algoritmos ya existentes, y es muy sencilla de implementar ya que se trata de estimadores lineales usando la información de los PMU. Como los PMU no permiten una observabilidad completa de la red, se utiliza la estimación de estados

clásica y luego se la mejora con algoritmos lineales [22].

Existen también estimadores alternativos a estos dos grandes grupos, donde se aprovecha la gran tasa de adquisición de datos que tienen los PMU. En los estimadores mencionados anteriormente se descartan los datos de PMUs que llegan entre 2 estimaciones clásicas, ya que se utilizan solo una vez por cada estimación clásica realizada. Al descartar todas estas medidas se pierde una información muy valiosa, que nos puede dar una idea del comportamiento dinámico del sistema. A continuación se estudiarán ejemplos de distintos algoritmos que utilizan medidas de PMU para mejorar la estimación de estados, en los cuales podremos entender más en profundidad la clasificación recién descrita.

Capítulo 2

Estudio y evaluación de los principales artículos

2.1. Introducción

A partir de los objetivos y el alcance del proyecto se inició una búsqueda bibliográfica que se puede dividir en 3 grandes etapas. La primera consistió en estudiar la estimación de estado. Esto involucro el aprendizaje de ciertas herramientas matemáticas y su aplicación en la estimación.
La segunda etapa consistió en una búsqueda orientada a algoritmos que incorporasen PMUs. Debido a que se encontró mucha información fue necesario realizar una clasificación de los artículos con el fin de organizar los algoritmos. Al mismo tiempo se fue visualizando el estado del arte en esta temática.
Por último se hizo una profundización en los 4 principales algoritmos encontrados para dar una presentación en un seminario donde estuvieron presentes integrantes del Grupo de Control y Estabilidad en Sistemas Eléctricos de Potencia. El resultado de dicho seminario fue un replanteo en el plan de proyecto, donde se modificó el plan inicial de implementar 2 o más algoritmos en Matlab a implementar solo uno, y además se definió cuál de ellos se iba a implementar.

Como ya se mencionó no es posible asociar a cada algoritmo un único artículo, debido a que algunos algoritmos son enriquecidos y puestos a prueba en varios artículos. Los algoritmos más destacados se dividen en las siguientes categorías:

1. Fusión directa entre datos del SCADA y de los PMUs para formar un único sistema de ecuaciones no lineales [25] [26].

2. Procesado en dos etapas, donde los datos de las PMU se incorporan en un segundo paso en forma lineal [25] [22] [26].

3. Resolución de los sistemas de ecuaciones no lineales a través de Rotaciones de Givens e información a priori. [2] [3] [21]

4. División de la red de potencia en sub-sistemas, aplicando a cada sub-zona algoritmos del tipo 1 o 2, y a través de un algoritmo de fusión hallar la estimación final [6] [24].

5. **Algoritmo adaptativo**. El mismo se compone de dos bucles, siendo uno lineal y el otro no. La novedad está en como se gestionan dichos bucles para ser eficiente computacionalmente [4].

2.2. Presentación e introducción a los principales algoritmos

En el presente capítulo se analizarán los 5 tipos de algoritmos mencionados en la sección anterior. En el caso del **algoritmo adaptativo** se realizara una profundización mayor en el capítulo siguiente. Antes de detallar cada uno de los algoritmos se realizara un resumen de términos y aspectos comunes a todos ellos.

El vector de estado y los vectores de medidas se componen como sigue a continuación:

$$X_{polar} = [V_1, .., V_n, \theta_1, .., \theta_n] \ , \tag{2.1}$$

$$X_{cartesianas} = [Re(\vec{V_1}), .., Re(\vec{V_n}), Im(\vec{V_1}), .., Im(\vec{V_n})] \ , \tag{2.2}$$

$$Z_{SCADA} = [.., P_i, Q_i, V_i, .., P_j, Q_j, V_j, ..] \ , \tag{2.3}$$

$$Z_{PMU} = [V_i, .., V_j, \theta_i, .., \theta_j, I_{ij}, .., I_{ik}, .., \theta_{Iij}, .., \theta_{Iik}, ..] \ , \tag{2.4}$$

$$Z_{PMU2} = [Re(\vec{V_i}), Im(\vec{V_i}), .., Re(\vec{V_j}), Im(\vec{V_j}), .., Re(\vec{I_{ij}}), Im(\vec{I_{ij}}), .., Re(\vec{I_{ik}}), Im(\vec{I_{ik}})] \ , \tag{2.5}$$

$$Z_{PMU3} = [.., \vec{V_i}, .., \vec{V_n}, \vec{I_{ij}}, .., \vec{I_{ik}}] \ . \tag{2.6}$$

Se observa en (2.1) un vector de variables de estado escrito en forma polar, y en (2.2) se observa las mismas magnitudes escritas en coordenadas rectangulares o cartesianas. En (2.3) se encuentra un vector de medidas proporcionado por SCADA, y por último se encuentran 3 representaciones para las medidas provenientes de los PMUs. Estos últimos 3 vectores tienen elementos asociados solamente a barras y líneas donde están midiendo los PMUs. Es importante observar que como las PMU proporcionan los fasores de tensión y corriente, estos números complejos se pueden expresar en sus diversas representaciones como lo indican las expresiones (2.4) (2.5) (2.6).

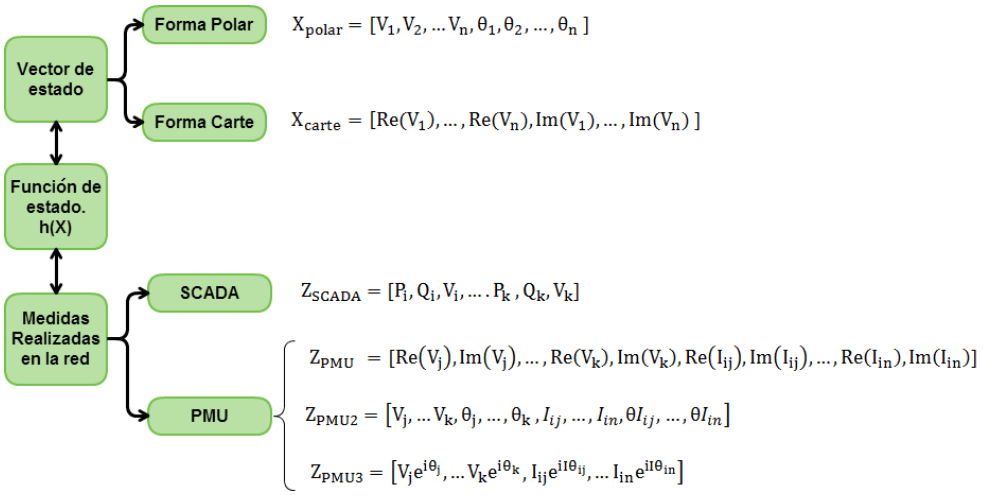


Figura 2.1: Esquema de medidas.

Por otro lado se tiene que la relación entre las medidas del sistema Z y las variables de estado X está dada por la función h:

$$Z = h(X) + \epsilon \ . \tag{2.7}$$

En la figura 2.1 se presenta un esquema que resume los conceptos antes mencionados.

2.2.1. Fusión directa entre SCADA y PMU

La novedad en este algoritmo [25] es incorporar a la función $h(X)$ medidas que provienen de los PMUs. Para esto se debe contar con las ecuaciones de vínculo entre el vector de estado y las variables que miden las PMUs.
Para mantener la estructura del algoritmo de estimación de estados clásica intacta, el vector de estado se toma en su forma polar X_{polar}, que es la forma que simplifica las ecuaciones de vínculo para las medidas de SCADA:

$$Z_{PMU} = [V_i, .., V_j, \theta_i, .., \theta_j, I_{ij}, .., I_{ik}, .., \theta_{Iij}, .., \theta_{Iik}, ..] \ , \tag{2.8}$$

$$Z_{PMU} = h(X) \ , \tag{2.9}$$

donde h(X) contiene funciones identidad para las medidas de tensión de PMU, ya que tanto el vector de estado como el vector de medida están en forma polar.
En cambio las corrientes se relacionan con el vector de estado con la siguiente expresión:

$$Re(\vec{I_i}) = V_i cos(\theta_i) g_{ii} - V_i sin(\theta_i) b_{ii} + \sum_{j=1, j\neq i}^{n} \{V_j cos(\theta_j) g_{ij} - V_j sin(\theta_j) b_{ij}\} \ , \tag{2.10}$$

$$Im(\vec{I_i}) = V_i cos(\theta_i) b_{ii} + V_i sin(\theta_i) g_{ii} + \sum_{j=1, j\neq i}^{n} \{V_j cos(\theta_j) b_{ij} - V_j sin(\theta_j) g_{ij}\} \ . \quad (2.11)$$

Es visible el hecho de que al agregar las ecuaciones (2.10) y (2.11) al sistema $h(X)$, estando el vector de estados en su forma polar, se deben realizar muchas más operaciones matemáticas y la matriz Jacobiana se vuelve más compleja. El costo computacional de este tipo de algoritmos es más alto comparado con otros, por ejemplo en [25] se realiza la comparación entre este método y el descripto a continuación, y se concluye que los resultados son idénticos y los tiempos son de hasta casi 2 segundos de diferencia a favor del otro algoritmo. Por este motivo el algoritmo es rechazado, pero tiene un aporte significativo a la hora de comprender como es la construcción de la función $h(X)$.
Otra forma de incorporar las medidas de corriente a la estimación de estado es utilizando la información de la topología de la red, la tensión en bornes de donde está ubicada la PMU y calcular la tensión en la otra punta de la línea. En la figura 2.2 se observa el esquema de la solución usando esta alternativa, la cual utiliza un modelo Pi de la línea como muestra la figura 2.3 donde se indica la tensión y corriente medida por PMU, y la tensión a calcular.

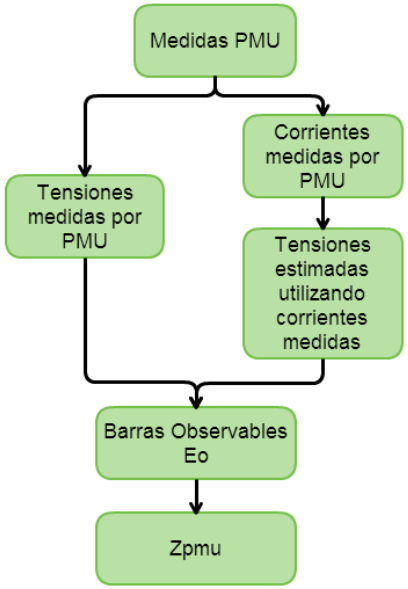


Figura 2.2: Componentes del vector Z_{PMU}

Una solución similar a esta fue la que se adoptó en el proyecto. En resumen las medidas que provienen de los PMU se unifican en un conjunto de tensiones, unas directamente medidas, y otras calculadas.

2.2.2. Procesado en doble etapa. Incorporación de Z_{PMU} de forma lineal

Este algoritmo cuenta con dos etapas [22]. La primera consiste en una estimación de estados tradicional, donde se calculan las tensiones de todas las barras (modulo y fase) utilizando solamente las medidas de SCADA. Esto puede ser realizado por ejemplo con la estimación clásica ya vista, recordar que el vector de estado es expresado en forma polar. Aquí se genera una estimación del vector de estados que denotamos como X_{SCADA}. Para poder utilizar esta estimación como entrada para la segunda etapa, se crea el vector de *pseudomedidas* Z_{pseudo}:

$$Z_{pseudo} = X_{SCADA-cartesianas} \ . \tag{2.12}$$

En la segunda etapa se realiza una estimación lineal. Para esto se toma un modelo lineal del sistema:

$$Z = H.X + \epsilon \ , \tag{2.13}$$

$$Z = \begin{bmatrix} Z_{pseudo} \\ Z_{\text{PMU}} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} V_{\text{i real}} \\ V_{\text{i imag}} \end{bmatrix}_{pseudo} \\ Z_{\text{PMU}} \end{bmatrix} \ . \tag{2.14}$$

Z es el vector de medidas, que contiene las tensiones y corrientes medidas por los PMU, y también las tensiones estimadas por el SCADA en el paso previo, también llamadas *pseudomedidas*, todas ellas escritas en forma cartesiana. El vector de estado del sistema también se escribe en forma cartesiana, esto significa que durante el proceso se debe convertir el vector de estado de forma polar a cartesiana. Las corrientes medidas por las PMU expresadas en forma cartesianas pueden ser vinculadas en función de las tensiones en forma lineal:

$$I_{\text{i real}} = V_{\text{i real}} G_{ii} - V_{\text{i imag}} B_{ii} + \sum_{j=1, j\neq i}^{n} \left(V_{\text{j real}} G_{ij} - V_{\text{j imag}} B_{ij} \right) \ , \tag{2.15}$$

$$I_{\text{i imag}} = V_{\text{i real}} B_{ii} + V_{\text{i imag}} G_{ii} + \sum_{j=1, j\neq i}^{n} \left(V_{\text{j real}} B_{ij} - V_{\text{j imag}} G_{ij} \right) \ . \tag{2.16}$$

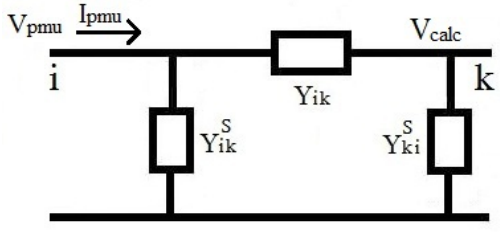


Figura 2.3: Modelo pi para calcular V_k

Por lo que podemos escribir la ecuación de la estimación como sigue:

$$\begin{bmatrix} \begin{bmatrix} V_{\text{i real}} \\ V_{\text{i imag}} \end{bmatrix}_{\text{estimados}} \\ \begin{bmatrix} V_{\text{i real}} \\ V_{\text{i imag}} \\ I_{\text{i real}} \\ I_{\text{i imag}} \end{bmatrix}_{\text{pmu}} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \\ \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \\ G_{ij} & -Bij \\ B_{ij} & G_{ij} \end{bmatrix} \end{bmatrix} \begin{bmatrix} V_{\text{i real}} \\ V_{\text{i imag}} \end{bmatrix} + [\epsilon] \ . \tag{2.17}$$

Donde I denota la matriz identidad, las matrices M_{11} hasta M_{22} son matrices que contienen 1 en los puntos que corresponda según qué medidas se tengan de las PMU, y las matrices G y B contienen elementos de la matriz Ybus.

H es entonces:

$$H = \begin{bmatrix} \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \\ \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \\ G_{ij} & -Bij \\ B_{ij} & G_{ij} \end{bmatrix} \end{bmatrix} . \tag{2.18}$$

Finalmente el sistema tiene una solución directa por mínimos cuadrados como sigue:

$$X = ([H]^T[R]^{-1}[H])^{-1}[H]^T R^{-1}[Z]. \tag{2.19}$$

2.2.3. Rotaciones de Givens e información a priori

El método consiste en dos pasos, estos son:

1. SCADA tradicional con información a priori.

2. Post-procesado con información de PMU.

La información a priori consiste en información adicional que se tiene sobre las variables de estado del sistema, previa a la ejecución de realizar la estimación de estado. A estos datos se le asigna cierta confianza mediante una matriz de covarianza. Los valores a priori contribuyen a la estimación de igual forma que lo hacen las mediciones, aunque por lo general se les asigna menos exactitud que a las medidas. Las características principales del método son:

- Utilizar rotaciones de Givens para solucionar el problema de mínimos cuadrados (robustez numérica).

- Permite incorporar la información a priori sin gastos computacionales extras.

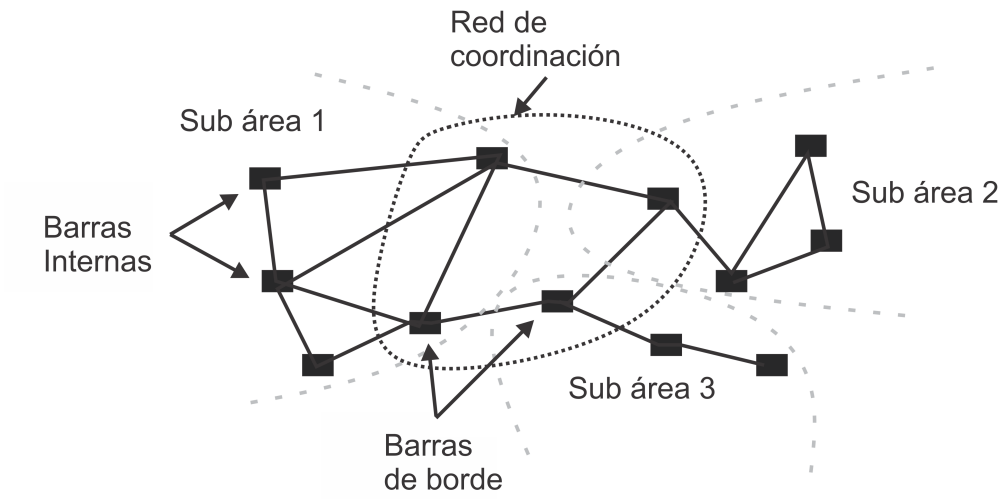


Figura 2.4: División del sistema en sub-sistemas

2.2.4. División de la red en sub-sistemas

El método basado en sub-sistemas [24] divide la red en varias zonas y realiza una estimación local en cada una de las zonas, para luego mediante un post-procesado obtener la estimación final.

A grandes rasgos el algoritmo se compone de los siguientes dos pasos:

1. Realizar la estimación en cada sub-sistema mediante la utilización del SCADA y los PMUs, igual a lo tratado en 2.2.1.

2. Realizar una estimación final (llamada nivel de coordinación) donde se unifican las estimaciones de cada sub-sistema, preferentemente se desea que sea lineal.

Una vez definidas las zonas, las barras se clasifican como:

- Barras internas
- Barras externas
- Barras de contorno (Barras que tienen conectado al menos una barra externa)

La figura (2.4) ilustra lo descrito. Las líneas que conectan barras de dos zonas distintas de definen como *Tie-lines*. La división de la red es fundamental para una buena estimación y coordinación, donde los principales criterios para la elección de zonas para los sub-sistemas están dados por las siguientes pautas:

- No hay solapamiento entre zonas. Es decir, no se comparten barras.
- Al menos existe un PMU por zona, y la misma debe estar en una barra de contorno.

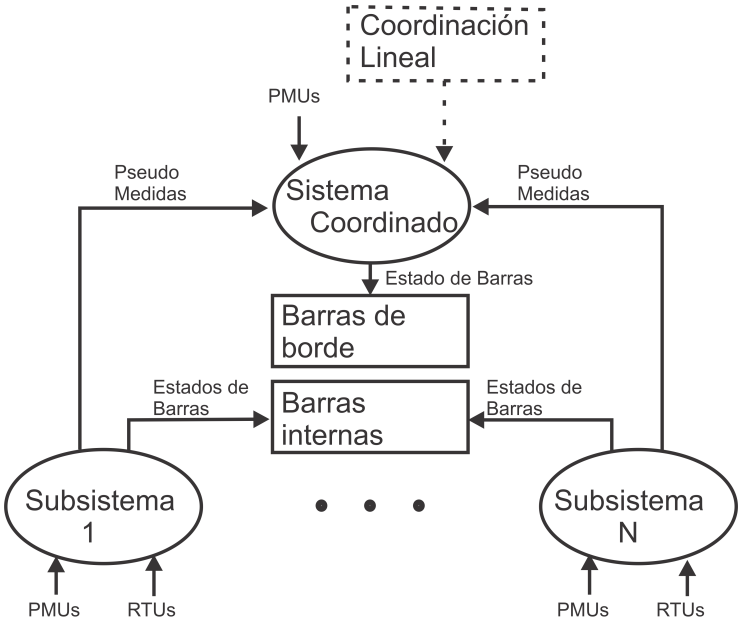


Figura 2.5: Diagrama del algoritmo para división de la red en sub-sistemas

Para el nivel de coordinación se utilizan todas las medidas de PMUs de las barras de contorno. Esto trae aparejado la ventaja de que se relacionan de forma lineal con el vector de estados. Con el fin de aumentar la redundancia de datos, se utiliza la información de las soluciones obtenidas en los sub-sistemas para agregar un conjunto de medidas llamadas *pseudofasores* de tensión, o *pseudomedidas*. La figura (2.5) contiene el diagrama del algoritmo.
El nivel de coordinación queda planteado con la siguiente igualdad:

$$Z = \begin{bmatrix} Z_P \\ Z_{PS} \end{bmatrix} = \begin{bmatrix} B \\ I_{PS} \end{bmatrix} X \tag{2.20}$$

En donde:

- Z_P es el vector de medidas de los PMUs.
- B es una matriz que relaciona corrientes y tensiones medidas por los PMUs con el vector de estado. B tiene vínculo con YBUS.
- Z_{PS} es el vector de pseudo-fasores de tensión obtenidos de las estimaciones.
- I_{PS} matriz que contiene 1 o 0 según corresponda.

2.2.5. Algoritmo adaptativo

Este es el algoritmo que se decidió implementar [4], por lo que se describirá en profundidad en el capítulo 3. Como ya fue mencionado, este algoritmo utiliza

internamente otros algoritmos de estimación, lo que enriquece desde el punto de vista académico su utilización.

2.3. Conclusión y seminario

En el seminario realizado en el IIE con integrantes del Grupo de Control y Estabilidad en Sistemas Eléctricos de Potencia, se expusieron todos los algoritmos antes mencionados para tener una orientación y decisión acerca de cuál era el mejor camino a seguir. La decisión tomada fue la de implementar el **algoritmo adaptativo**, el mismo era el único de los expuestos que tenía evidencias de poder ser ejecutado en tiempo real. Además tenía evidencia de resultados más que aceptables en casos de corto circuito y otras faltas. El proyecto entonces se encaminó a implementar el algoritmo en una red pequeña, y aprender a utilizar el simulador PSS/E para proveernos de los casos de prueba dinámicos.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 3

Descripción del algoritmo adaptativo

La principal ventaja del método a implementar es su rapidez, ya que este aprovecha el hecho de que los PMUs envían sus datos cada 20ms, siendo 50 veces más veloz que la actualización de datos del SCADA. La idea del algoritmo es aprovechar esta cualidad para estimar linealmente y de forma muy rápida el estado del sistema siempre que este varíe poco. Se plantea como hipótesis que la topología de la red no cambia, por lo tanto la matriz de admitancia permanece constante.
En el problema planteado se tiene una red completamente observable por SCADA, y con un conjunto limitado de PMU instalados en determinadas barras. Esto resulta en dos conjuntos de barras, las barras observables por PMU (O) y las barras no-observables por PMU (U). El grupo de barras observables (O) consisten en 2 tipos de barra, las que tienen instalado un PMU, y las que están conectadas a las anteriores a través de una línea que tiene medida de corriente mediante PMU. Como tenemos la corriente por la línea que llega a la barra donde éste se encuentra instalado, y a su vez tenemos la tensión de barra y el modelo de la línea, podemos calcular la tensión en la barra adyacente. El grupo de barras no-observables (U) son todas las barras que no pertenecen al grupo anterior.
El problema de estimación consiste entonces en calcular el voltaje en las barras no-observables (U) a partir de los voltajes medidos por los PMU en las barras observables (O). A continuación veremos el desarrollo de esta estimación.

3.1. Estimador lineal

Las ecuaciones de las corrientes inyectadas en un sistema de potencia se pueden escribir de forma matricial de la siguiente manera:

$$I_{\mathrm{BUS}} = Y_{\mathrm{BUS}}.V_{\mathrm{BUS}}. \tag{3.1}$$

La matriz de admitancias Ybus se divide en 4 sub-matrices, una correspondiente a las barras observables, otra a las no-observables, y las otras dos corresponden a las admitancias de las líneas que conectan barras observables con no-observables. Re-ordenando los vectores de tensiones y corrientes tenemos:

$$\begin{bmatrix} I_O \\ I_U \end{bmatrix} = \begin{bmatrix} Y_{OO} & Y_{OU} \\ Y_{UO} & Y_{UU} \end{bmatrix} . \begin{bmatrix} E_O \\ E_U \end{bmatrix} . \tag{3.2}$$

En un punto de operación particular, las inyecciones I_U de las barras (U) son modeladas como admitancias de carga equivalente. Si N_U es el número de barras (U), las inyecciones en estas barras se escribe como:

$$I_U = \left[\frac{P_i - jQ_i}{E_i^*} \right], \text{ i = 1,2,..., } N_U. \tag{3.3}$$

El vector de admitancias equivalentes sale de la relación $Y = I/E$:

$$Y_L = Y_U = \left[\frac{P_i - jQ_i}{|E_i|^2} \right], \text{ i = 1,2,..., } N_U, \tag{3.4}$$

donde Y_L es una matriz diagonal, siendo los elementos de su diagonal las admitancias equivalentes.
De esta forma, podemos utilizar la información del SCADA para realizar el modelado, ya que éste nos brinda las potencias activas y reactivas inyectadas en las barras, y los módulos de las tensiones en barras.
Insertando este resultado en la ecuación (3.2), esta se convierte en:

$$\begin{bmatrix} I_O \\ 0 \end{bmatrix} = \begin{bmatrix} Y_{OO} & Y_{OU} \\ Y_{UO} & Y_T \end{bmatrix} . \begin{bmatrix} E_O \\ E_U \end{bmatrix}, \tag{3.5}$$

en donde

$$Y_T = Y_{UU} - Y_L. \tag{3.6}$$

Realizando la operación matricial de la fila inferior de la ecuación (3.5), nos lleva a:

$$0 = Y_{UO} E_O + Y_T E_U. \tag{3.7}$$

Resolviendo para E_U nos lleva a la relación entre las tensiones de las barras (U) con las tensiones de las barras (O):

$$E_U = -Y_T^{-1}.Y_{UO}.E_O, \tag{3.8}$$

de donde definimos la matriz H como:

$$H = -Y_T^{-1}.Y_{UO}. \tag{3.9}$$

Esto nos lleva a nuestro modelo final de interpolación de las tensiones en las barras (U) en forma matricial:

$$E_U = H.E_O. \tag{3.10}$$

3.2. Actualización del estimador lineal

Cuando existe algún cambio en los flujos de potencia antes de la llegada de las próximas medidas de los PMU, el estimador lineal de la ecuación (3.9) debe ser actualizado. El interpolador calculado supone un punto de operación de referencia, al cual llamaremos '0':

$$H^0 = -(Y_T^0)^{-1}.Y_{UO}. \tag{3.11}$$

Cualquier desviación de este punto de operación va a producir errores en la ecuación de interpolación

$$E_U = H^0.E_O. \tag{3.12}$$

Una manera de actualizar el interpolador H cuando el sistema cambia su punto de operación se describe a continuación:

Expandiendo la ecuación (3.9) tenemos que:

$$H = -(Y_{UU} - \text{diag}\left[\frac{S_U^*}{|E_U|^2}\right])^{-1}.Y_{UO}, \tag{3.13}$$

ya que:

$$Y_T = Y_{UU} - \text{diag}\left[\frac{S_U^*}{|E_U|^2}\right]. \tag{3.14}$$

Por otro lado, partiendo de la ecuación (3.9) llegamos a:

$$Y_T.H = -Y_{UO}. \tag{3.15}$$

Tomando el diferencial de ambos lados de la ecuación matricial:

$$\Delta Y_T.H + Y_T.\Delta H = 0, \tag{3.16}$$

ya que la matriz Y_{UO} tiene solo admitancias de las líneas, las cuales se consideran constantes. Resolviendo para ΔH tenemos que:

$$\Delta H = -Y_T^{-1}.\Delta Y_T.H. \tag{3.17}$$

Para calcular ΔY_T, realizamos el diferencial de la ecuación (3.14):

$$\Delta Y_T = -\text{diag}\left[\frac{S_U^{1*}}{|E_U^1|^2} - \frac{S_U^{0*}}{|E_U^0|^2}\right], \tag{3.18}$$

ya que la matriz Y_{UU} es una matriz de elementos constantes. Los superíndices denotan la configuración actual (superíndice '1') y la configuración anterior (superíndice '0').
Sustituyendo la ecuación (3.18) en la ecuación (3.17), tenemos:

$$\Delta H = (Y_T^0)^{-1}.\text{diag}\left[\frac{S_U^{1*}}{|E_U^1|^2} - \frac{S_U^{0*}}{|E_U^0|^2}\right].H. \tag{3.19}$$

Esta última ecuación denota la relación entre el cambio incremental en la matriz de interpolación y el cambio en la potencia inyectada en las barras no-observables. Luego, la ecuación de interpolación para las tensiones no-observables es:

$$E_U = (H^0 + \Delta H).E_O. \tag{3.20}$$

Las medidas de los PMU pueden ser obtenidas en tiempos muy cortos comparados con los tiempos de actualización de las medidas del SCADA, entonces ΔH tiene que ser calculada con la llegada de las medidas de los PMU. Pero ΔH es una función de las inyecciones de potencia y los módulos de las tensiones de las barras no-observables para la configuración actual,

$$\Delta H = f(S_U^1, |E_U^1|), \tag{3.21}$$

entonces es necesario utilizar un método WLS para obtener los valores de S_U^1 y $|E_U^1|$. Podemos observar un diagrama de flujo simplificado del algoritmo adaptativo en la figura 3.1.

3.3. Solución no lineal

Para solucionar el problema anterior se aplica el método de mínimos cuadrados ponderados (WLS) a la siguiente ecuación:

$$Z = h(X) + \epsilon, \tag{3.22}$$

en donde X es el vector de estados compuestos por los módulos y fases de las tensiones de todas las barras, Z es un vector que incluye las potencias activas y reactivas adquiridas con los últimos datos de SCADA, y los módulos y fases de las tensiones adquiridas con los últimos datos de PMU de las barras observables. La función $h(X)$ es no-lineal, y es la que relaciona Z con X, y ϵ es el vector de ruido de las medidas, cuyas componentes se asume que son variables aleatorias independientes, gaussianas y de media nula.

La función objetivo a minimizar en el método WLS es la suma de los errores cuadráticos, que como vimos en la introducción (Capitulo 1) se puede definir como:

$$J(X) = [Z - h(X)]^T R^{-1} [Z - h(X)], \tag{3.23}$$

y como se vio en la introducción, utilizando el método de Gauss-Newton llegamos recursivamente a la solución:

$$G(X^k)\Delta X^k = P^T(X^k) R^{-1} [Z - h(X^k)], \tag{3.24}$$

$$G(X^k) = P(X^k)^T R^{-1} P(X^k), \tag{3.25}$$

$$X^{k+1} = \Delta X^k + X^k. \tag{3.26}$$

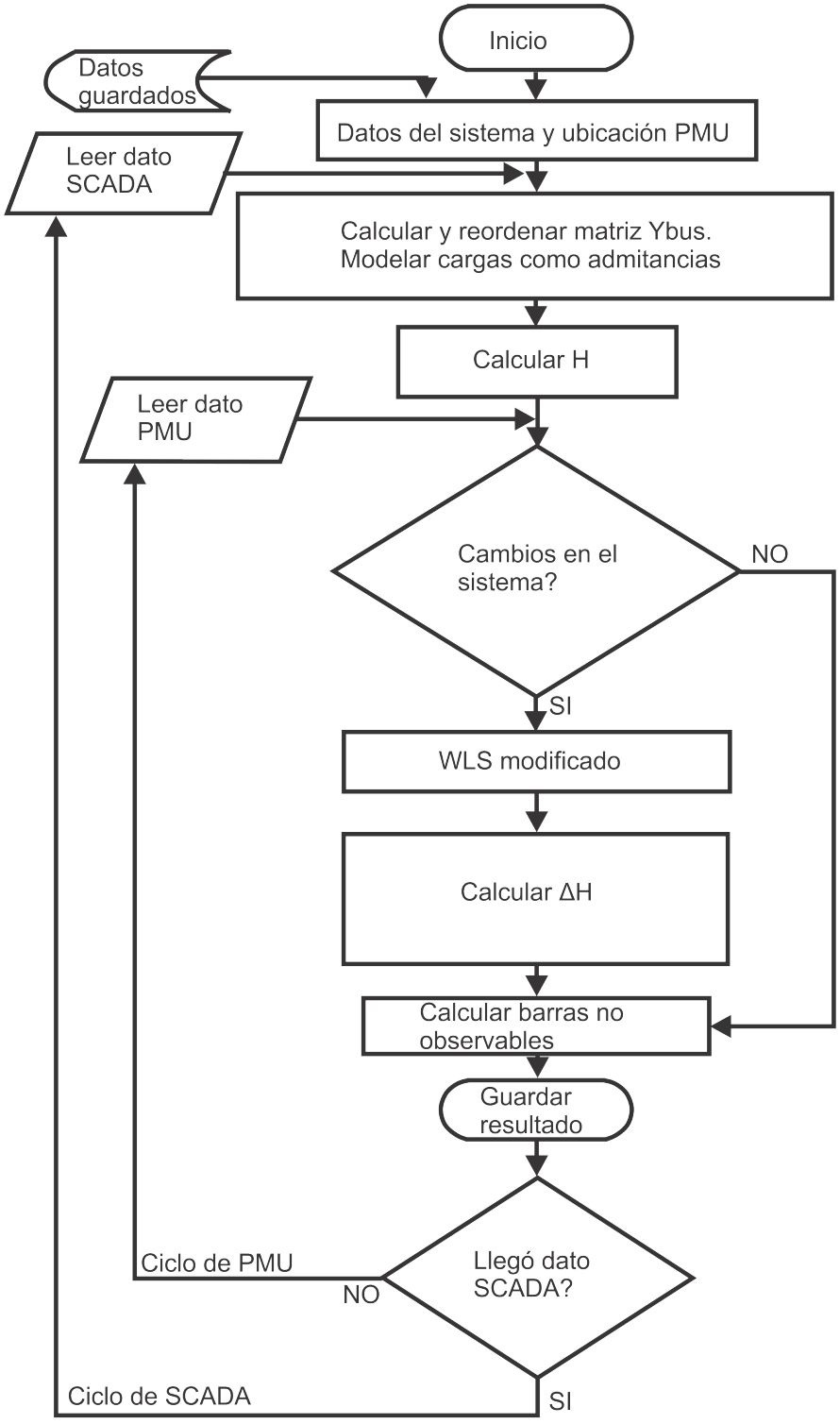


Figura 3.1: Diagrama de flujo del algoritmo adaptativo

Donde P es la Jacobiana de la función $h(X)$, y R es la matriz que contiene los pesos que se le asignan a cada medida como veremos a continuación.

Para inicializar el método iterativo no se utiliza una semilla *flat start*, sino que se utiliza como semilla la última estimación, de este modo se reduce el tiempo requerido para alcanzar la solución.

3.4. Detección de cambios en el sistema

Al momento de la llegada de datos de PMU, para definir si se utiliza el estimador lineal o el método no lineal, es necesario determinar si hubo un cambio o no en el punto de operación del sistema. Esto se hace del siguiente modo:

- Se toman las tensiones de la estimación anterior y utilizando los modelos de las líneas se calculan todas las corrientes.
- A partir de los datos de los PMU recién llegados se crea un mapa de tensiones "híbrido", donde se utilizan las tensiones recién enviadas por los PMU para las barras observables (O), y las tensiones de la estimación anterior para las barras no-observables (U). Al igual que en punto anterior, con estas tensiones se calculan todas las corrientes por las líneas.
- Se comparan estas corrientes, y si la diferencia entre ellas es mayor a un margen preestablecido se decide que hubo un cambio en el sistema.

Además de saber que hubo un cambio, también se tiene conocimiento de en que línea ocurrió el mayor cambio. Teniendo esto en cuenta, se divide el sistema en 4 áreas o grupos diferentes:

- La primer área incluye las barras no-observables (U) que se encuentran conectadas a la línea donde ocurrió el mayor cambio.
- La segunda contiene a todas las barras no-observables (U) conectadas por una línea a las barras de la primer área.
- En la tercer área se encuentra el resto de las barras no-observables (U).
- La última está compuesta por todas las barras observables a través de PMU (O).

Luego a cada una de ellas se le asigna un peso, asociado a la calidad de la información que se tiene de esa área. Por ejemplo, al área compuesta por las barras observables se le asigna el mayor peso (o la menor incertidumbre), y al área conectada a la línea donde se detectó el cambio se le asigna el menor peso. Estos pesos se colocan en la diagonal de la matriz R^{-1}, que es una matriz diagonal que sirve de ponderación para el método WLS.

3.5. Cambios realizados al algoritmo original

Luego de varias pruebas realizadas con el algoritmo implementado, se decidió que con algunos cambios sencillos se podía mejorar su funcionamiento. Estos cambios se implementaron y se verificó que efectivamente las estimaciones mejoraban. Se pasan a detallar los cambios realizados.

3.5.1. Información a priori

Como plantea el autor Simões Costa en uno de sus artículos [3], es posible mejorar la estimación de estado agregando información a priori. La información a priori consiste en un conocimiento adicional de las variables de estado, que está disponible antes de realizar la estimación de estados. En el caso de la estimación de estado aplicada a redes de potencia, existen variables que tienen un control automático sobre su valor para que este se mantenga dentro de determinado margen. Puede ser por ejemplo el caso de las barras de generación, donde se controla el módulo de la tensión para que cumpla una determinada consigna.
Si el vector $\bar{X}$ contiene los valores a priori para las variables de estado, y R_0 es su respectiva matriz de covarianza, esta información a priori puede ser agregada al problema de mínimos cuadrados expandiendo la función objetivo como se detalla a continuación:

$$min_{\hat{X}} \quad J(\hat{X}) = \frac{1}{2}[Z - h(\hat{X})]^T R^{-1}[Z - h(\hat{X})] + \frac{1}{2}(\hat{X} - \bar{X})^T R_0^{-1}(\hat{X} - \bar{X}). \quad (3.27)$$

La solución óptima para este problema nos lleva a la siguiente ecuación:

$$[P^T R^{-1} P + R_0^{-1}]\Delta X = P^T R^{-1} \Delta Z + R_0^{-1} \Delta \bar{X}, \quad (3.28)$$

en donde:

$$\Delta \hat{X} \triangleq (\bar{X} - X^k). \quad (3.29)$$

3.5.2. Actualizar vector de medidas Z

En el momento de realizar la estimación no-lineal, como se explicó en la sección 3.3, el vector Z se forma con las potencias activas y reactivas (P y Q) del último SCADA, y con las tensiones de los datos de PMU recién llegados. Esto maneja una mezcla de información nueva de los PMU con una información vieja del SCADA que puede haber llegado varias decenas de ciclos atrás.
Una mejor opción, y que muestra mejores resultados en la práctica, es generar un vector Z utilizando la última estimación calculada. A partir de la última estimación podemos crear un vector con las potencias activa y reactiva inyectadas en cada barra (P_i, Q_i), de esta forma la estructura del algoritmo original se mantiene.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 4

Simulaciones en PSS/E

PSS/E, siglas de *Power System Simulator for Engineering* es una herramienta de software utilizada para trabajar sobre redes de transmisión. Es un sistema capaz de simular, analizar y optimizar la *performance* del sistema de potencia y provee modelos probabilísticos y dinámicos del mismo. Desarrollado por Siemens PTI en la década del '70 [14], se convirtió en el software comercial más utilizado de este tipo.
Con esta herramienta es posible obtener información de la red la cual es análoga a los datos de entrada que nuestro algoritmo recibiría en el caso de que este estuviese realmente implementado en la red eléctrica. De esta forma, será necesario en la etapa de implementación del algoritmo generar una capa que modele a partir de los resultados obtenidos mediante las simulaciones en PSS/E, los paquetes de datos que los sistemas reales deberían aportar.
La red a estudiar debe ser cargada en el sistema PSS/E. Aquí se requiere información de la topología y de las cargas en las distintas barras. Si se utiliza una red estandarizada IEEE como en este caso, esta información es provista por paquetes de archivos asociados a cada red. En este trabajo se utiliza la red llamada *New England 39 bus system*. Estos archivos pueden ser descargados de la referencia [9].

4.1. Simulaciones estáticas

Para modelar los datos de entrada a nuestro sistema, el PSS/E deberá aportar los datos análogos a los entregados por el sistema SCADA. Este paquete de datos llega habitualmente a una tasa de un paquete cada varios segundos, y contiene información de la topología de la red, los módulos de las tensiones y las cargas de cada barra.
Para obtener estos datos, luego de cargar el modelo de la red en el PSS/E, se ejecuta un flujo de carga cuya resolución determina el valor las variables de estado en todas las barras del sistema. Los datos obtenidos son exportados en un archivo de texto bajo el formato estandarizado IEEE llamado *IEEE power flow common data format (IEEE CDF)*. Estos archivos constan de tablas estandarizadas con la información del sistema. Las variables que nuestro algoritmo debe extraer de

este archivo son las potencias generadas y demandadas en cada barra, los módulos de las tensiones y la topología de la red, donde se incluyen parámetros de líneas, generadores y transformadores.

4.2. Simulaciones dinámicas

Para analizar el comportamiento transitorio del sistema se ejecutan simulaciones dinámicas en el PSS/E. En ellas es posible simular cambios de carga, aperturas de líneas o cortocircuitos, y observar los transitorios que estos provocan. El PSS/E permite seleccionar las variables que se quieren almacenar durante estos transitorios, para luego exportarlas a un archivo *.out*. Éste es un formato propietario, y para importar la información a Matlab es necesario utilizar una función que convierte el archivo *.out* en una serie de matrices. A partir de aquí es posible emular los datos entregados por las PMU, tomando solo los valores simulados en las barras donde se encuentran las PMU.
En el caso de nuestro algoritmo las variables que se almacenan para luego exportarlas a Matlab son módulos y fases de las tensiones de todas las barras. De este modo podemos utilizar las tensiones de todas las barras como las medidas "reales" para luego compararlas con nuestra estimación a partir de PMU.

4.3. Simulaciones realizadas

Todas las simulaciones fueron realizadas sobre la red *New England 39 bus system*, lo cual nos permite darle "globalidad" a nuestros resultados, ya que esta red es pública y cualquiera puede tener acceso a ella. Estas simulaciones consisten en cambios de carga y cortocircuitos que se despejan muy rápidamente. El cambio de carga es considerado un cambio leve en la red y entra en las hipótesis planteadas. Por otro lado, si bien el cortocircuito se va de las hipótesis del problema, es de interés analizar la estimación del transitorio producido luego del despeje de la misma.

Los casos simulados se detallan a continuación.

4.3.1. Casos de prueba

Para analizar el algoritmo implementado, se crearon 4 casos de prueba en el software PSS/E, simulando transitorios eléctricos en la red *IEEE-39 bus system.*

A) El primer caso consiste en un cortocircuito en la línea que une la barra 26 con la barra 29. Éste es un cortocircuito trifásico con impedancia de cortocircuito $0 - j{,}2x10^9$ MVA. La evolución temporal del sistema es la siguiente:

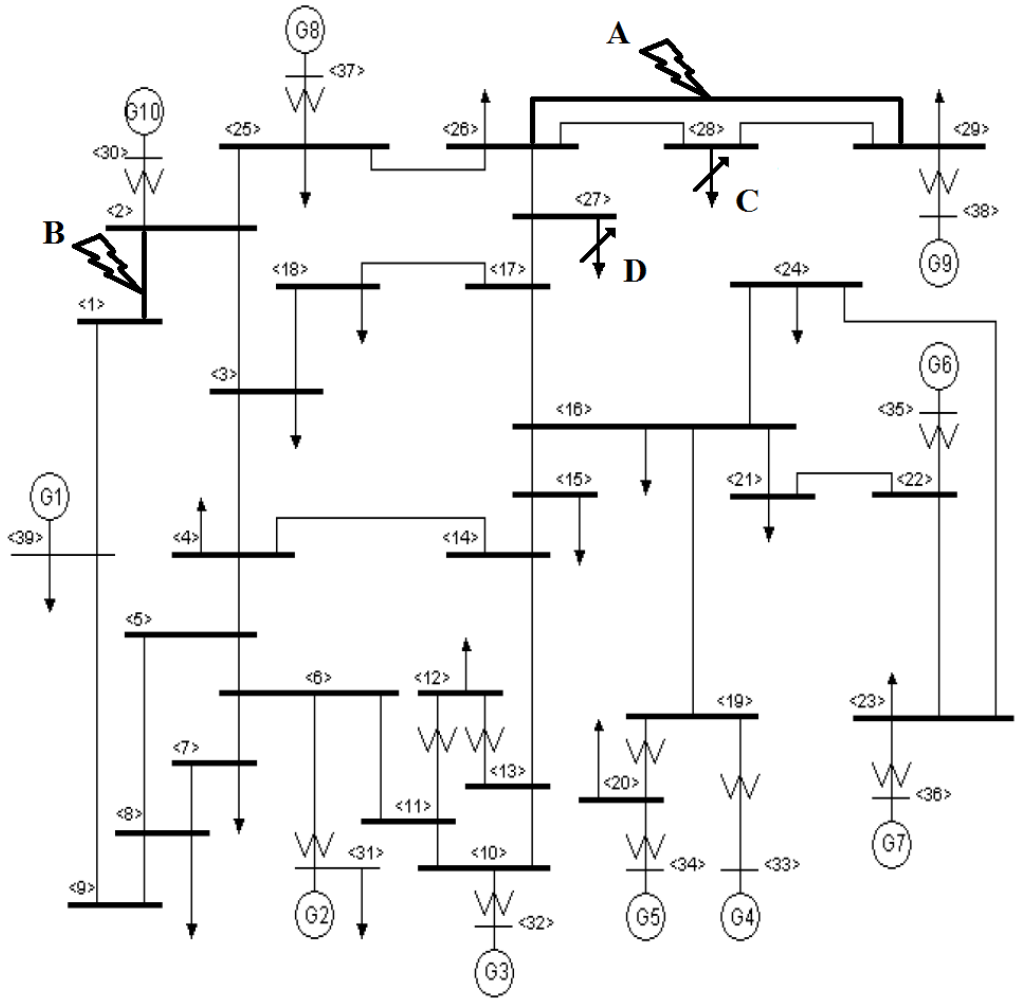


Figura 4.1: Red con casos de prueba

Tiempo	Evento
0 ms	El sistema se encuentra en régimen
250 ms	Entra en cortocircuito la línea 26-29
290 ms	Se abre la línea 26-29
390 ms	Se limpia el cortocircuito y se cierra la línea 26-29
3000 ms	Fin de la simulación

B) El segundo caso de prueba es un cortocircuito en la línea que une la barra 01 con la barra 02. Tiene las mismas características que el cortocircuito anterior, es un cortocircuito trifásico con impedancia de cortocircuito $0 - j{,}2x10^9$ MVA. La evolución temporal del sistema es la siguiente:

Tiempo	Evento
0 ms	El sistema se encuentra en régimen
250 ms	Entra en cortocircuito la línea 01-02
290 ms	Se abre la línea 01-02
390 ms	Se limpia el cortocircuito y se cierra la línea 01-02
3000 ms	Fin de la simulación

C) Este caso consiste en un cambio de carga en la barra 28. En el modelo de la carga se modifica la componente de admitancia constante, la cual pasa de tener un valor de $241 - j119$ MVA a valer $441 - j319$ MVA. La evolución temporal del sistema es la siguiente:

Tiempo	Evento
0 ms	El sistema se encuentra en régimen
250 ms	Se realiza el cambio de carga en la barra 28
8000 ms	Fin de la simulación

D) Al igual que el caso anterior, se realiza un cambio de carga, pero esta vez es en la barra 27. En el modelo de la carga se modifica la componente de admitancia constante, la cual pasa de valer $246 - j73$ MVA a valer $346 - j223$ MVA. La evolución temporal del sistema es la siguiente:

Tiempo	Evento
0 ms	El sistema se encuentra en régimen
250 ms	Se realiza el cambio de carga en la barra 27
8000 ms	Fin de la simulación

Capítulo 5

Implementación del algoritmo

El desarrollo del algoritmo se lleva a cabo en Matlab, herramienta de software matemático que ofrece un entorno de desarrollo y un lenguaje de programación propio y orientado a la matemática. La ventaja de desarrollar en este lenguaje es la facilidad para programar e investigar el algoritmo adaptativo. También simplifica la generación de contenido adicional, el ejemplo más inmediato es la facilidad para graficar. Incluye múltiples librerías a disposición del programador. La desventaja de este lenguaje es la velocidad de ejecución, lo que exige una posterior migración de código a otro lenguaje si se quiere mejorar la *performance*, pero para los efectos de desarrollo es el lenguaje adecuado.

El software en Matlab a programar debe extraer de las simulaciones ejecutadas en PSS/E la información que en un futuro le brindará el sistema SCADA y los PMUs. Se espera que la información que brinde el SCADA en cada ciclo de actualización del mismo sea la matriz de admitancia de la red, las potencias activas y reactivas inyectadas, y módulo del voltaje en las distintas barras. Asociado a esto, el software también debe ser capaz de obtener de las tablas del PSS/E la información que le llegaría de los PMUs en cada paquete de datos. Se le llama paquete de datos proveniente de los PMUs al conjunto de datos que llega en determinado instante, con contenido de todas las muestras reportadas por los distintos PMUs en el instante de muestreo.

El paquete de datos contiene módulo y fase de la tensión en cada barra que posea uno de estos sistemas de medición. Es opcional, y se contempla, que también contenga módulo y fase de la corriente por una o varias líneas conectadas a la barra. También debe incluir una marca de tiempo sincronizada asociada a la medida.

Para mejorar el tiempo de ejecución del algoritmo se migra a C++ parte del código del algoritmo. Aquí se extraerá de Matlab toda la generación previa del entorno para poder aplicar el algoritmo, de tal forma que el ejecutable en C++ sea capaz de funcionar sólo recibiendo los paquetes de datos provenientes de los PMUs.

5.1. Factibilidad

La factibilidad del proyecto está dada por la disponibilidad de recursos necesarios para alcanzar los objetivos que el algoritmo adaptativo debe cumplir. Se apoya en 3 aspectos: operativo, técnico y económico. La factibilidad operativa se vincula a si el objetivo puede ser garantizado. Esto se cumple al partir de la base que el algoritmo fue correctamente desarrollado y hay un artículo que muestra su desempeño. El aspecto económico no aplica al caso. Los recursos técnicos se disponen. Una computadora convencional actual tiene capacidad para resolver el algoritmo, ya que por la dimensión del sistema eléctrico elegido, el cual es pequeño, las matrices a procesar rara vez excederán los mil elementos, y la cantidad de estas matrices que deben ser almacenadas en memoria también es baja. Con estos tres puntos marcados, se concluye que es factible el desarrollo del algoritmo adaptativo.

5.2. Requisitos e hipótesis

Analicemos las necesidades del algoritmo adaptativo para que su implementación sea exitosa. Es un requisito que a la salida del algoritmo adaptativo se tengan estimaciones de todas las barras del sistema, por lo que se torna en una condición para la entrada tener información suficiente de cada barra. Una hipótesis que siempre se manejó en el desarrollo de este documento es que la red es totalmente alcanzable por SCADA. Esto implica que este entregue información de todas las barras y líneas de la red. Otro requisito es la tasa de estimación deseada, por lo cual es necesario que la tasa de llegada de paquetes de PMU sea mayor o igual a esta. Es necesario poder modificar ciertos parámetros, por ejemplo la ubicación de los PMUs, los pesos de cada área frente a un cambio en la red, los parámetros para determinar si existió un cambio y márgenes de parada para las iteraciones. Para facilitar el ingreso de esta información, desde el usuario hasta el programa, se utiliza de intermediario un archivo de texto llamado *configuracion.txt*, el cual alberga todos estos parámetros.

5.3. Diseño

Es importante diseñar correctamente la forma de implementar el algoritmo, sobre todo cuando se tiene en cuenta la futura migración de código. Para esto es muy útil desarrollar el programa siguiendo un diagrama de bloques, donde cada bloque cumple una función determinada, sus entradas son las salidas de uno o varios bloques que lo anteceden, y a su vez el resultado que este devuelve puede ser utilizado por los siguientes bloques.
En la figura (5.1) se muestra el diagrama de bloques desarrollado. Un análisis grueso permite separar este diagrama en tres secciones: Ingreso de información, procesado del SCADA y finalmente procesado de las PMUs. La primera parte corresponde a la inicialización del programa, donde se leen e ingresan en memoria los archivos de configuración y los archivos con las simulaciones tanto dinámi-

ca como estática, estos en el diagrama están representados por los bloques *leer configuración*, *IEEE common data format* y *Simulaciones Dinámicas*. El sistema desarrollado toma como hipótesis que el primer paquete de datos que le ingresa debe provenir de un SCADA, por lo cual queda en modo de espera hasta que le llegue el primer paquete de este tipo.

El diagrama de bloques presenta un bloque de bifurcación con una pregunta, *Llegó SCADA o PMU?*, es aquí donde según el paquete arribado es el tipo de procesamiento que se le da. En este punto acaba de llegar al sistema un paquete de datos proveniente del SCADA, por lo que se entra al segundo gran bloque, denominado *procesar SCADA*, quien se encarga de generar el entorno para trabajar. En el diagrama son los bloques de la mitad inferior de la figura. Este proceso comienza simulando la llegada del SCADA, el cual a partir de los resultados obtenidos de la simulación estática, genera el dato análogo al entregado por el sistema SCADA. Con estos datos el sistema es capaz de realizar cuatro procesamientos independientes: realizar una estimación clásica para determinar módulo y fase de todas las tensiones, identificar las barras que tengan generadores y generar una matriz guardando esa información, y detectar las barras observables del sistema las cuales combinadas con la matriz de admitancia permite calcular la matriz de estimación lineal H. Con el resultado de la estimación clásica recién obtenida se procede a calcular las corrientes por las líneas y este dato es guardado en memoria. Como fue armada la matriz de estimación lineal H, ahora el sistema está en condiciones de recibir un paquete de PMU.

El ciclo vuelve a la pregunta *Llegó SCADA o PMU?* pero ahora consideramos que llegó un PMU, por lo que se pasa a la tercer gran área de procesamiento, el procesado del dato de PMU. El comienzo de esta línea es el bloque que simula la llegada de PMU, aquí el sistema genera a partir de los datos extraídos de las simulaciones dinámicas, los datos análogos a los que serían necesariamente entregados por las PMU. Luego se entra en otro bloque diversificador, el cual detecta cambios en la red, como se vio en el capítulo (3.4). Si la respuesta es que no hubo cambios, se procede a la estimación lineal. De lo contrario, el sistema detecta donde fue que se produjo ese cambio y agrupa las barras según su proximidad al evento. Estos grupos, combinados con los datos de los PMU y con la última estimación realizada, se utilizan como entrada del bloque que genera la estimación no lineal explicada en el capítulo (3.3).

5.4. Análisis de frontera simulación-realidad

Es necesario implementar en forma adecuada la capa que convierte los datos provenientes de las simulaciones en lo que serían los datos aportados por los equipos en la realidad. Los principales equipos de medición involucrados son el sistema SCADA y el conjunto de PMUs.
La información que brinda el SCADA en cada ciclo de actualización es la matriz

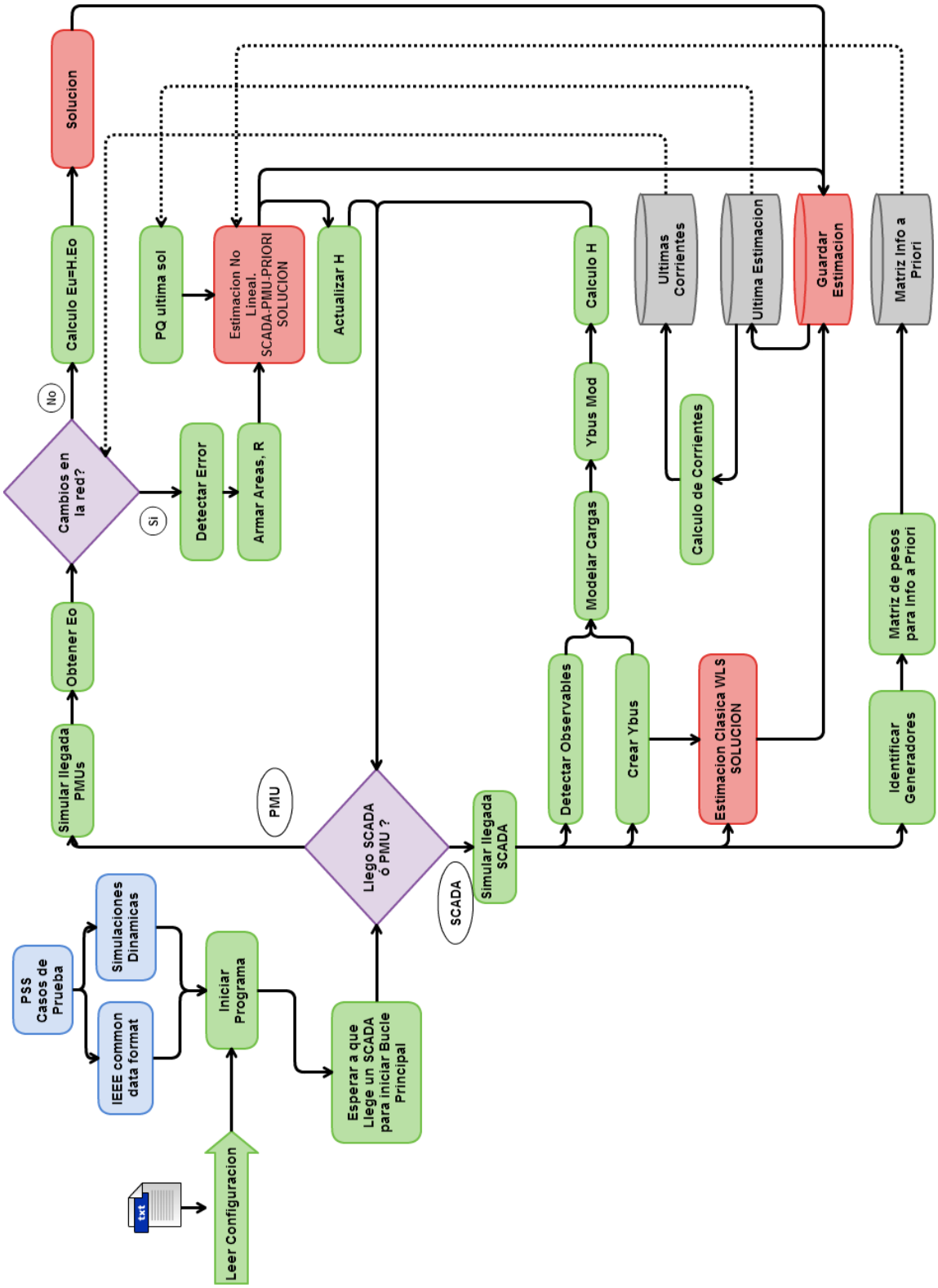


Figura 5.1: Diagrama de bloques del algoritmo implementado

de admitancia de la red, las potencias activas y reactivas inyectadas, y el módulo de tensión en las distintas barras. Estos valores son obtenidos del flujo de carga ejecutado por el PSS/E y exportados en los archivos estandarizados *IEEE common data format*. Estos archivos contienen dos tablas de donde se extrae la siguiente información:

Para cada barra i:

- Módulo de tensión (V_i).
- Carga y generación en MW (P_{Li} y P_{Gi} respectivamente).
- Carga y generación en MVAR (Q_{Li} y Q_{Gi}).
- Susceptancia *shunt* de la barra (B_{i0}).

Con esta información se genera el vector de potencias activas y reactivas inyectadas, llamado *z medidas*. Sea n el número de barras del sistema:

$$\text{z medidas} = [P_1, P_2, ..., P_n, Q_1, Q_2, ..., Q_n], \tag{5.1}$$

$$P_i = -P_{Li} + P_{Gi}, \tag{5.2}$$

$$Q_i = -Q_{Li} + Q_{Gi}. \tag{5.3}$$

Para cada línea i-j se obtiene la siguiente información:

- Susceptancia a tierra de la línea (B_{ij}).
- Resistencia y reactancia de la línea (R_{ij} y X_{ij}).

Pasando estos datos a admitancias quedan:

$$Y_{ij} = \frac{1}{R_{ij} + j.X_{ij}} \ , \tag{5.4}$$

$$Y_{ij}^s = \frac{1}{B_{ij}/2} \ , \tag{5.5}$$

$$Y_{i0}^s = \frac{1}{B_{i0}} \ . \tag{5.6}$$

La matriz de admitancia queda entonces como:

$$\begin{pmatrix} I_1 \\ \vdots \\ I_n \end{pmatrix} = \begin{pmatrix} m_{11} & \cdots & m_{1n} \\ \vdots & \ddots & \vdots \\ m_{n1} & \cdots & m_{nn} \end{pmatrix} \begin{pmatrix} V_1 \\ \vdots \\ V_n \end{pmatrix} \ , \tag{5.7}$$

en donde:

$$m_{ij} = -Yij \qquad i \neq j, \tag{5.8}$$

$$m_{ii} = \sum_{k=1}^{n} Y_{ik} + \sum_{k=1}^{n} Y_{ik}^{S} + Y_{i0}^{S}. \tag{5.9}$$

Hay información que debe ser ingresada por el usuario, por ejemplo qué barras del sistema tienen instaladas unidades PMU. Para esto se provee de un archivo de configuración, en el cual se ingresan varios datos importantes para el algoritmo. Las entradas para este archivo de configuración son las siguientes:

- Archivos de simulaciones PSS/E a cargar.
- Ubicación de las PMUs.
- Líneas cuya corriente es reportada por la PMU.
- Barra *slack* del sistema.
- Potencia base del sistema.
- Márgenes y límites para iteraciones.
- Varianzas para ponderar áreas de influencia.
- Muestra para exportar y procesar en C++.

Con esta información de configuración, el sistema está en condiciones de simular el dato proveniente de las PMUs. Como ya se expresó anteriormente, estos paquetes de datos contendrán módulo y fase de la tensión y opcionalmente módulo y fase de las corrientes por las líneas que sean monitoreadas. Denominamos barra observable(O) a toda barra que tenga instalada una PMU, y a las barras que estén conectadas a esta última mediante líneas que tengan su corriente monitoreada por la PMU.

Para cada instante de muestreo, los datos provenientes de las unidades son:

- Sincrofasores de tensión en barras con PMU instalado.
- Sincrofasores de corriente en líneas monitoreadas.

La forma de simular esto es, de todo el vector de tensiones simulado en PSS/E para ese instante de muestreo, aislar las muestras correspondientes a barras con PMU instalado.

Para un instante de muestreo T y un sistema de n barras:

$$V_T = [\overrightarrow{V_1}, \overrightarrow{V_2}, ..., \overrightarrow{V_i}, ..., \overrightarrow{V_k}, ..., \overrightarrow{V_n}]. \tag{5.10}$$

El vector que oficia como máscara, expresado en la siguiente ecuación, pone como ejemplo dos PMUs ubicados en la barra i y en la barra k.

$$\text{máscara PMUs} = [0, 0, ..., 1, ..., 1, ..., 0]. \tag{5.11}$$

Los vectores (5.10) y (5.11) son del mismo largo, por lo que si se multiplica elemento a elemento de dichos vectores, se obtiene un vector (5.12) con las tensiones asociadas a los sincrofasores entregados por las PMUs. Esta forma de representación es de gran utilidad ya que combina en un solo vector el dato del sincrofasor, y la barra a la cual está asociada esa medida.

$$V_{\text{PMU}} = [0, 0, ..., \overrightarrow{V_i}, ..., \overrightarrow{V_k}, ..., 0]. \tag{5.12}$$

El cálculo para la simulación de corrientes reportadas por la PMU sobre la línea ij se expresa en la ecuación (5.13):

$$\overrightarrow{I_{ij}} = \frac{\overrightarrow{V_i} - \overrightarrow{V_j}}{R_{ij} + j.X_{ij}} + \overrightarrow{V_i}.\frac{B_{ij}}{2} \ . \tag{5.13}$$

5.5. Salida de resultados

Los resultados obtenidos por la estimación son graficados, mostrando módulo o fase real de tensión contra la estimación en todas las barras del sistema. Esto permite una visión general de cómo se comporta el algoritmo en todas las barras. No es de interés presentar todas estas curvas en el capítulo de análisis de resultados, pero si analizar puntualmente el comportamiento de algunas barras en torno al punto de falla. Por otra parte, siempre es importante unificar los resultados de todas las gráficas en una sola para poder comparar el desempeño de varias estimaciones distintas bajo diferentes parámetros de configuración. Para esto se grafica también el error cuadrático medio de la estimación de cada muestra. Para el conjunto de datos recibido en el instante i sobre el cual se realiza la estimación, se define el error cuadrático medio de ese instante i como:

$$\text{ECM}_i = 100\sqrt{\frac{1}{n}\sum_{k=1}^{n} |\vec{V}_{k\ \text{REAL}} - \vec{V}_{k\ \text{ESTIMADO}}|^2} \ . \tag{5.14}$$

Es interesante conocer el tiempo insumido por Matlab para el procesado de cada paquete, por lo que también se despliega una gráfica de tiempos. Estos tiempos son calculados para cada paquete de datos provenientes de PMU, que es en donde se tiene la mayor exigencia de optimizar el tiempo. Este tiempo va desde que llega el paquete hasta que el sistema está en condiciones de recibir un paquete nuevo. Si se realizó el ciclo lineal, este tiempo incluye la detección de cambios en la red y el producto de la matriz H con el vector de tensiones observables. Si de lo contrario se realiza el algoritmo no lineal, este tiempo incluye aparte de la determinación de cambios en la red, detectar su ubicación, armar las áreas, la estimación no lineal SCADA-PMU, y la actualización de la matriz H. Estos resultados se muestran en escala logarítmica debido a la gran diferencia entre lo que requiere el procesado lineal, del orden de milisegundos, contra lo que requiere el procesado no lineal, del orden de segundos.

5.6. Migración de código

Se desea bajar el tiempo de ejecución del algoritmo para que pueda trabajar a una tasa de 50 paquetes de PMU por segundo. Para esto es necesario migrar el código de Matlab a un lenguaje más rápido. Aquí se opta por el lenguaje C++, ya que este es adecuado para generar algoritmos de gran velocidad de ejecución. Como referencia a esta decisión se tiene un artículo [8] donde se comparan cuatro lenguajes en función de su tiempo de ejecución, estos son C++, Java, Go y Scala. En el artículo se concluye que C++ es quién obtiene mejores tiempos de ejecución, destacándose aún más cuando el código se encuentra optimizado.

Se opta por migrar a C++ solamente el bucle que procesa los datos de PMU, desde que el paquete llega al sistema hasta que se obtiene la estimación realizada, y en el caso del procesado no lineal, la actualización a la matriz H. El algoritmo implementado en C++ necesita como datos de entrada, además del reporte de los PMUs, la demás información del entorno en ese instante. Esto es, para el caso lineal, se precisa la matriz H actual, y los sincrofasores reportados por los PMUs en barras observables. Para el caso no lineal, se necesita conocer el estado anterior al recibido, la matriz de admitancia, la ubicación de las PMUs y los pesos para las diferentes áreas.
La forma para implementar esto en Matlab es seleccionando que muestra se desea procesar con C++, de esta forma Matlab genera el archivo adecuado para ser levantado por el programa en C++. La utilidad de esto es poder evaluar el tiempo insumido para procesar una muestra por el C++, evaluando de esta forma si se alcanzó bajar el límite de los 20 ms para poder ejecutar el sistema a 50 Hz.
El código fue creado en Linux, en el entorno de desarrollo Kdevelop, y para la manipulación de matrices utiliza la biblioteca EIGEN.

Capítulo 6

Resultados obtenidos

Este capítulo desarrollará los resultados obtenidos al poner a prueba el algoritmo. Aquí se tratarán solamente los aspectos que consideramos son de interés para el lector. Por ejemplo, se mostrará la capacidad de estimación del sistema cuando frente a la misma falta se tienen más o menos PMU instalados y cuán cerca están estos de la barra a estimar.

Antes de comenzar con los resultados el lector deberá tener en mente dos clasificaciones que se le dan a las barras. La primera se asocia a la definición de *nivel de observabilidad* de una barra, que es la mínima cantidad de saltos que separan la barra en cuestión y la barra observable más cercana a ella, esta clasificación queda fija luego de definir la ubicación de los PMUs.
La segunda corresponde a la separación entre una barra no observable y la barra que se encuentra en falta, esta clasificación sólo tiene sentido durante el análisis de una falta en el sistema. Se denomina con la letra G, numerados de 1 a 4.
Las mismas se detallan en las siguientes tablas:

Nivel de observabilidad	**Detalle**
Nivel 0	Barras observables
Nivel 1	Barras a un salto de una observable
Nivel 2	Barras a dos saltos de una observable
Nivel 3	Barras a tres saltos de una observable

Grupo	**Detalle**
Barras G1	Barras pegadas a la falta y no observable
Barras G2	Barras conectadas a G1 y no observable
Barras G3	Barras conectadas a G2 y no observable
Barras G4	Barras observable

Este capítulo presentará:

- **Cantidad de PMU en la red**. En esta sección se variará la cantidad de PMUs en la red con el fin de evaluar la precisión de la estimación en función de este factor y de los niveles de observabilidad. Estas pruebas se realizan en un caso de cortocircuito (caso de prueba A).

- **Analizar los tiempos de ejecución del algoritmo**. Estudio del tiempo insumido en la estimación, comparando la *performance* de Matlab contra C++.

Más resultados son presentados en el anexo:

- **Precisión de la estimación ante cambio de carga**. En esta sección se evaluará la estimación frente a un cambio de carga observando el desempeño en tres barras con distintos niveles de observabilidad (caso de prueba D).
- **Precisión de la estimación según distancia de PMUs a la falta.** Estimación en barras de G3 con niveles de observabilidad 1 y 2, cuando se tiene o no PMU instalada en la barra en falta.
- **Análisis de las varianzas en la estimación no lineal**. En esta sección se analizará cómo afecta a la estimación la asignación de pesos a los distintos grupos G1 a G4.

6.1. Cantidad de PMU en la red

El objetivo de este análisis es estudiar la estimación bajo el mismo caso de prueba y observando las mismas barras cuando en la red se varía la cantidad de PMUs. Se utiliza como caso de prueba el cortocircuito en la barra 26 (caso de prueba A). La ubicación para los PMUs es extraída de un artículo en el que se estudian ubicaciones óptimas para estos equipos [17]. Éste sugiere que una distribución óptima para la red *IEEE 39 bus system* se da colocando 8 unidades, ubicadas en las barras 3, 8, 13, 16, 20, 23, 25 y 29. En esa situación el sistema no tiene ninguna barra nivel 3, por ese motivo se cambió a las barras 3, **4**, 13, 16, 20, 23, 25 y 29.

Se buscó que las barras a observar se encontrasen en distintas situaciones con respecto a la falta y a su nivel de observabilidad, a modo de tener un espectro lo más amplio posible en los resultados. Se seleccionaron las barras 8, 9, 26 y 27 que para ambas configuraciones de PMUs cumplen:

- Barra 26: Tiene nivel de observabilidad 1, y por su ubicación respecto a la falta pertenece al grupo G1.
- Barra 27: Tiene nivel de observabilidad 2, y pertenece al grupo G2.
- Barra 8: Tiene nivel de observabilidad 2, y pertenece al grupo G3.
- Barra 9: Tiene nivel de observabilidad 3, y pertenece al grupo G3.

Para la segunda parte de esta prueba se quitaron 4 PMUs de la red. El criterio para quitarlas fue que se mantengan las propiedades de las barras a observar, así fue que se mantuvieron los PMUs en las barras 4, 13, 25 y 29. En la figura 6.1 se

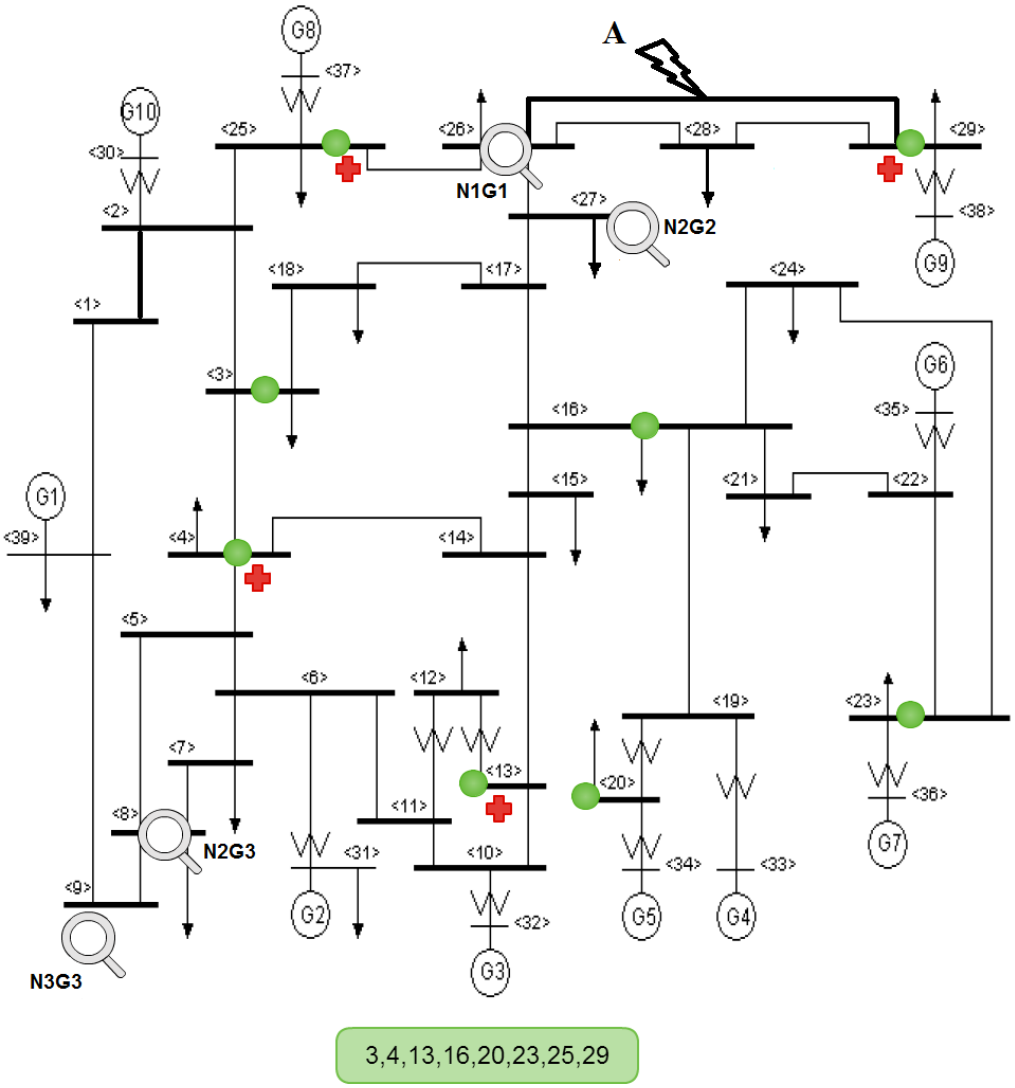


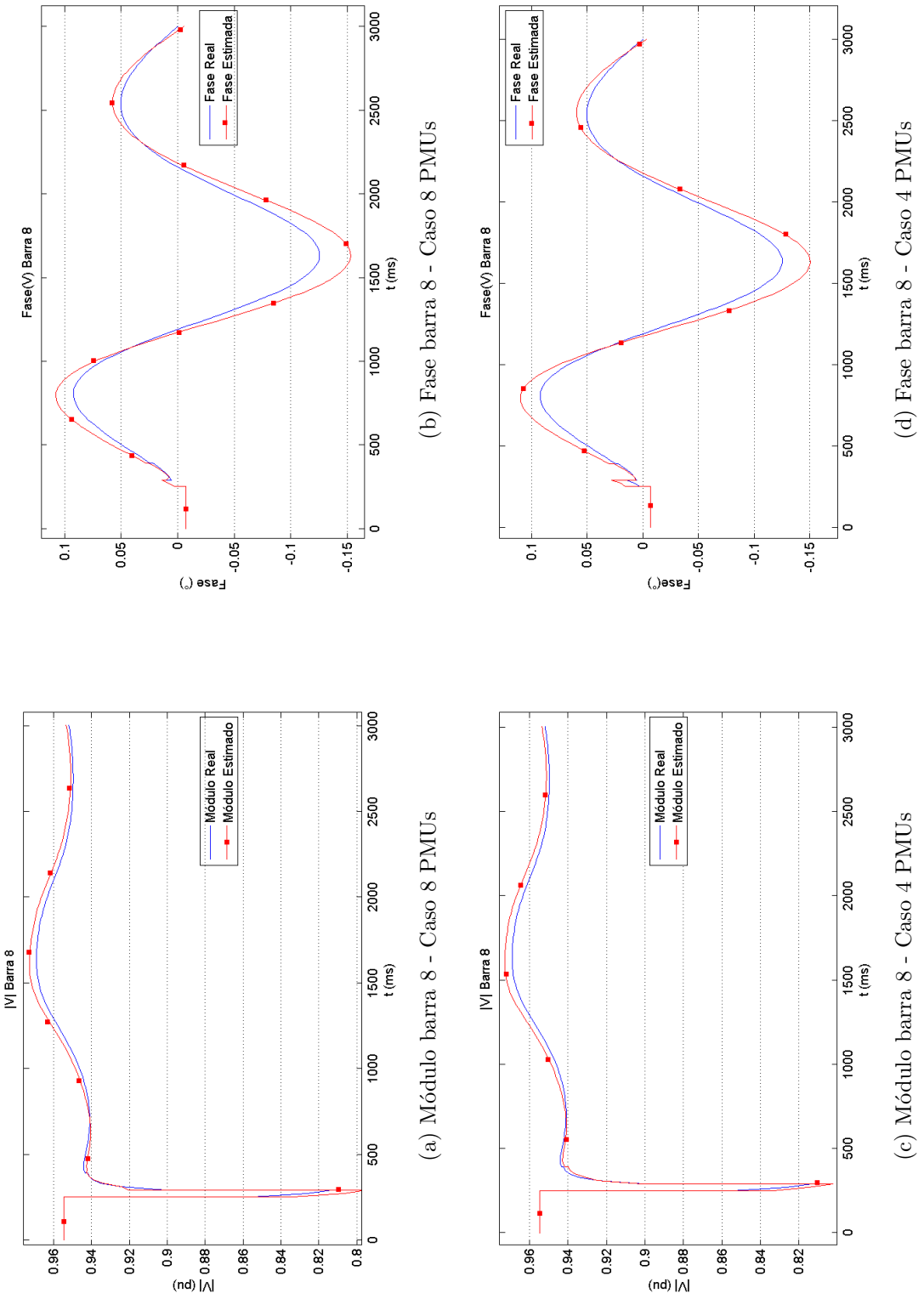
Figura 6.1: Monitoreo de la red y barras analizadas

puede observar la ubicación de las PMUs y que barras se analizaron.

En las próximas páginas veremos las gráficas con los resultados de las estimaciones (figuras 6.2, 6.3, 6.4 y 6.5), donde se muestra el valor real y el estimado para módulo y fase de la tensión de cada barra.

Estas gráficas concuerdan con lo esperado, cuanto más PMUs hay en la red, mejor es la estimación. Una forma de poder analizar realmente esta afirmación es mirando el error cuadrático medio de todas las barras (ver capítulo 5.5). Esto se grafica en las figuras 6.6 y 6.7.

Es importante aclarar que en el intervalo de tiempo que la red se encuentra en falta y cuando la línea está fuera de servicio (entre 250ms y 390ms), el modelo de



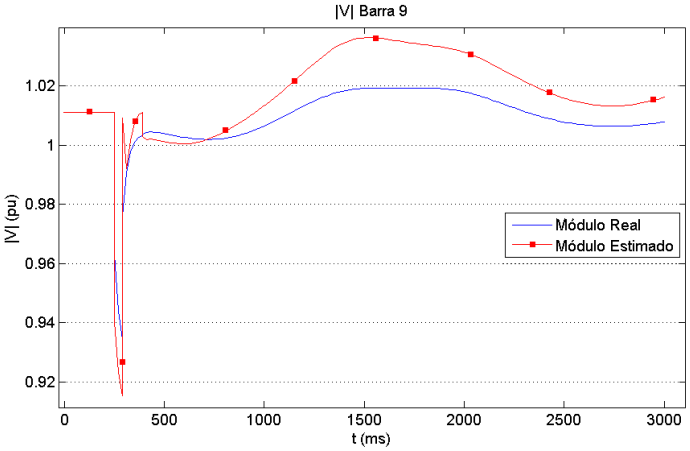
(a) Módulo barra 8 - Caso 8 PMUs

(b) Fase barra 8 - Caso 8 PMUs

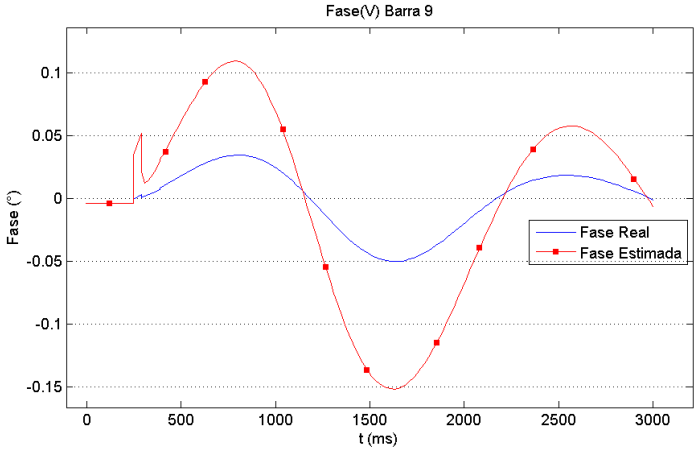
(c) Módulo barra 8 - Caso 4 PMUs

(d) Fase barra 8 - Caso 4 PMUs

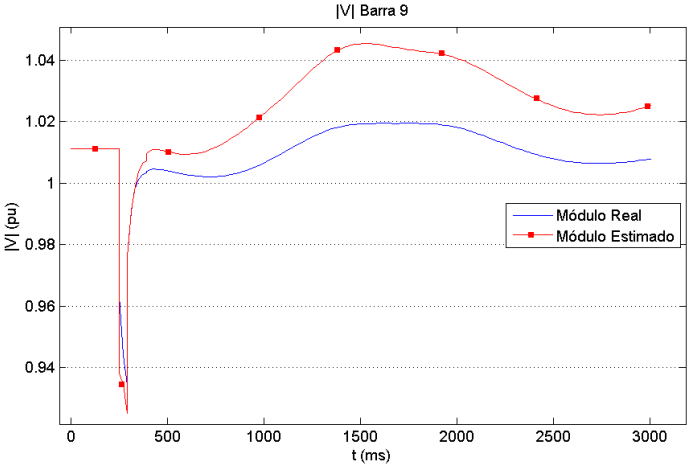
Figura 6.2: Estimaciones de tensión en barra 8 - Observabilidad Nivel 2 - Grupo G3



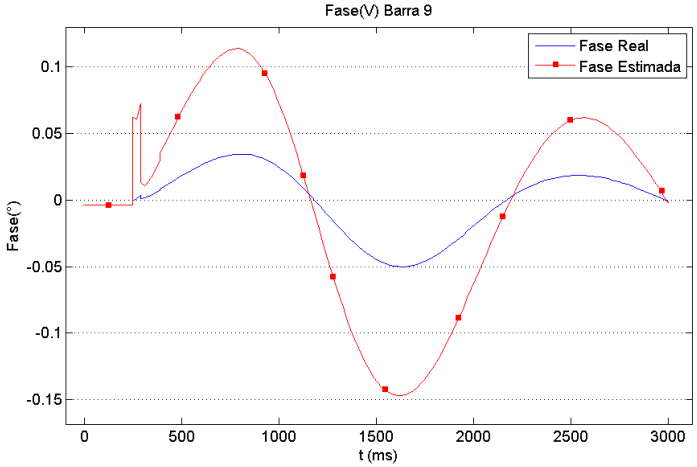
(a) Módulo barra 9 - Caso 8 PMUs



(b) Fase barra 9 - Caso 8 PMUs

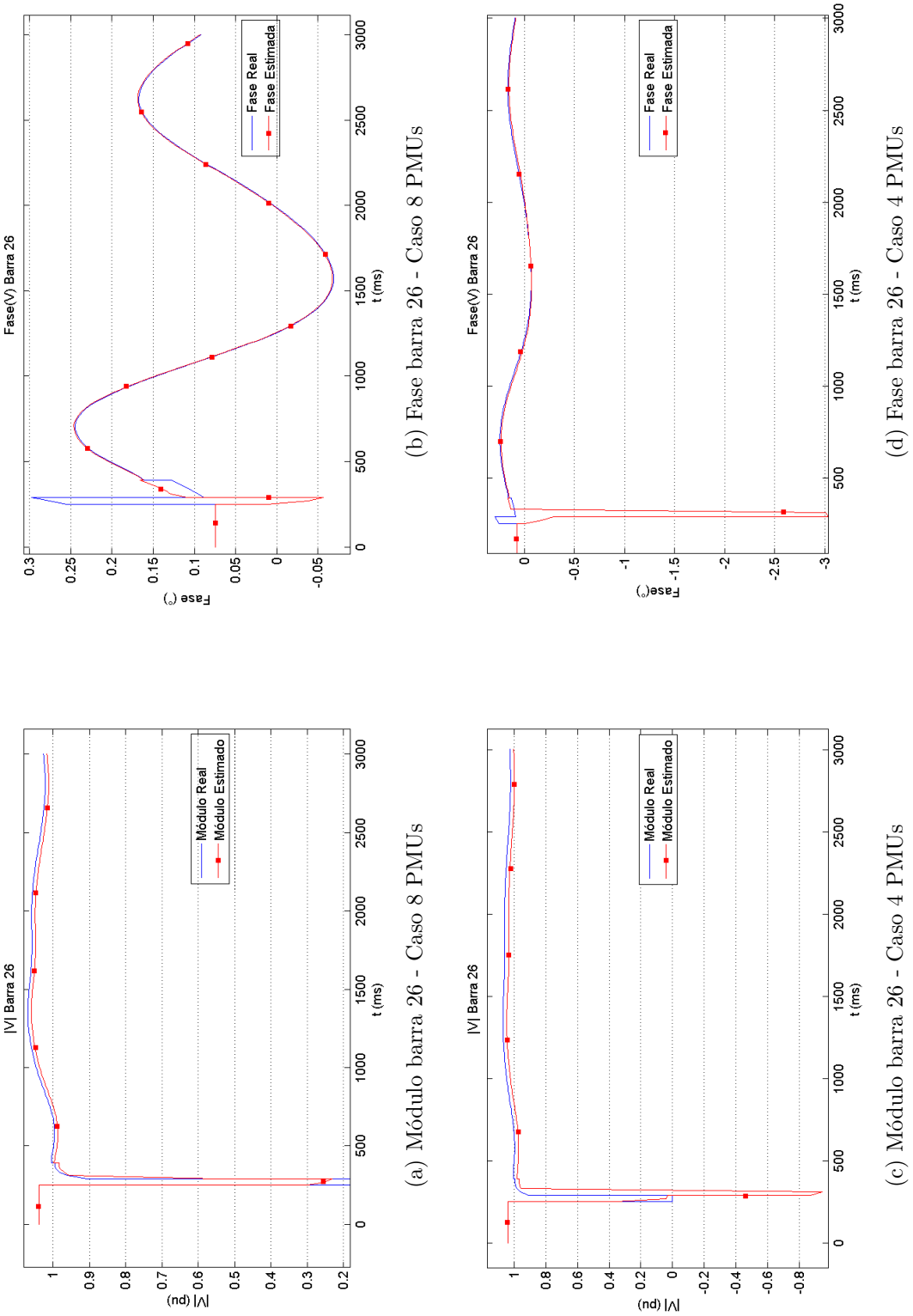


(c) Módulo barra 9 - Caso 4 PMUs



(d) Fase barra 9 - Caso 4 PMUs

Figura 6.3: Estimaciones de tensión en barra 9 - Observabilidad Nivel 3 - Grupo G3



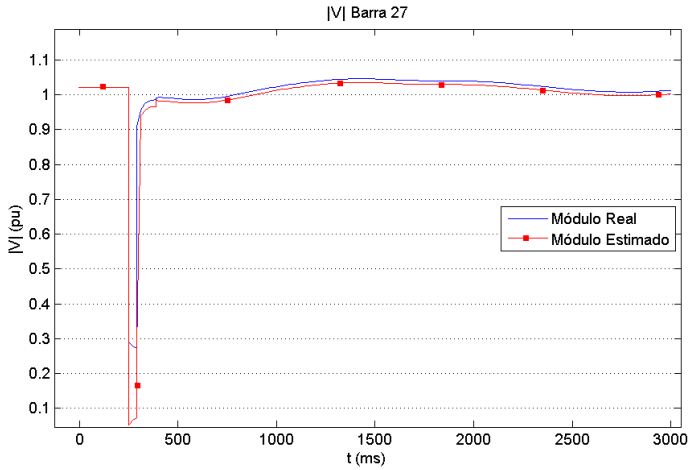
(a) Módulo barra 26 - Caso 8 PMUs

(b) Fase barra 26 - Caso 8 PMUs

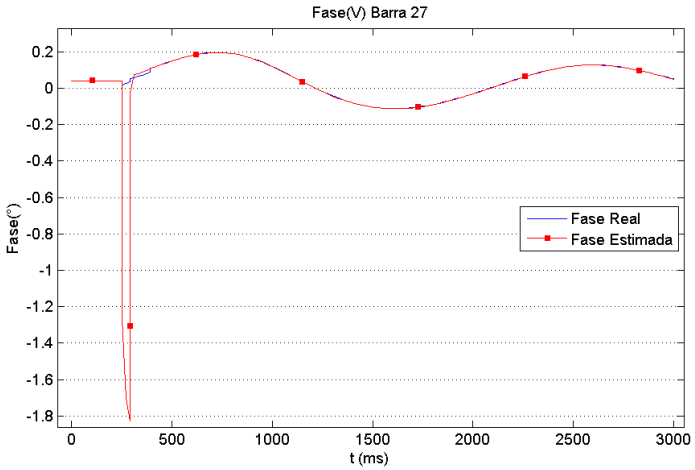
(c) Módulo barra 26 - Caso 4 PMUs

(d) Fase barra 26 - Caso 4 PMUs

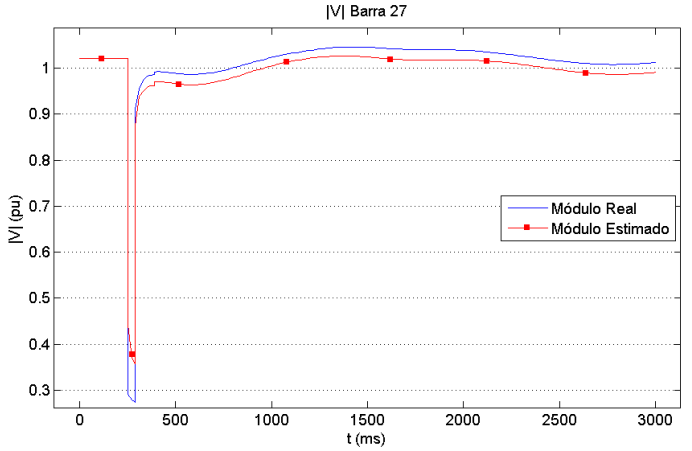
Figura 6.4: Estimaciones de tensión en barra 26 - Observabilidad Nivel 1 - Grupo G1



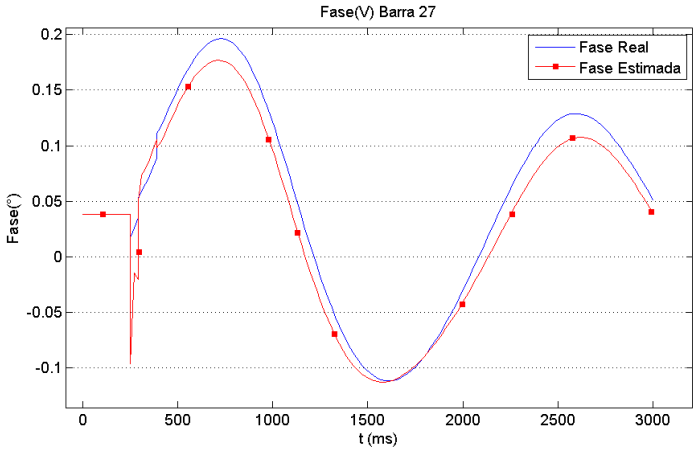
(a) Módulo barra 27 - Caso 8 PMUs



(b) Fase barra 27 - Caso 8 PMUs



(c) Módulo barra 27 - Caso 4 PMUs



(d) Fase barra 27 - Caso 4 PMUs

Figura 6.5: Estimaciones de tensión en barra 27 - Observabilidad Nivel 2 - Grupo G2

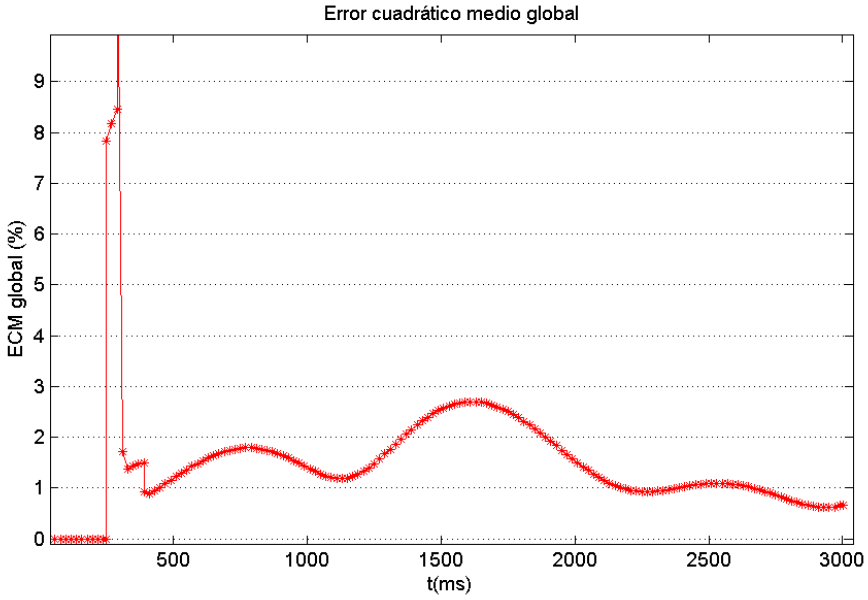


Figura 6.6: Error cuadrático medio para 8 PMUs.

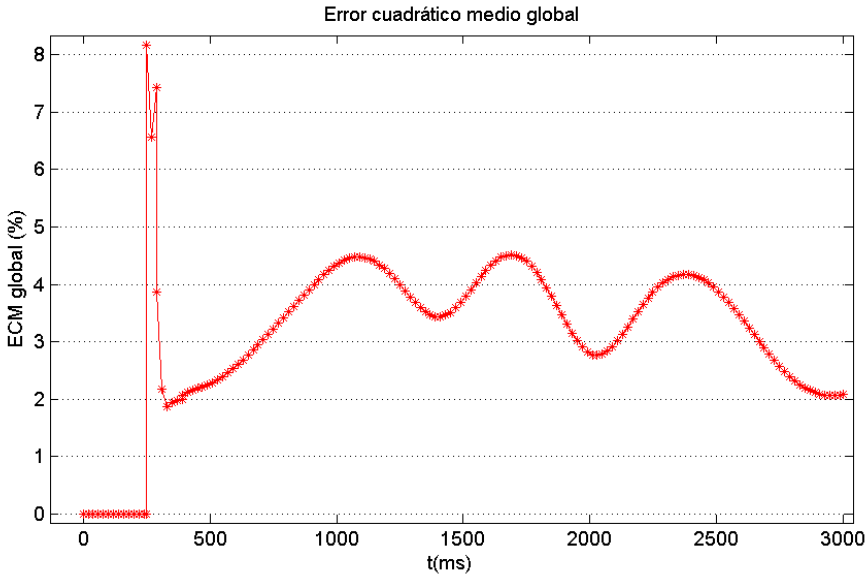


Figura 6.7: Error cuadrático medio para 4 PMUs.

la red y particularmente la matriz Ybus cambia, por ende el modelo que utiliza el algoritmo no es correcto. Por este motivo es que los errores en la estimación son mayores durante este intervalo, ya que se está excediendo el alcance del algoritmo.

6.2. Análisis de tiempo de ejecución

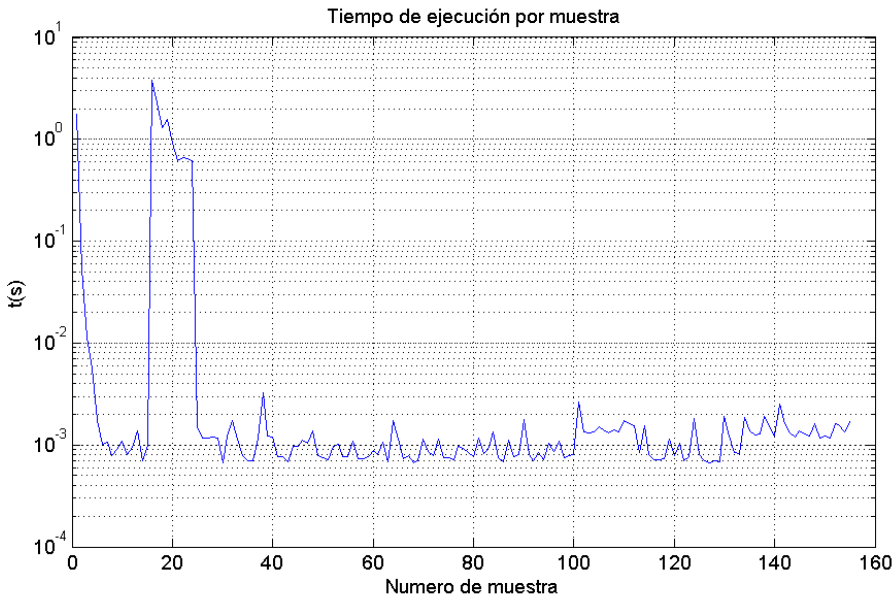
Es deseable que el algoritmo se ejecute a una velocidad suficiente para ser capaz de analizar los paquetes de sincrofasores a una tasa de 50 paquetes por segundo. Esta es la mayor tasa de reporte requerida por el *standard*, bajo la frecuencia nominal del sistema en 50Hz.

A los programas en Matlab y en C++ se le han agregado rutinas para almacenar los tiempos de ejecución de cada ciclo. Estos miden el tiempo en milisegundos desde que un paquete de muestras arriba al sistema, y el mismo es computado entregando las magnitudes en barras no-observables. Esto comprende el proceso de determinar si hubo o no una falla en la red, y el posterior procesado de la información.

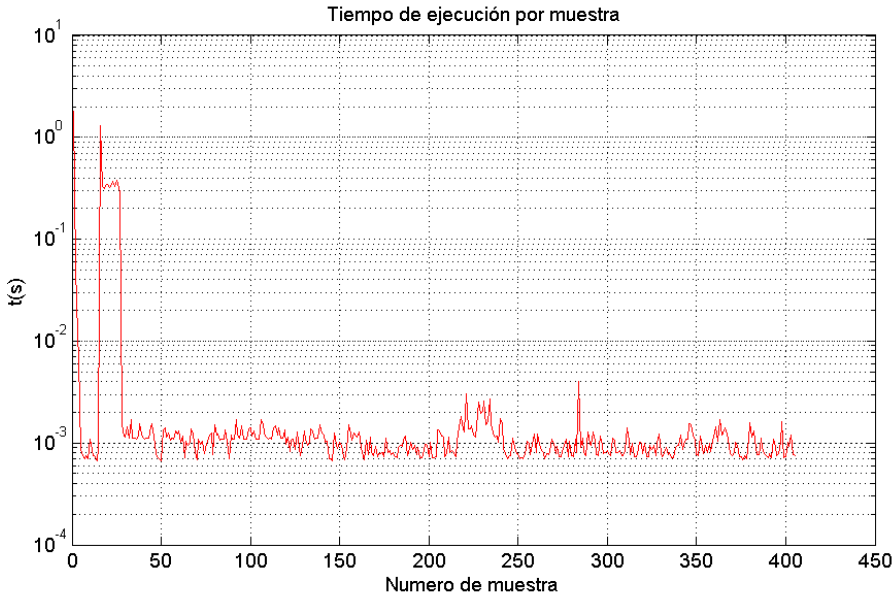
Para el análisis de la *performance* en Matlab, en la figura 6.8 se muestran las gráficas que corresponden al tiempo de procesamiento que fue requerido para computar adecuadamente cada paquete de sincrofasores (computadora I5-3337 @1.8GHz). El eje de abscisas numera cada muestra arribada al sistema, siendo la primera muestra que arriba al sistema identificada con el número uno, y así sucesivamente. El eje de ordenadas presenta el tiempo en segundos consumido para analizar dicho paquete de muestras. Con estas gráficas podemos ver el desempeño frente a diversas simulaciones, todas realizadas bajo la misma configuración de PMUs y varianzas.

Se observa que varias muestras requieren de un tiempo de procesado de algunos segundos. Estas coinciden en haber sido procesadas por el camino no lineal. Como era esperable, el camino no lineal consume varias veces más tiempo de procesamiento que el camino lineal. Esto muestra la necesidad de recortar en algunos ordenes de magnitud estos tiempos, un problema cuya solución se expondrá más adelante, al reprogramar el algoritmo en un lenguaje cuya ejecución sea más veloz que Matlab. Estos tiempos de ejecución serían inaceptables si se quisiese implementar el algoritmo de Matlab en un sistema real, pero para el análisis del desempeño del algoritmo, en Matlab resulta muy sencillo extraer gráficas e información asociada al proceso, como por ejemplo este análisis de *performance*.

Observando las muestras con menor tiempo de procesado, coincide que estas corresponden a paquetes cuyo procesamiento fue por el camino lineal, es fácil ver que todas ellas parecen haber sido ejecutadas a la misma velocidad. Este comportamiento es esperable. Recuerde que este camino lineal consta básicamente de dos partes: primero el proceso para determinar si hubo o no un cambio en el sistema, y en segundo lugar una serie de operaciones que ordenan vectores y multiplican matrices, donde todas las dimensiones están definidas a priori según la cantidad de barras observables y no observables por PMUs del sistema. Estas operaciones desde el punto de vista del procesador son una cantidad finita y determinada de operaciones a ejecutar, por lo que es de esperar que el tiempo insumido para su



(a) Cortocircuito



(b) Cambio de carga

Figura 6.8: Tiempo de ejecución en Matlab sobre dos simulaciones.

ejecución se mantenga casi constante. Comparando algunos números, para determinada muestra cuyo procesamiento fue por el camino lineal, Matlab requirió 400 micro segundos para determinar si hubo un cambio (devolviendo en este caso la ausencia de cambio para el margen especificado), y 950 micro segundos para rea-

lizar el resto de las operaciones. Esta *performance* es adecuada para la ejecución en tiempo real a 50 muestras por segundo si así se quisiese.

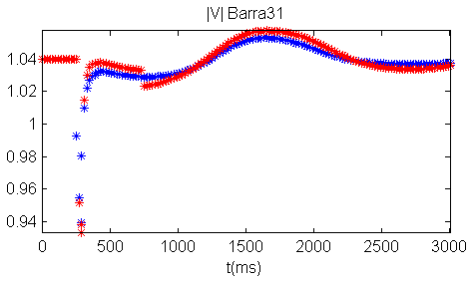
Realicemos el mismo análisis pero para muestras que fueron procesadas por el camino no lineal. A golpe de vista se perciben dos cosas muy importantes: gasta mucho tiempo y este tiempo no siempre es constante. El principal factor para este desempeño es quizás la naturaleza iterativa del proceso, donde a priori no es posible saber la cantidad de iteraciones que requerirá el computador para alcanzar (o no) la convergencia. Aparece por primera vez el compromiso en la relación tiempo-precisión. Ejecutando diversas pruebas bajo una misma simulación se intenta llegar empíricamente a una calibración para dicha relación, entre el margen de convergencia elegido y la cantidad de iteraciones realizadas. Muchas veces, para un margen de convergencia adecuado, este se alcanza en una sola iteración, y se comprueba que no por aumentar la precisión de la iteración y por ende el tiempo consumido, se mejora de forma notoria la estimación. La figura (6.9) detalla algunas pruebas realizadas en función del margen para iterar y cantidad de iteraciones realizadas, y en ella se muestran las estimaciones obtenidas. El lector notará que las tres estimaciones parecen ser la misma gráfica repetida tres veces, la realidad es que no varía en forma sensible la calidad de estimación al modificar este parámetro. La explicación reside en que hay factores que introducen errores de mayor magnitud que no dependen de la cantidad de iteraciones. El principal factor es la poca información que se tiene para realizar la estimación.

La conclusión es que dentro del proceso iterativo del camino no lineal, no hace falta realizar múltiples iteraciones, sino que en unas pocas el resultado obtenido es adecuado. Por lo menos esto aplica para las simulaciones realizadas en el marco de este proyecto.

Más allá de las iteraciones necesarias en este proceso, se le agrega dificultad de cómputo adicional en el momento en que es necesario invertir matrices en un proceso donde muchas de estas matrices son dispersas y tan grandes como la cantidad de barras o líneas tenga la red. El proceso que consume más tiempo del camino no lineal es la función llamada *Procesado SCADA-PMU*, que es la encargada de realizar la estimación propiamente dicha y cuyo tiempo de ejecución es mayor incluso que todas las demás funciones juntas. Aquí es necesario para cada iteración realizada calcular e invertir la matriz jacobiana de la función de transferencia.

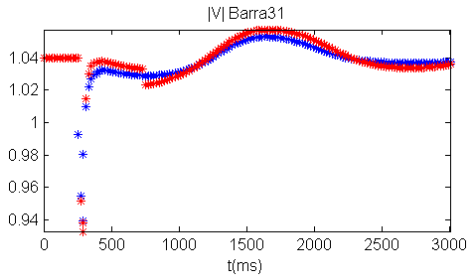
A modo de ejemplo y para llevar los tiempos a números, analicemos los tiempos que le toma a Matlab procesar una muestra que fue computada por el camino no lineal. La primer parte del proceso es común a los dos caminos, es determinar si hubo un cambio en la red, el cual insume 400 micro segundos. El resto del camino no lineal insume aproximadamente 1 segundo, y algunas veces se observó que alcanza los 3 segundos. Por este motivo es necesario cambiar de lenguaje de programación.

(a) Margen de iteración: 0,001



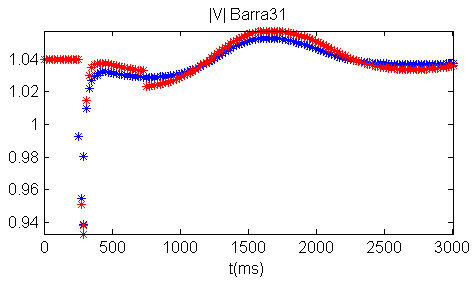
(b) Margen de iteración: 0,001

(c) Margen de iteración: $1{,}0x10^{-8}$



(d) Margen de iteración: $1{,}0 \times 10^{-8}$

(e) Margen de iteración: 0,1



(f) Margen de iteración: 0,1

Figura 6.9: Tiempo de ejecución en Matlab frente a distintos márgenes de iteración y su estimación asociada.

Un área todavía no analizada es saber cuánto le toma al software ingresar la información. Aquí se debe tomar en cuenta el tiempo que lleva leer los datos provenientes del SCADA para la configuración inicial y también lo que es muy importante, leer los datos provenientes de los PMUs, y procesarlos en una capa tal que los entregue al software del algoritmo en la forma adecuada. El análisis detallado de estos tiempos exceden el alcance del proyecto.

Ahora nos enfocaremos en la solución al problema de la *performance* de Matlab. Es necesario llevar el algoritmo implementado en Matlab a un lenguaje cuyo tiempo de ejecución sea de varios órdenes menor [8]. El principal lenguaje compilado

para generar algoritmos de gran velocidad de ejecución es C++, este es quien presenta mejores tiempos que sus principales competidores, Java, Go y Scala. La referencia presentada concluye que C++ es quién obtiene mejores tiempos de ejecución, destacándose aún más cuando el código se encuentra optimizado.

Dada la estructura de bloques implementada en Matlab, se genera cierta facilidad para migrar el código entre lenguajes. Basta llevar bloque a bloque todas las rutinas de Matlab a C++, respetando los datos de entrada y salida en cada bloque, y que en su interior el algoritmo de procesado de datos sea igual al de Matlab. Visto desde este punto, como funciones que son cajas negras, si se cumple que ante una misma entrada se obtiene la misma salida entre Matab y C++, se concluye que la función fue bien migrada. Finalmente, si todos los bloques de funciones fueron exitosos en C++, la concatenación de estos generará el código para el algoritmo. Solo basta esperar que cada bloque se ejecute en C++ a mayor velocidad que en Matlab para alcanzar el objetivo.

Realicemos un análisis de tiempo de ejecución. Corremos un caso de prueba en Matlab, cortocircuito trifásico en línea 26-29 (caso de prueba A), y graficamos el tiempo de ejecución que fue necesario para procesar cada muestra (figura 6.10). Las muestras que requirieron mayor tiempo de procesamiento son: muestra 1 y muestras de 16 a 41. Como fue tratado previamente, estas muestras corresponden a situaciones en las que el algoritmo realizó el procesamiento mediante el camino no lineal. Nos enfocamos en el análisis de las muestras posteriores a la primera. Ahora comparemos cuanto le toma a C++ procesar alguna estas muestras.

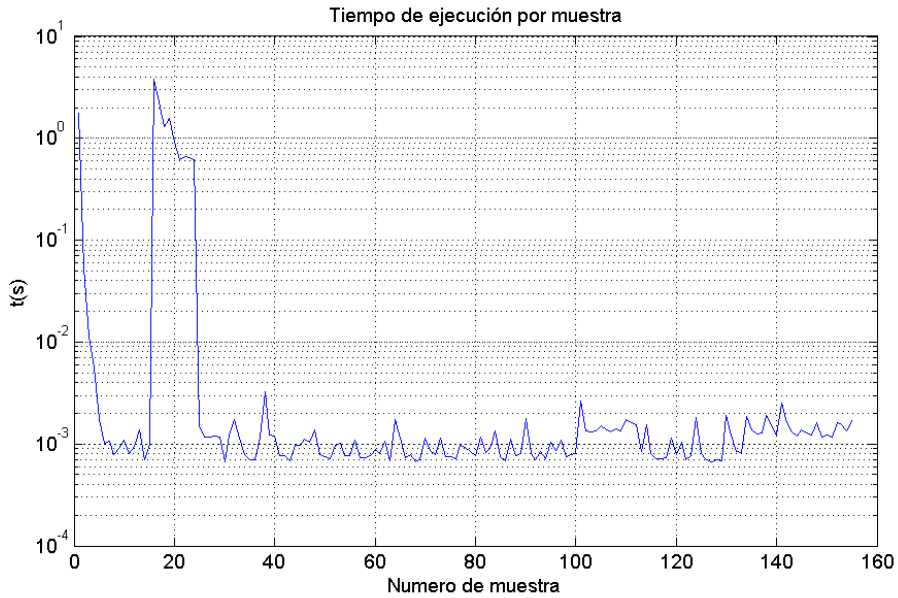


Figura 6.10: Tiempos de ejecución por muestra en simulación A

Tomemos la muestra 16, la cual corresponde al comienzo de la falta, que es donde el sistema más debe procesar para estimar. Mantenemos el margen de iteración suficientemente grande como para que el sistema no requiera realizar más de una iteración.

En un *Pentium Dual Core* a Matlab le consume 4 segundos procesar dicha muestra, y en C++ fueron necesarios 90 milisegundos. Si bien la diferencia aquí entre Matlab y C++ es muy grande, todavía no se alcanza la meta de 20 milisegundos. La primera prueba para bajar el tiempo es ganar velocidad aumentando la capacidad del hardware.
Tomamos cinco computadoras a nuestro alcance para ver este mismo tiempo de ejecución. Los resultados se expresan en la siguiente tabla:

Procesador	Vel. Procesador (GHz)	Tiempo C++ (ms)
Intel i7	4.6	24
Intel i5	3.10	43
Intel Core 2 Duo	3.00	51
Intel i7	2.4	54
Intel Pentium Dual Core	1.73	90

Aquí vemos que con la implementación actual del algoritmo, y con un ordenador comercial es posible acercarse al objetivo planteado. En el capítulo conclusiones se enumeran posibles mejoras a realizar para reducir estos tiempos.

Capítulo 7

Conclusiones

7.1. Conclusiones

En el transcurso de los meses que se dedicaron a la búsqueda bibliográfica nos encontramos con que esta temática es relativamente reciente, todos los años se publican artículos presentando metodologías novedosas, dentro de las cuales se encuentra la implementada que fue presentada en 2012. Al momento de finalizar la búsqueda bibliográfica no se encontraron artículos que presentasen soluciones definitivas al problema, por lo cual podemos decir que la temática se encuentra en plena etapa de investigación.

Respecto a la implementación en Matlab, el algoritmo es muy sensible respecto a los parámetros de entrada. Es necesario ajustar la configuración que mejor se adecue a la red a estimar. Los resultados obtenidos están dentro de lo esperado, se mantienen dentro de un rango de aceptación adecuado respecto a la relación calidad de la estimación contra cantidad de unidades PMU desplegadas en la red. En todos los casos se pudo seguir la forma de onda, la falta nunca pasó inadvertida. La estimación en las fases consideramos que fue mucho más acertada que en los módulos. Esto es beneficioso para el seguimiento de ángulos de generadores. Se obtuvo con la migración del algoritmo a C++ una reducción en el tiempo de ejecución, y con la computadora adecuada que se puede ejecutar a 50 Hz en una red de dimensiones similares. Se concluye que la migración de código fue exitosa y se cumplen los objetivos esperados. En el lapso donde ocurre la falta de cortocircuito la topología de la red cambia. Durante este período el algoritmo queda fuera de hipótesis. Destacamos que para los casos observados en ese lapso la estimación no se dispara a valores muy lejanos, y cuando la falta desaparece, la estimación vuelve a engancharse.

7.2. Posibles mejoras

En el momento de cierre del proyecto quedan pendientes algunas ideas a aplicar para mejorar el algoritmo. Desde el punto de vista del algoritmo:

- Investigar ingresar información adicional al sistema para mejorar la estimación, por ejemplo, conocer el estado de los interruptores.

- Mejorar la forma en que se introducen los estados previos de la red.

- Agregarle complejidad a las decisiones del algoritmo, por ejemplo, en la determinación de cambios poder discernir que evento fue el que ocurrió, y en base a esto tomar distintas acciones para mejorar la estimación.

Referente a la implementación:

- Mejorar las inversiones de matrices dispersas, tanto en Matlab como en C++.

- Optimizar el código, especialmente en C++, para bajar el tiempo de ejecución.

Apéndice A

Más resultados

A.1. Evaluación de la precisión de la estimación según niveles y grupos

En el presente capítulo se ilustran y detallan más resultados que complementan los ya vistos. Se verá que dichos resultados son de variada índole. Recordar que existen dos clasificaciones definidas en el capítulo *Resultados*. Estas clasificaciones agrupan las barras según la distancia al evento simulado, ya sea cortocircuito o cambio de carga (clasificación en grupos), y según distancia a las PMU (clasificación en niveles).

Los casos que se van a estudiar son: una barra pegada al evento y con una PMU instalada en ella, y el caso opuesto: una barra lejos del evento y lejos de las PMUs. Otro parámetro relevante es la lejanía de una PMU al punto de mayor cambio en el sistema (este parámetro es independiente de los grupos y niveles). En los ensayos se tuvo la precaución de dejar fija la distancia entre el evento simulado y la PMU más cercana.

El conjunto de casos a estudiar se puede resumir en la siguiente tabla:

	Nivel 1	Nivel 2	Nivel 3
G1	**Caso 1**		
G2		**Caso 2**	
G3			**Caso 3**

Para realizar el análisis se define un nuevo indicador global de error que es individual a cada barra, este indicador es diferente al definido en la ecuación (5.14) (ECMi). El viejo indicador resume el error global entre todas las barras para determinado instante k. En el nuevo indicador solo participa una barra, y lo que se realiza es un promedio del error en esa barra en el correr del tiempo. Este nuevo indicador será útil para poder comparar barras contra barras en diferentes circunstancias, y se define para la barra i como:

$$EG_i = \frac{100}{m} \sum_{k=1}^{m} |\vec{V}^k_{i\,estimado} - \vec{V}^k_{i\,real}|, \tag{A.1}$$

donde m es la cantidad de muestras de la estimación.

En las siguientes secciones se detallarán las pruebas realizadas, la barra analizada y a que grupo y nivel corresponde.
La configuración del algoritmo es en todos los casos igual y los lugares de las PMU también permanecen fijos. Estas configuraciones se indican a continuación:

- Caso de prueba: Cambio de carga en la barra 27 (Caso D).
- Margen de cambio: 0.3.
- Varianza G1: 100.
- Varianza G2: 100.
- Varianza G3: 1.
- Varianza G4: 0.01.
- Varianza Priori: 0.5.
- Lugar de las PMU: 1, 8, 13, 15, 20, 23, 26, 29.

En la figura A.1 se puede observar con círculos el lugar de las PMU y con una lupa las barras monitoreadas. La nomenclatura N1G1, N2G2, N3G3 significa nivel y grupo de la barra respectivamente. Las barras seleccionadas para estos tres casos son:

- Barra 27 para caso 1, resultado en figura A.2.
- Barra 17 para caso 2, resultado en figura A.3.
- Barra 18 para caso 3, resultado en figura A.4.

Análisis de los Resultados

Los resultados obtenidos se pueden comparar utilizando el indicador EG_i definido en la sección anterior, que para los casos estudiados es:

	Nivel 1	Nivel 2	Nivel 3
G1	**0.7663**		
G2		**0.09466**	
G3			**0.0389**

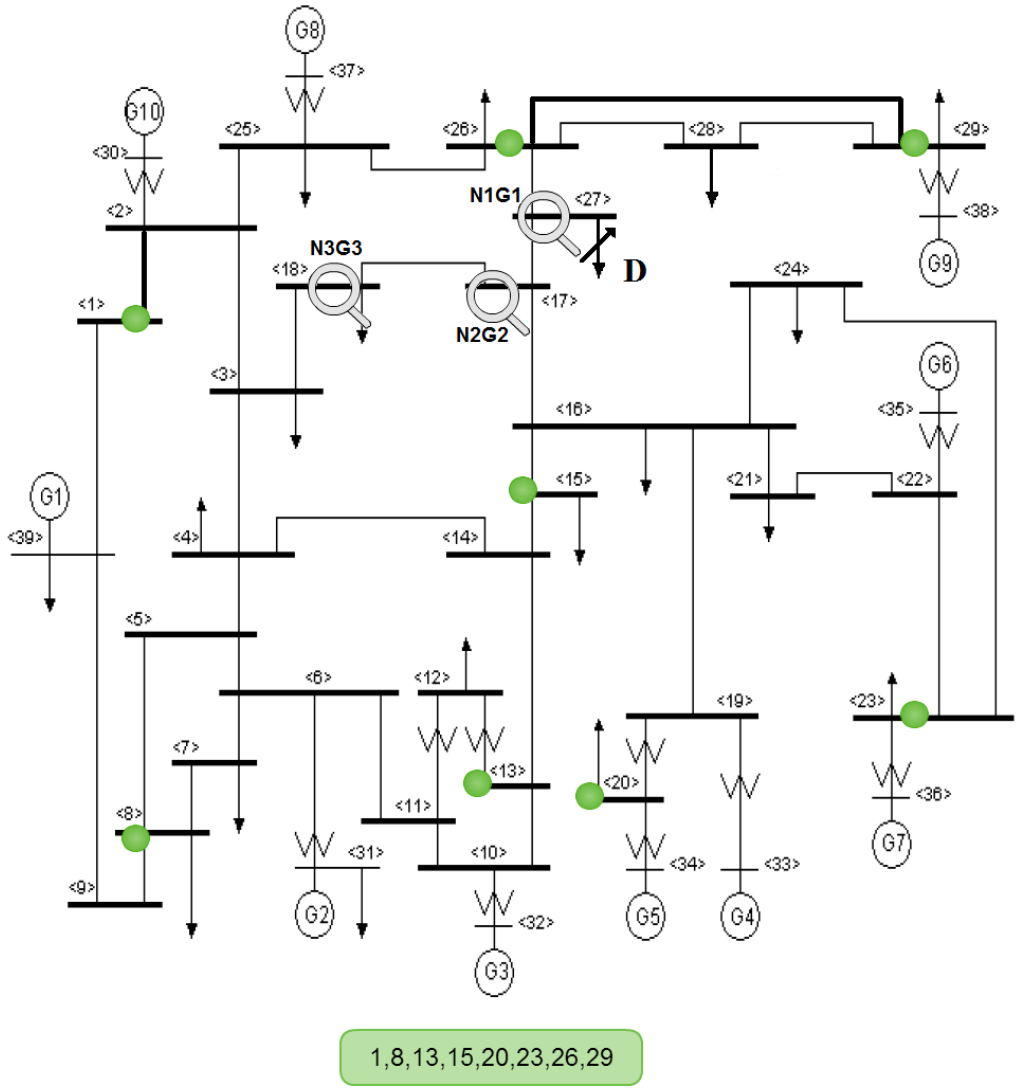


Figura A.1: Disposición de las PMUs y barras monitoreadas

El primer resultado notable se observa en el caso de la barra N3G3. La misma tiene dos propiedades muy dispares, está lejos de la barra donde ocurrió el cambio de carga y por otro lado se encuentra lejos de una PMU. Es decir, por un lado se espera que no tenga grandes variaciones por encontrarse lejos de la barra 27, y por el contrario la misma no tiene una observabilidad óptima como para que el algoritmo pueda seguir los cambios que se produzcan en ella. Pese a esto, los resultados obtenidos para la misma son muy buenos, incluso mejores que para la barra N1G1. Se hace énfasis en que no es posible realizar una conclusión generalizada que afirme que cuando una barra está lejos de la falta, la misma tendrá errores bajos. Eso depende de la red, de la falta y de la cercanía de una PMU a la falta.

Por otro lado, se observa que en la barra N1G1 se tienen los mayores errores. Ésta es la que tiene una PMU y al cambio de carga más próximos. Se puede concluir

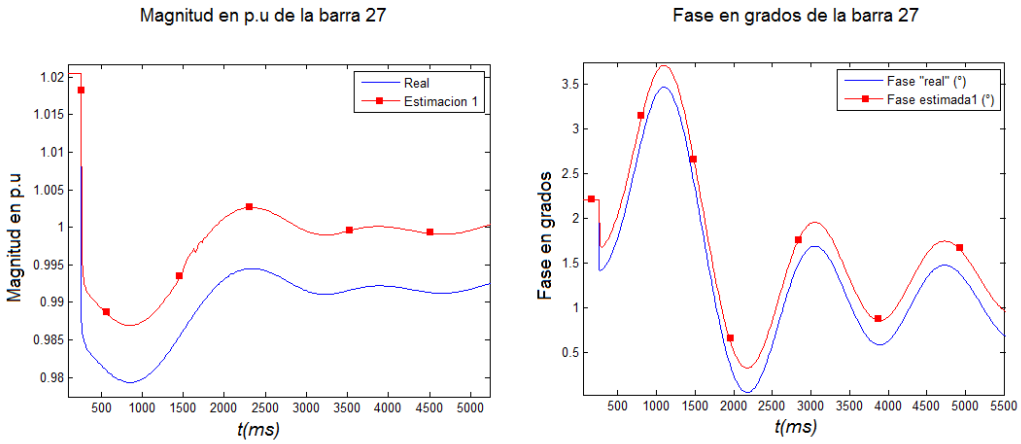


Figura A.2: Caso 1, Barra 27, Grupo 1 y Nivel 1

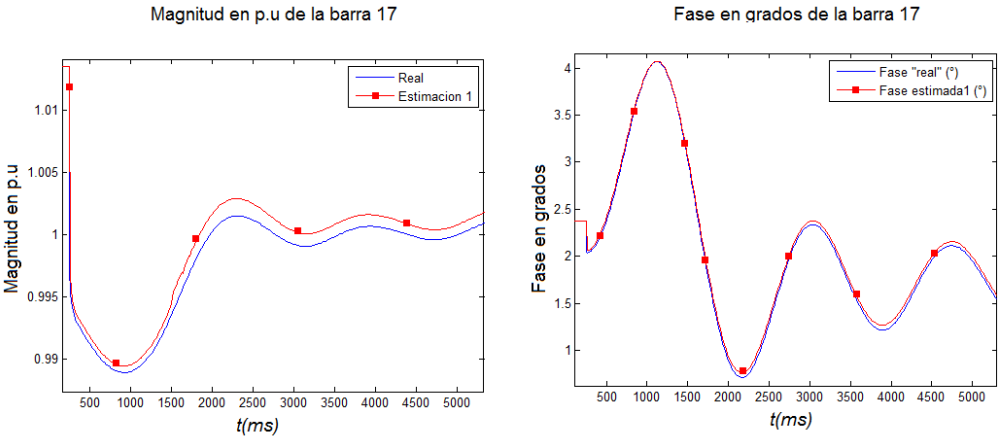


Figura A.3: Caso 2, Barra 17, Grupo 2 y Nivel 2

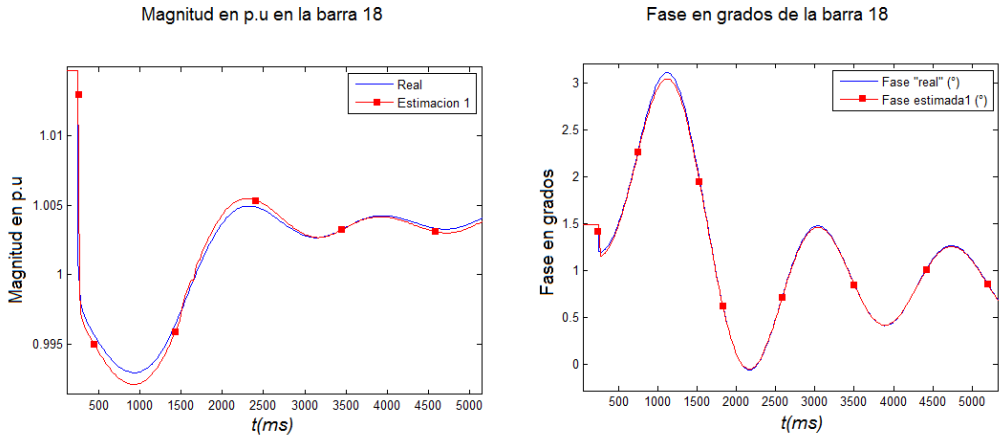


Figura A.4: Caso 3, Barra 18, Grupo 3 y Nivel 3

entonces que una barra nivel 1 no siempre debe tener los errores más bajos de la estimación.

A.2. Evaluación de la precisión de la estimación según distancia de PMUs a la falta

En la sección previa se mantuvo fija la observabilidad del evento con respecto a una PMU. Esto significa que la barra pegada al evento siempre mantuvo una distancia fija a una PMU en todos los casos. En esta sección se realizará un estudio de cómo afecta alejar las PMU del lugar del evento ocurrido. Para realizar esto se utilizará el caso de prueba B (corto circuito en la línea que une las barras 1 con 2).

En la figura A.5 se observan los casos llevados a la práctica. Como se observa, se realizarán dos pruebas, la primera tiene una PMU en la barra 2, esto significa que la falta tiene una PMU pegada (nivel de observabilidad 0). En el caso de la derecha, se observa que la PMU es cambiada de lugar hacia la barra 25, de esta manera la falta queda a nivel 1. Por otro lado, en la figura se observa que la barra analizada es la 14, que es nivel 2 en ambos casos de prueba.

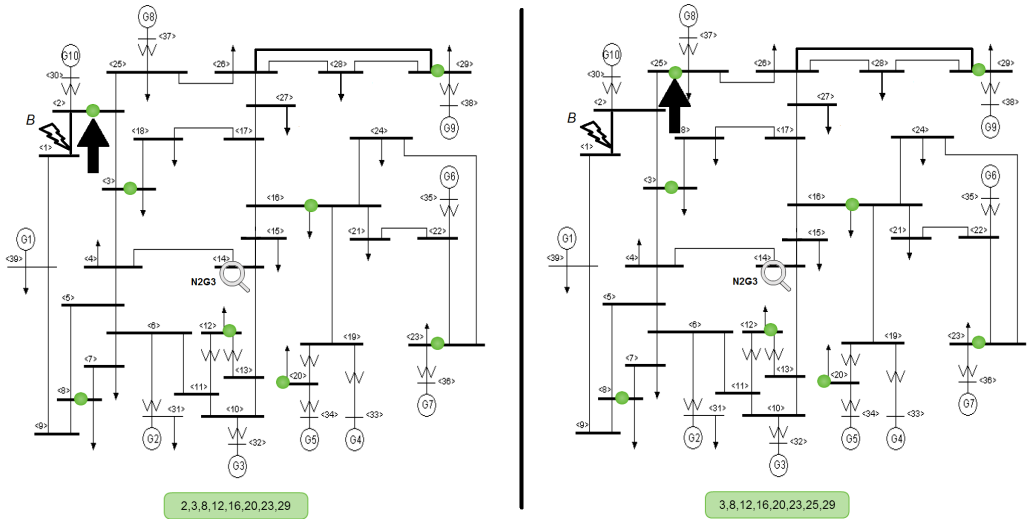


Figura A.5: izquierda: Barra en falta Nivel 0 , Derecha: Barra en falta Nivel 2

Los resultados obtenidos para las pruebas serán resumidos con el indicador EG_i, y se pueden visualizar en la siguiente tabla:

Barra	Prueba 1 (falta a nivel 0)	Prueba 2 (falta a nivel 1)
Barra 14	**0.1232**	**0.1791**

En este caso simulado se puede concluir que tener PMUs cerca de la falta ayuda en la estimación en barras que puedan estar lejos del lugar de la falta.

A.3. Análisis de las varianzas en la estimación no lineal

El archivo de configuración del algoritmo permite asignar los valores de varianza para cada grupo generado. Dada la red y la falta, automáticamente se generan los grupos o áreas (capítulo 3.4). Asignarle una varianza a un grupo significa que en el método de mínimos cuadrados ponderados las medidas provenientes de SCADA para esta barra van a tener menos o más influencia en el resultado. Por ejemplo, una barra que se encuentra lejos del evento, pertenece al grupo G3, y se le asigna una baja varianza (una alta confianza). Esto es porque en esta barra lejana los datos de SCADA que provienen de resolver un problema estático siguen siendo válidos, ya que esa zona no se ve tan afectada por el cambio ocurrido. Cuanto mayor es la varianza, el dato SCADA asociado tiene menor influencia en el resultado final.

El objetivo de esta sección es comparar los resultados de realizar modificaciones en las varianzas de la matriz R. Para ello se fuerza al algoritmo a realizar siempre la estimación no-lineal (capítulo 3.3), ya que dicha matriz solo se utiliza en esta etapa.

Para este análisis se utilizó el *caso de prueba B*. Las PMU están ubicadas en las barras:

3 8 13 16 20 23 25 29

A.3.1. Caso de Prueba

La primera comparación se hará con las configuraciones de varianza que se detallan en la siguiente tabla:

Campo	**Nota**	**Caso1**	**Caso2**
Varianza G1	Barras en la Falta	100	0.01
Varianza G2	Barras no-obs. y conectadas a G1.	100	0.01
Varianza G3	Barras no-obs. lejanas.	1	1
Varianza G4	Barras obs.	0.01	0.01
Varianza a Priori	Barras de Generación.	0.5	0.5

Se presentarán gráficas de la tensión de la barra 2. Las características de esta barra son:

- Nivel de observabilidad 1.
- Grupo de varianza G1.

En las gráficas A.6 y A.7 se observa la magnitud y la fase en la barra 2 en el momento del cortocircuito.

Análisis de los Resultados:

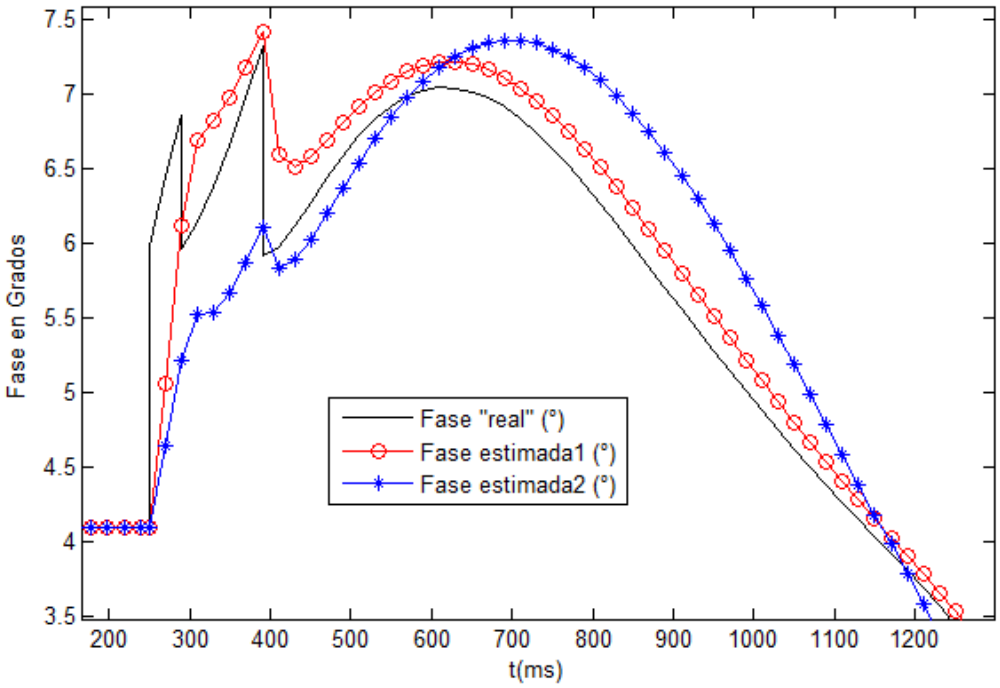


Figura A.6: Fase en la barra 2

Los resultados son los esperados, la estimación en el caso 2 no sigue la forma de onda tan bien como lo hace la estimación del caso 1. El motivo de esto es que en el caso 2 se le dio igual confianza a la vieja medida aportada por el SCADA que a la medida aportada por el PMU en ese instante, y la solución tiende a pegarse al estado previo a la falta. Por lo contrario, en el caso 1 se le asigna mayor importancia al dato de PMU y la estimación está más cercana a la realidad.

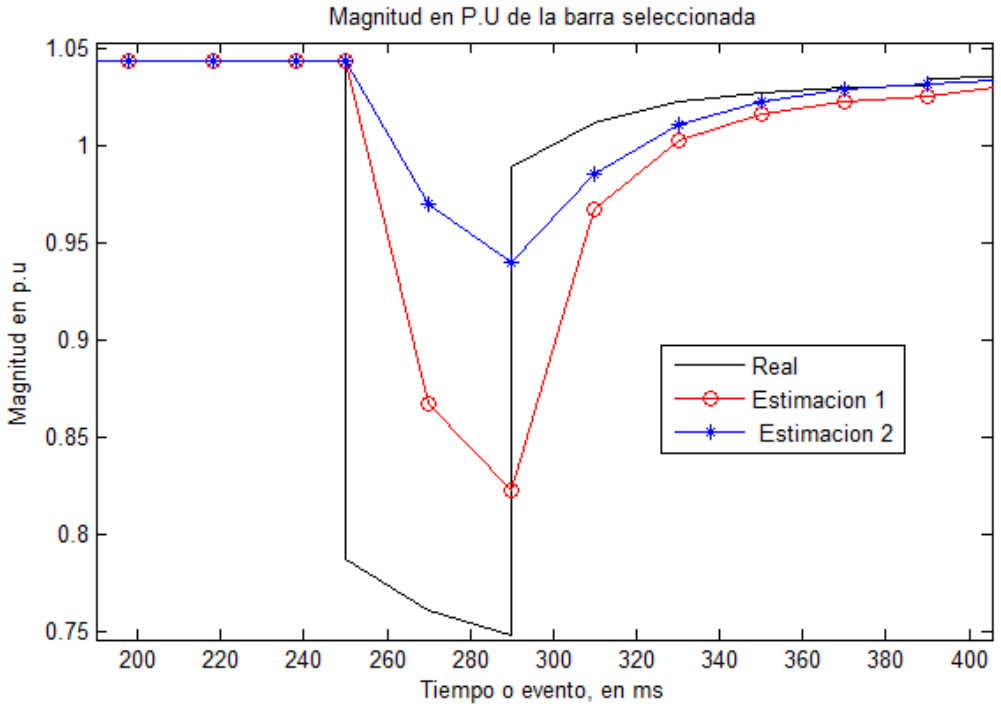


Figura A.7: Magnitud en la barra 2

Referencias

[1] Chapra, Steven y Raymond P. Canale: *Métodos numéricos para ingenieros*. McGraw-Hill, 2007, ISBN 9789701061145. `http://www.mhhe.com/chapra`.

[2] Costa, A. Simões y André Albuquerque: *A two-stage orthogonal estimator to incorporate phasor measurements into power system real-time modeling*. En *Power Systems Computation Conference (PSCC)*, 2011.

[3] Costa, A. Simões, André Albuquerque y Daniel Bez: *An estimation fusion method for including phasor measurements into power system real-time modeling*. IEEE Transactions on Power Systems, 28(2):1910–1920, 2013.

[4] Das, Kaushik, Jagabondhu Hazra, Deva P. Seetharam, Ravi Kiran Reddi y Avinash K. Sinha: *Real-time hybrid state estimation incorporating SCADA and PMU measurements*. En *Innovative Smart Grid Technologies (ISGT Europe), 2012 3rd IEEE PES International Conference and Exhibition on*, páginas 1–8. IEEE, 2012.

[5] Fernandez, Rafael Guzmán: *Sistemas SCADA en Distribución de Energía Eléctrica*, 1993. `http://bibdigital.epn.edu.ec/bitstream/15000/7019/1/T64.pdf`, Tesis de especialización de sistemas eléctricos de potencia.

[6] Gómez-Expósito, Antonio y Antonio de la Villa Jaen: *Two-level state estimation with local measurement pre-processing*. Power Systems, IEEE Transactions on, 24(2):676–684, 2009.

[7] HAIM, Prof. Ing. Isi: *Cartilla de Fórmulas curso de Redes Eléctricas de la Facultad de Ingeniería de la UDELAR*, 2014. `https://eva.fing.edu.uy/mod/folder/view.php?id=32403`, Internet.

[8] Hundt, Robert: *Loop Recognition in C++/Java/Go/Scala*, 2011. Loop Recognition PDF.

[9] IEEE: *New England test system, IEEE 39 Bus System*. 39 Bus System files.

[10] IEEE: *IEEE Standard for Synchrophasor Data Transfer for Power Systems*. IEEE Std C37.118.2-2011 (Revision of IEEE Std C37.118-2005), páginas 1–53, Dec 2011.

[11] IEEE: *IEEE Standard for Synchrophasor Measurements for Power Systems*. IEEE Std C37.118.1-2011 (Revision of IEEE Std C37.118-2005), páginas 1–61, Dec 2011.

[12] IEEE: *IEEE Standard for Synchrophasor Measurements for Power Systems – Amendment 1: Modification of Selected Performance Requirements*. IEEE Std C37.118.1a-2014 (Amendment to IEEE Std C37.118.1-2011), páginas 1–25, April 2014.

[13] Kezunovic, M.: *Application of Time-synchronized Measurements in Power System Transmission Networks*. Kluwer international series in engineering and computer science: Power electronics & power systems. 2014, ISBN 9783319062181. http://books.google.com.uy/books?id=J9AkBAAAQBAJ.

[14] Kostyniak, Ted: *PSS/E – Four Decades and Still Growing* . PSS/E overview.

[15] MATLAB: *version 7.10.0 (R2010a)*. The MathWorks Inc., Natick, Massachusetts, 2010.

[16] Pena, Pablo y Ricardo Franco: *Taller introducción al uso de Sincrofasores en sistemas Eléctricos de potencia. CUGRE/CIGRÉ, Montevideo, Uruguay*, 2013. http://www.cugre.org.uy/index.php/uruguay?layout=blog.

[17] Peng, Chunhua y Xuesong Xu: *A hybrid algorithm based on BPSO and immune mechanism for PMU optimization placement*. En *Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on*, páginas 7036–7040, June 2008.

[18] Penin, A.R.: *Sistemas SCADA*. Marcombo, 2007, ISBN 9788426714503. http://books.google.com.uy/books?id=32kgCNG34TwC.

[19] Romero Morales, Cristóbal y Carlos de Castro Lozano: *Introducción a SCADA*, 2012. http://www.uco.es/grupos/eatco/automatica/ihm/descargar/scada.pdf, Internet.

[20] Rosales, Dr. Ing Andrés: *Presentación: Sistemas de Control Distribuido y SCADA*, 2012. http://es.slideshare.net/androsaco/dcs-scada-rev, Maestria Automatizacion y Control Industrial.

[21] Simões Costa, A. y V.H. Quintana: *An orthogonal row processing algorithm for power system sequential state estimation*. Power Apparatus and Systems, IEEE Transactions on, (8):3791–3800, 1981.

[22] Soni, Sandeep, Sudhir Bhil, Dhirendra Mehta y Sushama Wagh: *Linear state estimation model using phasor measurement unit (PMU) technology*. En *CCE*, páginas 1–6, 2012.

[23] Stroustrup, B.: *The C++ Programming Language*. Always learning. Addison-Wesley, 2013, ISBN 9780321563842. http://books.google.com.uy/books?id=0klsAQAAQBAJ.

[24] Yang, X., X. Zhang y S. Zhou: *Coordinated algorithms for distributed state estimation with synchronized phasor measurements.* Applied Energy, 96:253–260, 2012.

[25] Zhou, M. y V. A. Centeno: *An Alternative for Including Phasor Measurements in State Estimators.* IEEE Transactions on Power Systems, 21(4):1930–1937, 2006.

[26] Zhou, Ming: *Advanced system monitoring with phasor measurements.* Tesis de Doctorado, Virginia Polytechnic Institute and State University, 2008.

[27] Zima, Marek y Marija Bockarjova: *Operation, monitoring and control technology of power systems.* EEH Power Systems Laboratory, ETH Zurich,[Online]. Available: http://www. eeh. ee. ethz. ch, Accessed March, 2007.

Esta es la última página.
Compilado el jueves 30 octubre, 2014.
http://iie.fing.edu.uy/