



UNIVERSIDAD DE CHILE  
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS  
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

MÉTODOS AVANZADOS PARA LA EVALUACIÓN DEL DESEMPEÑO DE  
ALGORITMOS DE ESTIMACIÓN Y PRONÓSTICO BASADOS EN MÉTODOS  
SECUENCIALES DE MONTE CARLO

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN CIENCIAS DE LA  
INGENIERÍA, MENCIÓN ELÉCTRICA

ISMAEL SEBASTIÁN JARAS CASTAÑOS

PROFESOR GUÍA:  
MARCOS ORCHARD CONCHA

MIEMBROS DE LA COMISIÓN:  
JORGE SILVA SÁNCHEZ  
MILAN DERPICH MUSA

Este trabajo ha sido parcialmente financiado por Proyecto CONICYT FONDECYT  
1140774

SANTIAGO DE CHILE  
2017



RESUMEN DE LA TESIS PARA OPTAR  
AL GRADO DE MAGÍSTER EN CIENCIAS DE LA INGENIERÍA, MENCIÓN ELÉCTRICA  
POR: ISMAEL SEBASTIÁN JARAS CASTAÑOS  
FECHA: 2017  
PROF. GUÍA: SR. MARCOS ORCHARD CONCHA

MÉTODOS AVANZADOS PARA LA EVALUACIÓN DEL DESEMPEÑO DE  
ALGORITMOS DE ESTIMACIÓN Y PRONÓSTICO BASADOS EN MÉTODOS  
SECUENCIALES DE MONTE CARLO

Diagnosticar y pronosticar la evolución del estado de un sistema dinámico es un problema complejo que desde hace algún tiempo ha comenzado a tomar más fuerza y relevancia dentro de la ingeniería. En este sentido, la comunidad de *Prognostics and Health Management*, ha escogido a los algoritmos basados en métodos Secuenciales de Monte Carlo (SMC) como los algoritmos de facto del estado-del-arte, por su capacidad de aproximar el filtrado óptimo cuando se estudia un sistema no-lineal y con incertidumbre no necesariamente Gaussiana. Sin embargo, el costo a pagar por estos algoritmos es la complejidad computacional que requieren para generar sus resultados, lo que sugiere obtener versiones simplificadas de estos algoritmos -en desmedro de la calidad de los resultados- que puedan resolver el problema en tiempo real y con el objetivo de ser implementados en sistemas embebidos. De acuerdo con lo expresado, nace la necesidad de desarrollar un marco de trabajo que posibilite la comparación de algoritmos basados en el enfoque secuencial de Monte Carlo y que permita, por una parte, incluir las variaciones estadísticas inherentes de estos algoritmos, como también la implementación de medidas que incorporen la caracterización probabilística de la evolución del estado del sistema y, de esta forma, cuantificar la degradación en la calidad de los resultados producto de las simplificaciones.

Es común observar comparaciones entre diferentes aproximaciones al método secuencial de Monte Carlo mediante medidas como el MSE, o momentos de las distribuciones, pero, ¿logran estas medidas incorporar la descripción probabilística que se asume como caracterización de la evolución del estado?. En el presente trabajo de Tesis se aborda esta interrogante y se utiliza como punto de partida para el desarrollo de un método de comparación que desafía la forma actual en la que se contrastan las diferentes aproximaciones a SMC. El método propuesto incluye el desarrollo de un marco comparativo -mediante el análisis PACC- donde se toma en cuenta el comportamiento estocástico propio de los algoritmos basados en SMC, por lo que es posible estimar el máximo error teórico en el que incurren distintas aproximaciones. Además, para abarcar la descripción estadística que generan los algoritmos sobre la evolución del estado del sistema, se introducen medidas de la información para cuantificar el desempeño de distintas aproximaciones y, de esta manera, estimar el máximo error teórico incorporando toda la información sobre la descripción estadística del proceso.

En particular, para la implementación y análisis de la metodología propuesta en esta Tesis, se utiliza la estrategia comparativa para estudiar el desempeño de distintas aproximaciones de SMC al problema de estimar y pronosticar el estado de carga en baterías de Ion-Litio. Se muestra la factibilidad de obtener un algoritmo de referencia para ambos casos. Además, se concluye acerca del algoritmo más preciso computacionalmente, en cuanto a pérdida de información y representación de la incertidumbre.



*“The important thing is to not stop questioning. Curiosity has its own reason for existing.”*  
— Albert Einstein



# Agradecimientos

Gracias a todas las personas que me acompañaron y que he conocido, agradezo también todas las situaciones que ocurrieron en el transcurso de este trabajo y a lo largo de mi vida.

Quiero darle las gracias a mis amigos; Pancho, un hermano que me ha entregado la vida, a La Bred, a la FAE, a los amigos de la Universidad, a los del colegio y de la vida. Gracias por tantos momentos de expresión y por compartir sus perspectivas de la vida, información que siempre me ha enseñado a desafiar mis propias creencias.

Agradezco a mi familia. A mi abuelo quien me transmitió el amor por la ciencia y las matemáticas desde mis primeros años. También a mi abuela, a mi madre, mi padre y mi hermana, quienes siempre han confiado en mi y me han apoyado. Gracias a la Andreita, por acompañarme y siempre escucharme con tanta apertura, por enseñarme a concluir las mismas cosas, desde otros caminos.

Gracias a mis amigos del grupo de investigación, por su dedicación y determinación en el trabajo, por enseñarme y transmitir sus interpretaciones y conocimientos, por el trabajo anterior que han desarrollado; apoyo que posibilita continuar profundizando en el conocimiento. Al profesor Marcos Orchard, quien siempre mostró una actitud empática y acogedora, además de entregarme su orientación y apoyo a lo largo de este trabajo.





# Tabla de Contenido

<b>Índice de Tablas</b>	<b>xi</b>
<b>Índice de Ilustraciones</b>	<b>xii</b>
<b>1. Introducción</b>	<b>1</b>
1.1. Objetivos . . . . .	2
1.2. Hipótesis . . . . .	2
<b>2. Fundamentos Generales</b>	<b>3</b>
2.1. Cómputo Probablemente Exacto y Correcto . . . . .	4
2.2. Inferencia Bayesiana en Modelos ocultos de Markov . . . . .	5
2.2.1. Filtro de Kalman . . . . .	7
2.2.2. Unscented Kalman Filter . . . . .	7
2.2.3. Métodos Secuenciales de Monte Carlo . . . . .	8
2.2.4. Algoritmo secuencial de Monte Carlo genérico . . . . .	12
2.2.5. Suavizado en SMC . . . . .	14
2.3. Pronóstico . . . . .	14
2.3.1. Pronóstico: Basado en Monte Carlo Secuencial . . . . .	15
2.3.2. Pronóstico Mediante Filtro de Partículas: Regularización . . . . .	16
2.3.3. Pronóstico Mediante Filtro de Partículas con propagación de incertidumbre via sigma-points . . . . .	16
2.4. Lazo de Corrección Externa . . . . .	18
2.5. Medidas de Información . . . . .	18
2.5.1. Entropía . . . . .	18
2.5.2. Divergencia Kullback-Liebrer . . . . .	19
2.5.3. Estimación no-paramétrica de Medidas de Información . . . . .	19
2.6. Estimación de densidad de Kernel . . . . .	20
2.6.1. Selección del parámetro de suavizado . . . . .	20
2.7. Baterías de Ion-Litio . . . . .	21
2.7.1. Respresentación en Espacio-Estado . . . . .	22
<b>3. Metodología de Comparación para algoritmos basados en SMC</b>	<b>24</b>
3.1. Análisis PACC en Métodos Secuenciales de Monte Carlo . . . . .	25
3.2. Medidas de Información en Métodos secuenciales de Monte Carlo . . . . .	26
3.3. Estimación de Medidas de la Información en SMC . . . . .	27
3.4. Esquema Comparativo . . . . .	29
3.4.1. Procedimiento de Comparación . . . . .	30

3.5. Comentarios Sobre el Esquema Comparativo . . . . .	31
<b>4. Implementación de Esquema Comparativo en Algoritmos de Estimación y Pronóstico: Baterías de Ion-Litio</b>	<b>32</b>
4.1. Metodología comparativa en Estimación . . . . .	33
4.1.1. Datos y Parámetros del Modelo . . . . .	33
4.1.2. Algoritmo de Referencia para Estimación . . . . .	33
4.1.3. Algoritmos a Comparar . . . . .	35
4.1.4. Resultados de la comparación . . . . .	36
4.2. Metodología comparativa en Pronóstico . . . . .	43
4.2.1. Algoritmo de Referencia para Pronóstico . . . . .	43
4.2.2. Algoritmos a Comparar . . . . .	45
4.2.3. Resultados de la comparación . . . . .	46
4.3. Comentarios sobre la metodología . . . . .	56
<b>Conclusión</b>	<b>56</b>
<b>5. Bibliografía</b>	<b>59</b>
<b>Anexo</b>	<b>62</b>
<b>A. Tiempos de Procesamiento de Algoritmos</b>	<b>62</b>

# Índice de Tablas

4.1. Valores parámetros del modelo en espacio-estado para datos FUDS . . . . .	34
4.2. Máximo error empírico tomando el 90 % de los datos. 1060 realizaciones . . .	41
4.3. Máximo error empírico tomando el 70 % de los datos. 1060 realizaciones . . .	41
4.4. Máximo error empírico tomando el 50 % de los datos. 1060 realizaciones . . .	41
4.5. Máximo error empírico tomando el 30 % de los datos. 1060 realizaciones . . .	42
4.6. Máximo error empírico comenzando el pronóstico desde el 70 % de los datos. 1060 realizaciones . . . . .	52
4.7. Máximo error empírico comenzando el pronóstico desde el 50 % de los datos. 1060 realizaciones . . . . .	53
4.8. Máximo error empírico comenzando el pronóstico desde el 30 % de los datos. 1060 realizaciones . . . . .	53
A.1. Coeficientes polinomio grado 1 asociado a cada algoritmo para estimación . .	62
A.2. Coeficientes polinomio grado 2 asociado a PF Regularizado, considerando 3 perfiles de corriente distintos para pronóstico . . . . .	64
A.3. Coeficientes polinomio grado 2 asociado a PFSG, considerando 3 perfiles de corriente distintos para pronóstico . . . . .	64

# Índice de Ilustraciones

4.1.	Perfiles voltaje y corriente FUDS . . . . .	33
4.2.	Evolución de la esperanza del SoC, según distintas aproximaciones. . . . .	37
4.3.	Representación mediante pesos y partículas de la distribución del <i>SoC</i> y respectiva reconstrucción de pdfs mediante KDE. instante $k = 621$ . . . . .	38
4.4.	Evolución de la entropía para una realización del proceso de estimación, para cada algoritmo, por medio de estimación no paramétrica. $T_s = 10[seg]$ . . . .	39
4.5.	Evolución de la Divergencia para una realización del proceso de estimación, para cada algoritmo, por medio de estimación no paramétrica. $T_s = 10[seg]$ .	40
4.6.	Evolucion de la Esperanza del <i>SoC</i> para el algoritmo PF, considerando tres perfiles de corriente distintos: $Ir =$ Corriente Real , $Im =$ Corriente promedio y $mch =$ corriente Cadena de Markov. . . . .	47
4.7.	Evolucion de la Esperanza del <i>SoC</i> para el algoritmo PFSG, considerando tres perfiles de corriente distintos: $Ir =$ Corriente Real , $Im =$ Corriente promedio y $mch =$ corriente Cadena de Markov. . . . .	47
4.8.	Algoritmo PF. Izquierda: Reconstrucción de pdfs mediante KDE, para instante $k = 3275$ . Derecha: EoL (20%) según cada perfil de corriente. . . . .	49
4.9.	Algoritmo PFSG. Izquierda: Reconstrucción de pdfs mediante KDE, para instante $k = 3275$ . Derecha: EoL (20%) según cada perfil de corriente. . . . .	49
4.10.	Evolucion de la Entropia, según cada perfil de corriente para el algoritmo PF; una realización del pronóstico. $T_s = 10[seg]$ . . . . .	50
4.11.	Evolucion de la Entropia, según cada perfil de corriente para el algoritmo PFSG; una realización del pronóstico. $T_s = 10[seg]$ . . . . .	50
4.12.	Evolución de la Divergencia según cada perfil de corriente para el algoritmo PF; una realización del pronóstico. $T_s = 10[seg]$ . . . . .	51
4.13.	Evolución de la Divergencia según cada perfil de corriente para el algoritmo PFSG; una realización del pronóstico. $T_s = 10[seg]$ . . . . .	51
A.1.	Tiempo de procesamiento que demora el algoritmo en función de la cantidad total de instantes a estimar. Se consideran los 3 algoritmos utilizados en 4.1.3.	63
A.2.	Tiempo de procesamiento que demora el algoritmo en función de la cantidad total de instantes a pronosticar. Se considera el algoritmos PF Regularizado y los 3 perfiles de corrientes descritos en 4.2.2. . . . .	63
A.3.	Tiempo de procesamiento que demora el algoritmo en función de la cantidad total de instantes a pronosticar. Se considera el algoritmos PFSG y los 3 perfiles de corrientes descritos en 4.2.2. . . . .	64

# Capítulo 1

## Introducción

El desafío de cuantificar el desempeño de distintas aproximaciones al problema de estimación y pronóstico sobre la evolución del estado de un sistema aún es un tema no resuelto. En particular, al considerar los algoritmos de filtrado bayesiano basados en el método secuencial de Monte Carlo, los esfuerzos realizados hasta el momento no han logrado incorporar la descripción probabilística que entregan aquellos algoritmos, para comparar la calidad de distintas aproximaciones.

Específicamente, existe un gran interés académico asociado al problema de la correcta modelación de baterías, en particular la estimación y pronóstico del estado de carga de baterías (*SoC* - del Inglés *State-of-Charge*). Para la resolución del problema de la estimación y predicción de autonomía, actualmente existen sofisticados algoritmos basados en inferencias bayesianas y aproximaciones de Monte Carlo secuencial que destacan por la posibilidad de aproximar el filtrado óptimo, aún cuando la dinámica del modelo es no lineal y la fuente de incertidumbre no-Gaussiana. Sin embargo, al considerar las restricciones que deben cumplir éstos algoritmos al momento de ser incorporados dentro de sistemas embebidos, se debe tener especial atención en cuanto a la capacidad de funcionamiento en tiempo real y el gasto energético o complejidad computacional que requieren. De esta forma, emerge la necesidad de simplificar algoritmos de estimación y pronóstico de fallas, junto con el desarrollo de un marco de trabajo que permita comparar y cuantificar la degradación que experimentan los resultados producto de las simplificaciones a los algoritmos.

El documento está organizado como sigue. El capítulo 2 presenta las bases teóricas necesarias para la comprensión del documento. Se define el problema que trata el *Cómputo Probablemente Exacto y Correcto*, se introducen conceptos de teoría de la información y se exponen las bases matemáticas que permiten el desarrollo de algoritmos de estimación y pronóstico. Además, se entrega información sobre el caso particular a tratar; baterías de Ion-Litio y la estimación y predicción del estado de carga. En el capítulo 3 se destinan esfuerzos al desarrollo de un marco teórico apropiado para la evaluación del desempeño de algoritmos de estimación y pronóstico. Se contextualiza el *Cómputo Probablemente Exacto y Correcto*, junto con medidas de información, en el problema de comparar simplificaciones a algoritmos basados en métodos secuenciales de Monte Carlo. Finalmente, en el Capítulo 4 se implementa la metodología para contrastar diferentes aproximaciones; unscented Kalman filter, filtro de

partículas, filtro de partículas regularizado, filtro de partículas con propagación de incertidumbre via sigma-points, todas aplicadas al caso de baterías de Ion-Litio, y se utiliza el marco comparativo para determinar el desempeño de cada uno. Se establecen posibles mejoras para los algoritmos evaluados, y se presenta la posibilidad de extender la metodología a casos más generales que el de filtrado bayesiano.

## 1.1. Objetivos

El objetivo de esta Tesis es definir un marco comparativo apropiado para el problema asociado a la evaluación de algoritmos de estimación y pronóstico de fallas. En particular, se desarrolla una metodología que permite contrastar el desempeño de distintas aproximaciones al problema de filtrado bayesiano mediante el método secuencial de Monte Carlo, incluyendo el análisis PACC; herramienta que posibilita aproximar el máximo error teórico de algún algoritmo -dada una función de pérdida-, en conjunto con medidas de información que permiten incorporar la descripción estadística de la evolución del sistema y, de esta manera, cuantificar el desempeño de diferentes aproximaciones a SMC. Específicamente, se desarrolla un caso de estudio donde se analizan distintas aproximaciones al problema de estimación y pronóstico para celdas de Ion-Litio. El objetivo es definir un *Ground Truth* tanto para el escenario de estimación como para el de pronóstico, sujeto a un modelo en espacio de estado, para luego utilizar esta referencia dentro del análisis comparativo y determinar la precisión computacional de aproximaciones más simples, sujetas al mismo modelo en espacio de estado.

## 1.2. Hipótesis

Las hipótesis a validar en este trabajo son las siguientes:

- Incorporar medidas de información para cuantificar el desempeño de algoritmos basados en métodos secuenciales de Monte Carlo permite integrar adecuadamente la descripción estadística que entregan aquellos algoritmos, dentro de un esquema comparativo.
- La factibilidad de obtener una aproximación del máximo error teórico -basado en medidas de información y análisis PACC- en el que incurren distintas aproximaciones de algoritmos de estimación y pronóstico de fallas, y así evaluar su exactitud computacional.
- Es posible generar un algoritmo de referencia para el caso de estimación del estado de carga de celdas de Ion-Litio, asumiendo un modelo en espacio de estado y datos observados de corriente y voltaje.
- Para el escenario de pronóstico se puede generar un algoritmo de referencia para predecir el estado en carga de celdas de Ion-Litio, teniendo acceso a la estimación del algoritmo de referencia en los instantes anteriores al pronóstico y a datos reales de corriente para el horizonte de predicciones, asumiendo un modelo en espacio de estado y ruido de proceso independiente e idénticamente distribuido.

# Capítulo 2

## Fundamentos Generales

Incorporar complejos algoritmos dentro de un sistema embebido presenta problemas; los algoritmos desarrollados a la actualidad han sido diseñados para ejecutarse en hardware disponibles en computadores, es decir, con las herramientas y capacidades que de éste se tiene disposición. En contraposición a estas cualidades, los hardware que deben ser considerados para los sistemas embebidos deben responder a diversas restricciones: tamaño, precio, requerimiento de poco gasto energético, dentro de otros.

Las restricciones mencionadas anteriormente, cobran especial sentido al momento de analizar el problema de estimación y pronóstico de estado de carga en baterías, considerando la importancia de conocer esta variable en sistemas energizados y la necesidad de obtener una estimación en tiempo real de ella. De esta manera, se hace necesario un importante desarrollo: La simplificación y ajuste de algoritmos de estimación de autonomía (estado-de-carga de baterías) con miras a ser embebidos en dispositivos electrónicos de tiempo real, manteniendo la precisión y exactitud demostradas en condiciones controladas de experimentación. En este sentido, distintas simplificaciones a algoritmos de estimación y pronóstico generan un impacto tanto en el gasto energético, como en la calidad de los resultados. En relación a lo anterior, surge la necesidad de incorporar herramientas que permitan cuantificar y describir la degradación en la performance de la aplicación, es decir, la disimilitud que existe entre el desempeño de la implementación computacional sujeta a restricciones de arquitectura y/o físicas, con respecto al caso ideal sin restricciones. De esta forma, es posible evaluar el efecto real producto de las simplificaciones e incertezas en el desempeño de algoritmos de estimación bayesiana.

Una herramienta que permite evaluar estadísticamente el desempeño de algún algoritmo, mediante la aproximación del máximo error teórico en que incurre aquel algoritmo, es el *Cómputo Probablemente Exacto y Correcto*, como se explica en la próxima sección.

## 2.1. Cómputo Probablemente Exacto y Correcto

El Cómputo Probablemente Exacto y Correcto (PACC - del inglés Probably Approximately Correct Computation) [3] es una metodología que nace de la teoría de aprendizaje estadístico (*Statistical Learning Theory* - en Inglés) [34] y del estudio de *Randomized Algorithms* [4], que permite estimar la pérdida en performance en algoritmos con niveles arbitrarios de precisión y confianza. En contraste al enfoque PACC, está el worst-case analysis [8], esta metodología considera el peor escenario posible para cuantificar la pérdida en desempeño de algún *Randomized Algorithm*. La metodología PACC permite cuantificar probabilísticamente la pérdida en performance siendo, de esta forma, una metodología menos conservadora.

Si se considera un algoritmo de referencia y se introducen perturbaciones afectando el funcionamiento ideal del algoritmo, entonces se transforma en su contraparte aproximada. Si se denota por  $f(x)$  -donde  $x \in X$  es el espacio de entrada- al algoritmo de referencia, y por  $f_{\Delta}(x)$  el algoritmo simplificado. Entonces, para caracterizar la precisión computacional del algoritmo simplificado con respecto al algoritmo de referencia, es necesario medir la discrepancia entre  $f(x)$  y  $f_{\Delta}(x)$  por medio de alguna función de pérdida -que debe ser medible de acuerdo a Lebesgue-, como se observa en la ecuación 2.1 :

$$u(x, \Delta) = u(f(x), f_{\Delta}(x)) \quad (2.1)$$

Entonces, para evaluar la pérdida en precisión del algoritmo simplificado es necesario conocer  $u(x, \Delta)$ . Sin embargo, por lo general esta función no puede determinarse de forma cerrada. Para solucionar este problema se puede considerar la ecuación 2.2 y probar si:

$$u(x, \Delta) \leq \gamma \quad \text{se cumple o no, } \forall x \in X \quad (2.2)$$

en correspondencia con todos los valores positivos que puede tomar  $\gamma$ . Si se denota por  $\bar{\gamma}$  el mínimo valor  $\gamma$  tal que 2.2 se cumple completamente, i.e.  $u(x, \Delta) \leq \bar{\gamma}$  se cumple  $\forall x \in X$ . Entonces  $\bar{\gamma}$  se identifica como la máxima pérdida en performance -o máximo error- inducida por las perturbaciones (simplificaciones) y, por lo tanto, entrega un índice de precisión en la performance.

Más allá de las dificultades en encontrar el valor exacto de  $\bar{\gamma}$ , éste valor puede ser demasiado conservador. De hecho, el máximo error surge, en general, con una probabilidad muy pequeña, lo que significa que encontrar  $\bar{\gamma}$  de esta forma, suele ser una opción riesgosa, ya que se tiende a sobredimensionar  $\bar{\gamma}$  con respecto a las necesidades de la aplicación.

La definición del cómputo probablemente exacto y correcto nace de formular el problema probabilístico dual de 2.2:

**Definición 2.1** *Se dice que PACC alcanza el nivel  $\gamma > 0$  con probabilidad  $\eta$ , cuando:*

$$Pr\{u(x, \Delta) \leq \gamma\} \geq \eta, \forall x \in X$$

El problema probabilístico dual es más débil que el caso determinístico y tolera la existencia de un subconjunto de entradas para las cuales la cota asociada a la pérdida de performance



no se cumple ( $u(x, \Delta) > \gamma$ ); la probabilidad de encontrar tales puntos críticos es más pequeña que  $1 - \eta$ . La idea detras de esto es que si se introduce una pérdida de performance  $\gamma$ , entonces se puede garantizar que el algoritmo simplificado no introducirá un error mayor que  $\gamma$  para al menos el  $\eta\%$  de las entradas.

Si se denota por  $p_\gamma = Pr\{u(f(x), f_\Delta(x)) < \gamma\}$  la probabilidad de que la pérdida en performance sea satisfecha a un cierto nivel de pérdida  $\gamma$ , el enfoque probabilístico necesita el conocimiento de  $p_\gamma$ , lo que implica explorar todo el espacio  $X$ ; un problema computacionalmente duro (*computationally hard* - en Inglés). Para sobreponerse a esta dificultad, una opción es aproximar  $p_\gamma$  por  $\hat{p}_{N_C}$ :

$$\hat{p}_{N_C} = \frac{1}{N_C} \sum I(f(x), f_\Delta(x))$$

Donde:

$$I(f(x), f_\Delta(x)) = \begin{cases} 1 & \text{Si } u(f(x), f_\Delta(x)) < \gamma \\ 0 & \text{Si } u(f(x), f_\Delta(x)) > \gamma \end{cases} \quad (2.3)$$

La adherencia de  $p_\gamma$  a  $\hat{p}_{N_C}$  depende del nivel requerido de precisión  $\varepsilon$ , tal que  $|p_\gamma - \hat{p}_{N_C}| \leq \varepsilon$ . Además,  $\hat{p}_{N_C}$  es función del número  $N_C$  de muestra (para distintos valores de  $N_C$  se obtienen distintas estimaciones de  $p_\gamma$ ). Estas fluctuaciones estocásticas pueden ser superadas recurriendo a las probabilidades e introduciendo un grado de confianza  $1 - \delta$ .

Finalmente, la cota de Chernoff [10] entrega el número mínimo de muestras, como se muestra en la ecuación 2.4:

$$N_C \geq \frac{\ln \frac{2}{\delta}}{2\varepsilon^2} \quad (2.4)$$

Tal que se garantiza 2.5:

$$Pr\{|p_\gamma - \hat{p}_{N_C}| < \varepsilon\} > (1 - \delta) \quad (2.5)$$

- Entonces, se obtiene una aproximación de  $\hat{p}_{N_C}$  que es función de  $\gamma$ ;  $\hat{p}_{N_C} = \hat{p}_{N_C}(\gamma) \forall \gamma \geq 0$ . Luego, se debe escoger el mínimo valor  $\hat{\gamma}$  tal que  $\hat{p}_{N_C}(\hat{\gamma}) = 1, \forall \gamma \geq \hat{\gamma}$ .
- Finalmente el análisis PACC permite “estimar la probabilidad  $p_\gamma$  por medio de  $\hat{p}_{N_C}$  con precisión  $\varepsilon$ ” y esta declaración es verdadera al menos con probabilidad  $(1 - \delta)$ . Entonces,  $\hat{\gamma}$  (máximo error empírico) es una aproximación de  $\bar{\gamma}$  (máximo error teórico) que depende del número de muestras  $N_C$ . Considerando lo anterior, el valor que se obtiene para  $\hat{\gamma}$  es un indicador de la exactitud computacional para el algoritmo simplificado. Si  $D_1$  (algoritmo simplificado 1) obtiene una pérdida en performance  $\hat{\gamma}_1$  y  $D_2$  (algoritmo simplificado 2) una pérdida en performance  $\hat{\gamma}_2$ , entonces:  
La exactitud computacional de  $D_1$  es mayor que la de  $D_2 \iff \hat{\gamma}_1 < \hat{\gamma}_2$ .

## 2.2. Inferencia Bayesiana en Modelos ocultos de Markov

Un problema transversal en distintas disciplinas científicas es la estimación de cantidades desconocidas a partir de observaciones de algún proceso, un caso particular es la modelación

de series de tiempo. En general, los modelos ocultos de Markov entregan un marco extremadamente flexible para la modelación de series de tiempo. Sin embargo, el gran poder descriptivo de estos modelos debe ser compensado por el problema de la intratabilidad computacional. Además, es imposible obtener soluciones analíticas al problema de la inferencia salvo algunos casos simples.

Si se considera un proceso de Markov [16],  $\mathcal{X}$ -valuado y de tiempo discreto  $\{X_n\}_{n \geq 1}$  tal que:

$$X_1 \sim \mu(x_1) \quad y \quad X_n | (X_{n-1} = x_{n-1}) \sim f(x_n | x_{n-1}) \quad (2.6)$$

Donde  $\mu(x)$  es una función de densidad de probabilidad y  $f(x|x')$  denota la densidad de probabilidad asociada a la transición desde  $x'$  a  $x$ .

Entonces el objetivo es estimar  $\{X_n\}_{n \geq 1}$ , pero sólo teniendo acceso al proceso  $\mathcal{Y}$ -valuado  $\{Y_n\}_{n \geq 1}$ . Donde se asume que, dado  $\{X_n\}_{n \geq 1}$ , las observaciones ruidosas  $\{Y_n\}_{n \geq 1}$  son estatódicamente independientes y su densidad marginal esta dada por:

$$Y_n | (X_n = x_n) \sim g(y_n | x_n) \quad (2.7)$$

Los modelos que son compatibles con las ecuaciones 2.6 y 2.7 son conocidos como Modelos ocultos de Markov (HMM - del inglés *Hidden Markov Model*) o modelos generales en espacio-estado (SSM - del inglés *general state-space Models*).

Además, las ecuaciones 2.6 y 2.7 definen un modelo Bayesiano [14]. Mediante la ecuación (2.6) se puede definir una distribución a priori (2.8), mientras que por (2.7) se define la función de verosimilitud (2.9):

$$p(x_{1:n}) = \mu(x_1) \prod_{k=2}^n f(x_k | x_{k-1}) \quad (2.8)$$

$$p(y_{1:n} | x_{1:n}) = \prod_{k=1}^n g(y_k | x_k) \quad (2.9)$$

En el contexto Bayesiano, inferencias sobre  $X_{1:n}$  dada una realización de la observación  $Y_{1:n} = y_{1:n}$  se basan en la distribución a posteriori:

$$p(x_{1:n} | y_{1:n}) = \frac{p(x_{1:n}, y_{1:n})}{p(y_{1:n})} \quad (2.10)$$

Bajo supuestos Markovianos, la ecuación (2.10) satisface la siguiente recursión:

$$p(x_{1:n} | y_{1:n}) = p(x_{1:n-1} | y_{1:n-1}) \frac{f(x_n | x_{n-1}) g(y_n | x_n)}{p(y_n | y_{1:n-1})} \quad (2.11)$$

Luego, integrando fuera  $x_{1:n-1}$ , se obtiene la distribución marginal  $p(x_n | y_{1:n})$ :

$$p(x_n | y_{1:n}) = \frac{g(y_n | x_n) p(x_n | y_{1:n-1})}{p(y_n | y_{1:n-1})} \quad (2.12)$$

En general, los métodos de inferencia bayesiana se apoyan en aproximaciones numéricas de la recursión 2.12, como se explicará en las siguientes subsecciones.

### 2.2.1. Filtro de Kalman

El Filtro de Kalman (FK) es uno de los algoritmos más simples que permiten estimar recursivamente el estado de un sistema utilizando un modelo dinámico y observaciones en línea. Si el sistema que se estudia es lineal y Gaussiano, entonces el FK resulta una solución óptima. Es decir, ningún otro algoritmo de filtrado entrega una mejor estimación, tomando como criterio el mínimo error cuadrático medio.

Los supuestos del FK, permiten representar la evolución del sistema [6] según las ecuaciones 2.13 y 2.14:

$$x_k = F_{k-1}x_{k-1} + w_{k-1} \quad (2.13)$$

$$y_k = G_k x_k + v_k, \quad (2.14)$$

donde  $\forall k$   $x_k \in \mathbb{R}^m$  representa el estado del sistema, mientras que  $y_k \in \mathbb{R}^n$  las mediciones.  $F_k \in \mathbb{R}^{m \times m}$  y  $G_k \in \mathbb{R}^{n \times m}$  son matrices variantes en el tiempo. Finalmente, las perturbaciones  $w_k \sim N(0, Q_k)$  y  $v_k \sim N(0, R_k)$  son secuencias i.i.d..

Se puede resumir la metodología de FK según el Algoritmo 1:

---

**Algorithm 1:** Filtro de Kalman

---

- 1 Condiciones Iniciales :
  - 2  $\hat{x}_{0|0} \quad \hat{P}_{0|0}$
  - 3 Predicción :
  - 4  $\hat{x}_{k|k-1} = F_{k-1}\hat{x}_{k-1|k-1}$
  - 5  $\hat{P}_{k|k-1} = F_{k-1}P_{k-1|k-1}F_{k-1}^T + Q_{k-1}$
  - 6 Innovación :
  - 7  $S_k = G_k\hat{P}_{k|k-1}G_k^T + R_k$
  - 8 Ganancia :
  - 9  $K_k = \hat{P}_{k|k-1}G_k^T S_k^{-1}$
  - 10 Actualización :
  - 11  $\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - G_k\hat{x}_{k|k-1})$
  - 12  $\hat{P}_{k|k} = \hat{P}_{k|k-1} - K_k S_k K_k^T$
- 

### 2.2.2. Unscented Kalman Filter

Frente al problema de la estimación del estado de un sistema, FK permite obtener un solución óptima; la minimización del error cuadrático medio, asumiendo, por una parte, la Gaussianidad en el sistema y, por otra, linealidad. Cuando la dinámica del estado del sistema o las observaciones son no-lineales, entonces la función de densidad de probabilidad condicional que obtiene el mínimo error cuadrático medio no es necesariamente Gaussiana. En este sentido, la aproximación mediante FK puede presentar bastante sesgo al momento de estimar el estado de un sistema. Frente a esta problemática se han propuesto diversas extensiones subóptimas al FK; como los algoritmos Filtro de Kalman Extendido (EKF -

del Inglés *Extended Kalman Filter*) [29], Filtro de Kalman Unscented (UKF - del Inglés *Unscented Kalman Filter*) [35].

La principal diferencia entre EKF y UKF radica en la manera en como la variable aleatoria Gaussiana (GRV - del inglés *Gaussian Random Variable*) es representada en la propagación a través de la dinámica del sistema. En el algoritmo EKF la distribución del estado es aproximada por una GRV, que luego es propagada analíticamente a través de la linealización de primer orden del sistema no-lineal. Por otra parte, UKF asume este problema por medio del enfoque de muestreo, utilizando una representación mínima de sigma point, escogidos cuidadosamente. Estos puntos capturan completamente la esperanza y covarianza hasta el segundo orden de exactitud para cualquier no-linealidad. En contraste, EKF sólo alcanza exactitud de primer orden. Considerando, además, que UKF tiene costo computacional del mismo orden que EKF.

Asumiendo el sistema dinámico -con variable exógena  $u_k$  para el instante  $k$ - según las ecuaciones 2.15 y 2.16:

$$x_k = f(x_{k-1}, u_{k-1}) + w_{k-1} \quad (2.15)$$

$$y_k = g(x_k) + v_k \quad (2.16)$$

Suponiendo que el estado del sistema es de dimensión  $n_x$ , entonces se generan  $2n_x + 1$  sigma points. En relación a los pesos, estos se definen:  $W_0^{(m)} = \lambda/(n_x + \lambda)$ ,  $W_0^{(c)} = \lambda/(n_x + \lambda) + (1 + \alpha^2 + \beta)$ ,  $W_i^{(m)} = W_i^{(c)} = 1/(2(n_x + \lambda))$ ,  $i = 1, \dots, 2n_x$ . Además se consideran los siguientes parámetros de escala:  $\gamma = \sqrt{n_x + \gamma}$ ,  $\lambda = \alpha^2(n_x + \kappa) - n_x$  y  $\kappa$ . La constante  $\alpha$  determina la dispersión de los *sigma points* en torno a  $\hat{x}$ , mientras que  $\beta$  corresponde a un grado de libertad adicional que permite incorporar conocimiento a priori extra. Un resumen de la rutina UKF se puede apreciar en la Algoritmo 2

### 2.2.3. Métodos Secuenciales de Monte Carlo

Los Métodos Secuenciales de Monte Carlo (SMC) [5] son una clase general de métodos de Monte Carlo que muestrean secuencialmente desde una secuencia de densidades de probabilidad objetivo  $\{\pi_n(x_{1:n})\}$  de dimensión creciente, donde cada distribución  $\pi_n(x_{1:n})$  esta definida en el espacio producto  $\mathcal{X}^n$ . Si se escribe  $\pi_n(x_{1:n})$  según la ecuación 2.17:

$$\pi_n(x_{1:n}) = \frac{\gamma_n(x_{1:n})}{Z_n} \quad (2.17)$$

Entonces, solo se requiere que  $\gamma_n : \mathcal{X}^n \rightarrow \mathbb{R}^+$  sea conocida punto a punto. Además, si la constante de normalización -que puede ser desconocida- cumple:

$$Z_n = \int \gamma_n(x_{1:n}) dx_{1:n} \quad (2.18)$$

Entonces, si se considera el instante  $m$ , SMC entrega una aproximación para  $\pi_m(x_{1:m})$  y una estimación para  $Z_m$ .

---

**Algorithm 2:** Filtro de Kalman Unscented

---

1 Condiciones Iniciales :

2  $\hat{x}_{0|0} \quad \hat{P}_{0|0}$

3 Cálculo de Sigma Points :

4  $\mathcal{X}_{k-1|k-1}^0 = \hat{x}_{k-1|k-1}$

5  $\mathcal{X}_{k-1|k-1}^i = \hat{x}_{k-1|k-1} + (\gamma\sqrt{\hat{P}_{k-1|k-1}})_i \quad i = 1, \dots, n_x$

6  $\mathcal{X}_{k-1|k-1}^{i+n_x} = \hat{x}_{k-1|k-1} - (\gamma\sqrt{\hat{P}_{k-1|k-1}})_i \quad i = 1, \dots, n_x$

7 Predicción del estado :

8  $\mathcal{X}_{k|k-1}^i = f(\mathcal{X}_{k-1|k-1}^i, u_{k-1})$

9  $\hat{x}_{k|k-1} = \sum_{i=0}^{2n_x} W_i^{(m)} \mathcal{X}_{k|k-1}^i$

10 Cálculo de Sigma Points para la observación :

11  $\mathcal{X}_{k|k-1}^0 = \hat{x}_{k|k-1}$

12  $\mathcal{X}_{k|k-1}^i = \hat{x}_{k|k-1} + (\gamma\sqrt{\hat{P}_{k|k-1}})_i \quad i = 1, \dots, n_x$

13  $\mathcal{X}_{k|k-1}^{i+n_x} = \hat{x}_{k|k-1} - (\gamma\sqrt{\hat{P}_{k|k-1}})_i \quad i = 1, \dots, n_x$

14 Predicción de la observación :

15  $\mathcal{Y}_{k|k-1} = g(\mathcal{X}_{k|k-1})$

16  $\hat{y}_{k|k-1} = \sum_{i=0}^{2n_x} W_i^{(m)} \mathcal{Y}_{k|k-1}^i$

17 Residuos de la observación :

18  $\varepsilon_{k|k-1}^i = \mathcal{Y}_{k|k-1}^i - \hat{y}_{k|k-1}$

19  $R_{k|k-1}^{\varepsilon\varepsilon} = \sum_{i=0}^{2n_x} W_i^{(c)} \varepsilon_{k|k-1}^i (\varepsilon_{k|k-1}^i)^T + R_{vv}(k)$

20 Ganancia :

21  $R_{k|k-1}^{x\varepsilon} = \sum_{i=0}^{2n_x} W_i^{(c)} (\mathcal{X}_{k|k-1}^i - \hat{x}_{k|k-1}) (\varepsilon_{k|k-1}^i)^T$

22  $K_k = R_{k|k-1}^{x\varepsilon} (R_{k|k-1}^{\varepsilon\varepsilon})^{-1}$

23 Actualización :

24  $\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k (y_k - \hat{y}_{k|k-1})$

25  $\hat{P}_{k|k} = \hat{P}_{k|k-1} - K_k R_{k|k-1}^{\varepsilon\varepsilon} K_k^T$

---

Es importante destacar que SMC es una metodología general, donde dependiendo de la elección de  $\gamma_n$  y  $Z_n$  se pueden obtener distintas distribuciones objetivos. Esto es lo que permite utilizar SMC para filtrado, donde se escoge, por ejemplo,  $\gamma_n = p(x_{1:n}, y_{1:n})$  y  $Z_n = p(y_{1:n})$ , luego  $\pi_n(x_{1:n}) = p(x_{1:n}|y_{1:n})$

## Métodos de Monte Carlo

El objetivo es aproximar la densidad de probabilidad genérica  $\pi_n(x_{1:n})$  para algun  $n$ . Entonces, si se muestrean  $N$  variables aleatoria (VA) independientes,  $X_{1:n}^i \sim \pi_n(x_{1:n})$  para  $i = 1, \dots, N$ , luego los métodos de Monte Carlo aproximan  $\pi_n(x_{1:n})$  mediante la medida empírica:

$$\hat{\pi}_n(x_{1:n}) = \frac{1}{N} \sum_{i=1}^N \delta_{X_{1:n}^i}(x_{1:n}) \quad (2.19)$$

Donde  $\delta_{x_0}(x)$  es la función delta Dirac localizada en  $x_0$ .

Mediante esta aproximación es posible aproximar cualquier marginal  $\pi_n(x_k)$ . [14].

En relación a las aproximaciones por los métodos de Monte Carlo existen algunos problemas que se pueden presentar: 1. Si  $\pi_n(x_{1:n})$  es una distribución de probabilidades compleja de alta dimensionalidad, entonces no se puede muestrear desde ella. 2. Incluso en el caso de que se pudiera muestrear exactamente desde  $\pi_n(x_{1:n})$ , el costo computacional asociado a este esquema de muestro es típicamente, al menos, lineal en el numero de variables  $n$ . Por lo que muestrear exactamente desde  $\pi_n(x_{1:n})$  trae como consecuencia un alto costo computacional.

## Muestreo de Importancia

En relación al primer problema de los métodos de Monte Carlo; la imposibilidad de muestrear desde la distribución de probabilidad objetivo, es que se desarrolla el método de Muestreo de Importancia (IS- del Inglés *Importance Sample*). IS se sutenta en la introducción de una densidad de importancia (*importance density*- en Inglés)  $q_n(x_{1:n})$  tal que contenga el soporte de la distribución objetivo:

$$\pi_n(x_{1:n}) > 0 \Rightarrow q_n(x_{1:n}) > 0 \quad (2.20)$$

En consecuencia, si se utilizan las ecuaciones 2.17 y 2.18, se obtiene 2.21 y 2.22:

$$\pi_n(x_{1:n}) = \frac{w_n(x_{1:n})q_n(x_{1:n})}{Z_n} \quad (2.21)$$

$$Z_n = \int w_n(x_{1:n})q_n(x_{1:n})dx_{1:n} \quad (2.22)$$

Donde  $w_n(x_{1:n})$  es la función de pesos sin normalizar:

$$w_n(x_{1:n}) = \frac{\gamma_n(x_{1:n})}{q_n(x_{1:n})} \quad (2.23)$$

Entonces, si se extraen  $N$  muestras independientes de  $X_{1:n}^i \sim q_n(x_{1:n})$ , luego utilizando la aproximación de Monte Carlo de  $q_n(x_{1:n})$  en 2.21 y 2.22, se obtiene:

$$\hat{\pi}_n(x_{1:n}) = \frac{1}{N} \sum_{i=1}^N W_n^i \delta_{X_{1:n}^i}(x_{1:n}) \quad (2.24)$$

$$\hat{Z}_n = \frac{1}{N} \sum_{i=1}^N w_n(X_{1:n}^i) \quad (2.25)$$

$$W_n^i = \frac{w_n(X_{1:n}^i)}{\sum_{j=1}^N w_n(X_{1:n}^j)} \quad (2.26)$$

Y de esta manera se puede obtener una aproximación de la distribución objetivo sin muestrear directamente de ella.

## Muestreo de Importancia Secuencial

En relación al segundo problema con los métodos de Monte Carlo; el aumento en la complejidad computacional relacionada a la estimación de la distribución objetivo, el Muestreo de Importancia secuencial (SIS -del Inglés *Sequential Importance Sample*) propone una alternativa que admite una complejidad computacional fija. Esta solución implica seleccionar una *distribucion de importancia* que tiene la siguiente estructura:

$$q_n(x_{1:n}) = q_{n-1}(x_{1:n-1})q_n(x_n|x_{1:n-1}) = q_1(x_1) \prod_{k=2}^n q_k(x_k|x_{1:k-1}) \quad (2.27)$$

En la práctica esto significa que para obtener muestras  $X_{1:n}^i \sim q_n(x_{1:n})$  en el instante  $n$ , se muestrea  $X_1^i \sim q_1(x_1)$  en el instante 1, luego  $X_k^i \sim q_k(x_k|X_{1:k-1}^i)$  en el instante  $k$ , donde  $k = 2, \dots, n$ . Los pesos sin normalizar asociados pueden ser calculados recursivamente según la ecuación 2.28:

$$w_n(x_{1:n}) = w_{n-1}(x_{1:n-1})\alpha_n(x_{1:n}) = w_1(x_1) \prod_{k=2}^n \alpha_k(x_{1:k}) \quad (2.28)$$

Donde  $\alpha_n(x_{1:n})$  es la función *incremental importance weight*, como muestra la ecuación 2.29:

$$\alpha_n(x_{1:n}) = \frac{\gamma_n(x_{1:n})}{\gamma_{n-1}(x_{1:n-1})q_n(x_n|x_{1:n-1})} \quad (2.29)$$

## Remuestreo

Además de los problemas que resuelven el Muestreo de Importancia y Muestreo de Importancia Secuencial respectivamente, existe un tercer problema asociado a la estimación

mediante estos métodos; el incremento de la varianza a medida que se realiza el filtrado. Este aumento en la varianza de los pesos significa que se asocian ponderaciones despreciables a algunas partículas -es decir, hay partículas que representan estados muy poco probables, por lo que son poco útiles-, lo que conduce a la degeneración en la población. En este sentido, las técnicas de remuestreo (*Resample*- en Inglés) buscan atacar esta degeneración. El objetivo es utilizar algún criterio que permita cuantificar la dispersión de las partículas y, en base a él, determinar si es necesario ejecutar una rutina de remuestreo.

Si se considera, inicialmente, una aproximación IS  $\hat{\pi}_n(x_{1:n})$  de la distribución objetivo  $\pi_n(x_{1:n})$ , es decir muestras ponderadas de  $q_n(x_{1:n})$ . Entonces, si esta aproximación IS muestra degeneración en la población es posible muestrear directamente desde  $\hat{\pi}_n(x_{1:n})$  para aproximar  $\pi_n(x_{1:n})$ . Esta nueva aproximación ( $\bar{\pi}_n(x_{1:n})$ ) consiste en seleccionar  $X_{1:n}^i$  con probabilidad  $W_n^i$ ; esta rutina se llama remuestreo. Es importante mencionar que  $\bar{\pi}_n(x_{1:n})$  es una aproximación insesgada de  $\hat{\pi}_n(x_{1:n})$ .

Un criterio típico utilizado para medir degeneración en la población de partículas es el *effective particle sample size*, como muestra la ecuación 2.30

$$N_{eff}(k) = \frac{N_p}{1 + Var_{p(\cdot|y_{1:k})}}(w(x_k)) \quad (2.30)$$

La decisión se toma al verificar si se cumple que  $N_{eff}(k) \leq N_{tresh}$ , donde  $N_{tresh}$  es un umbral fijo. Si se cumple esta relación, entonces la población de partículas ya se ha degenerado lo suficiente como para ser necesaria una rutina de remuestreo y, de esta forma, eliminar partículas poco representativas y darle más valor a aquellas representaciones más probables del estado.

## 2.2.4. Algoritmo secuencial de Monte Carlo genérico

Los métodos secuenciales de Monte Carlo son una combinación de los algoritmos SIS y Resample. En el instante  $n$  el algoritmo entrega una aproximación de  $\pi_n(x_{1:n})$  según la ecuación 2.31:

$$\hat{\pi}_n(x_{1:n}) = \sum_{i=1}^N W_n^i \delta_{X_{1:n}^i(x_{1:n})} \quad (2.31)$$

Luego, si se cumple el criterio para remuestrear, entonces se obtiene la aproximación de la ecuación 2.32:

$$\bar{\pi}_n(x_{1:n}) = \frac{1}{N} \sum_{i=1}^N \delta_{\bar{X}_{1:n}^i(x_{1:n})} \quad (2.32)$$

Donde  $\bar{X}_{1:n}^i$  se obtienen por medio de la rutina de remuestreo. Para resumir, un algoritmo general para los métodos secuenciales de Monte Carlo se muestra en el Algoritmo 3:



---

**Algorithm 3:** Monte Carlo Secuencial

---

- 1 En el instante  $n = 1$  :
  - 2 Muestrear  $X_1^i \sim q_1(x_1)$
  - 3 Calcular los pesos  $w_1(X_1^i)$  y  $W_1^i \propto w_1(X_1^i)$
  - 4 Si el criterio de resmuestreo se cumple entonces remuestrear  $\{W_1^i, X_1^i\}$  para obtener  $N$  partículas con el mismo peso  $\{\frac{1}{N}, \bar{X}_1^i\}$  y definir  $\{\bar{W}_1^i, \bar{X}_1^i\} \leftarrow \{\frac{1}{N}, \bar{X}_1^i\}$ , de otra forma  $\{\bar{W}_1^i, \bar{X}_1^i\} \leftarrow \{W_1^i, X_1^i\}$
  - 5 En el instante  $n \geq 2$  :
  - 6 Muestrear  $X_n^i \sim q_n(x_n | \bar{X}_{1:n-1}^i)$  y definir  $X_{1:n}^i \leftarrow (\bar{X}_{1:n-1}^i, X_n^i)$
  - 7 Calcular los pesos  $\alpha_n(X_{1:n}^i)$  y  $W_n^i \propto \bar{W}_{n-1}^i \alpha_n(X_{1:n}^i)$
  - 8 Si el criterio de resmuestreo se cumple entonces remuestrear  $\{W_n^i, X_n^i\}$  para obtener  $N$  partículas con el mismo peso  $\{\frac{1}{N}, \bar{X}_{1:n}^i\}$  y definir  $\{\bar{W}_n^i, \bar{X}_n^i\} \leftarrow \{\frac{1}{N}, \bar{X}_n^i\}$ , de otra forma  $\{\bar{W}_n^i, \bar{X}_n^i\} \leftarrow \{W_n^i, X_n^i\}$
- 

**SMC para filtrado Bayesiano: Filtro de Partículas**

En el contexto del filtrado el objetivo es obtener una aproximación numerica de la distribución  $\{p(x_{1:n}|y_{1:n})\}_{n \geq 1}$  secuencialmente en el tiempo. Esto es posible aplicando los métodos secuenciales de Monte Carlo, descritos anteriormente, a la secuencia de distribuciones objetivos  $\pi_n(x_{1:n}) = p(x_{1:n}|y_{1:n})$ . Es importante destacar que al modificar la distribución objetivo es posible obtener algoritmos más complejos. SMC para el filtrado, también llamado Filtro de Partículas, se puede resumir en el Algoritmo 4:

---

**Algorithm 4:** Filtro de Partículas

---

- 1 En el instante  $n = 1$  :
  - 2 Muestrear  $X_1^i \sim q_1(x_1)$
  - 3 Calcular los pesos  $w_1(X_1^i) = \frac{\mu(X_1^i)g(y_1|X_1^i)}{q(X_1^i|y_1)}$  y  $W_1^i \propto w_1(X_1^i)$
  - 4 Si el criterio de resmuestreo se cumple entonces remuestrear  $\{W_1^i, X_1^i\}$  para obtener  $N$  partículas con el mismo peso  $\{\frac{1}{N}, \bar{X}_1^i\}$  y definir  $\{\bar{W}_1^i, \bar{X}_1^i\} \leftarrow \{\frac{1}{N}, \bar{X}_1^i\}$ , de otra forma  $\{\bar{W}_1^i, \bar{X}_1^i\} \leftarrow \{W_1^i, X_1^i\}$
  - 5 En el instante  $n \geq 2$  :
  - 6 Muestrear  $X_n^i \sim q_n(x_n | y_n, \bar{X}_{n-1}^i)$  y definir  $X_{1:n}^i \leftarrow (\bar{X}_{1:n-1}^i, X_n^i)$
  - 7 Calcular los pesos  $\alpha_n(X_{n-1:n}^i) = \frac{g(y_n|X_n^i)f(X_n^i|X_{n-1}^i)}{q(X_n^i|y_n, X_{n-1}^i)}$  y  $W_n^i \propto \alpha_n(X_{n-1:n}^i)$
  - 8 Si el criterio de resmuestreo se cumple entonces remuestrear  $\{W_n^i, X_n^i\}$  para obtener  $N$  partículas con el mismo peso  $\{\frac{1}{N}, \bar{X}_{1:n}^i\}$  y definir  $\{\bar{W}_n^i, \bar{X}_n^i\} \leftarrow \{\frac{1}{N}, \bar{X}_n^i\}$ , de otra forma  $\{\bar{W}_n^i, \bar{X}_n^i\} \leftarrow \{W_n^i, X_n^i\}$
- 

De esta forma, en el instante  $n$  se puede estimar la distribución a posteriori del estado según la ecuación 2.33:

$$\hat{p}(x_{1:n}|y_{1:n}) = \sum_{i=1}^N W_n^i \delta_{X_{1:n}^i}(x_{1:n}) \quad (2.33)$$

## 2.2.5. Suavizado en SMC

Un problema que esta muy relacionado con el filtrado, pero computacionalmente más desafiante, es conocido como suavizado (*smoothing*- en Inglés). Mientras el filtrado corresponde a la estimación de la distribución del estado actual de un HMM basados en observaciones recibidas hasta el tiempo actual, el suavizado corresponde a la estimación de la distribución del estado en un tiempo específico dadas todas las observaciones hasta un tiempo *futuro* [14].

Formalmente, si se tiene acceso a las observaciones  $y_{1:T}$ , entonces el objetivo es calcular las distribuciones marginales  $\{p(x_n|y_{1:T})\}$  donde  $n = 1, \dots, T$ , o muestrear desde  $p(x_{1:T}|y_{1:T})$ . En principio, las marginales  $p(x_n|y_{1:T})$  se pueden obtener directamente al considerar la distribución conjunta  $p(x_{1:T}|y_{1:T})$  e integrar fuera las variables  $(x_{1:n-1}, x_{n+1:T})$ . Extendiendo este razonamiento en el contexto de métodos de partículas, se puede utilizar la identidad

$$p(x_n|y_{1:T}) = \int p(x_{1:T}|y_{1:T}) dx_{1:n-1} dx_{n+1:T}$$

y tomar el mismo enfoque utilizado en el filtro de partículas: Utilizar algoritmos de Monte Carlo para obtener una caracterización aproximada de la distribución conjunta y luego utilizar las distribuciones marginal asociadas para aproximar la distribución de interés. Uno de los algoritmos que utiliza este enfoque es la variante Forward Filter-Backward Smoothing.

### Forward Filter-Backward Smoothing

El objetivo de este procedimiento es obtener  $\{p(x_n|y_{1:T})\}$  muestreando desde  $\{p(x_{1:T}|y_{1:T})\}$ . El esquema que se muestra a continuación genera muestras distribuidas aproximadamente de acuerdo a  $p(x_{1:T}|y_{1:T})$ . Primeramente, se realiza un filtrado desde  $n = 1$  hasta  $n = T$ , de esta forma se obtiene  $\{\hat{p}(x_{1:T}|y_{1:T})\}$ . Luego, se muestrea  $X_T \sim \hat{p}(x_T|y_{1:T})$  y para  $n = T - 1, T - 2, \dots, 1$  muestrear  $X_n \sim \hat{p}(x_n|X_{n+1}, y_{1:n})$ . Entonces se puede realizar una aproximación de  $p(x_n|y_{1:T})$  por medio de partículas, según la ecuación 2.34:

$$\hat{p}(x_n|y_{1:T}) = \sum_{i=1}^N W_n^i \left[ \sum_{j=1}^N W_{n+1|T}^j \frac{f(X_{n+1}^j|X_n^i)}{\sum_{l=1}^N [W_n^l f(X_{n+1}^l|X_n^l)]} \right] \delta_{X_n^i}(x_n) \quad (2.34)$$

Un esquema que permite obtener una realización de esta aproximación es el Algoritmo 5 [31]. Para generar otras realizaciones independientes basta con repetir el procedimiento la cantidad de veces que sea necesario.:

## 2.3. Pronóstico

El desafío de realizar pronóstico sobre la condición o estado de un sistema, puede comprenderse como un problema análogo al de estimación del estado, pero cuando ya no se tiene acceso a observaciones del proceso, por lo que no es factible obtener la distribución *a posteriori*, ya que no es posible modificar la distribución *a priori* pues no hay información que

---

**Algorithm 5:** Forward Filter Backward Smoothing; Realización por muestra

---

- 1 En el instante  $n = T$  :
  - 2 Escoger la partícula  $\hat{x}_T = x_T^i$  con probabilidad  $w_T^i$
  - 3 En el instante  $n \leq T$  :
  - 4 Calcular  $w_{t|t+1}^i \propto w_t^i f(\hat{x}_{t+1}|x_t^i)$
  - 5 Escoger  $\hat{x}_t = x_t^i$  con probabilidad  $w_{t|t+1}^i$
  - 6 Finalmente:
  - 7  $\hat{x}_{1:T} = (\hat{x}_1, \hat{x}_2, \dots, \hat{x}_T)$  es una realización aproximadamente distribuida según  $\hat{p}(x_{1:T}|y_{1:T})$
- 

permita manipular la creencia inicial del estado. En este sentido, el problema de predecir la evolución del sistema, también puede comprenderse como un caso particular de Monte Carlo Secuencial, donde la distribución objetivo a estimar, en el instante  $n + T_p$  (donde  $T_p \geq 1$  es el último instante que se obtienen observaciones) es  $\pi_{n+T_p}(x_{1:n+T_p}) = p(x_{n+T_p}|x_{1:T_p}, y_{1:T_p})$ , teniendo acceso solamente a  $\{Y_k\}_{T_p \geq k \geq 1}$ .

### 2.3.1. Pronóstico: Basado en Monte Carlo Secuencial

Como se explica en la Subsección 2.2.4, se puede aproximar cualquier distribución objetivo por medio de métodos Secuenciales de Monte Carlo. Además se dispone de una metodología genérica para realizar esta estimación que consiste en una técnica que utiliza Resampling, SIS y la aproximación de Monte Carlo. Este enfoque permite, fundamentalmente, muestrear secuencialmente desde una distribución de importancia ( $q_{n+T_p}(x_{1:n+T_p})$ ) en miras a aproximar una distribución objetivo ( $\pi_{n+T_p}(x_{1:n+T_p}) = p(x_{n+T_p}|x_{1:T_p}, y_{1:T_p})$ ). Un esquema que permite obtener esta aproximación [5] es el Algoritmo 6. Éste Algoritmo es, básicamente, el Algoritmo 3 (Monte Carlo Secuencial), pero sin la modificación de los pesos, e incluyendo un Resampling en el primer paso, pero sin volver a ejecutar esta rutina. La densidad de importancia, para

---

**Algorithm 6:** Monte Carlo Secuencial Sin observaciones

---

- 1 En el instante  $n \leq T_p$  :
  - 2 Se aproxima  $\pi_n(x_{1:n}) \approx \hat{\pi}_{1:n}(x_{1:n})$  por medio del Algoritmo 3
  - 3 En el instante  $n = T_p$  :
  - 4 Remuestrear  $\{W_{T_p}^i, X_{T_p}^i\}$  para obtener N partículas con el mismo peso  $\{\frac{1}{N}, \bar{X}_{T_p}^i\}$  y definir  $\{\bar{W}_{T_p}^i, \bar{X}_{T_p}^i\} \leftarrow \{\frac{1}{N}, \bar{X}_{T_p}^i\}$
  - 5 En el instante  $n \geq T_p + 1$  :
  - 6 Muestrear  $X_n^i \sim q_n(x_n|\bar{X}_{1:n-1}^i)$  y definir  $X_{1:n}^i \leftarrow (\bar{X}_{1:n-1}^i, X_n^i)$
  - 7 Definir  $\{\bar{W}_n^i, \bar{X}_n^i\} \leftarrow \{\frac{1}{N}, \bar{X}_n^i\}$
- 

el instante  $n \geq T_p$  cumple con la ecuación 2.35

$$q_n(x_{T_p:n}) = q_{n-1}(x_{T_p:n-1})q_n(x_n|x_{T_p:n-1}) = q_{T_p}(x_{T_p}) \prod_{k=2}^n q_k(x_k|x_{1:k-1}) \quad (2.35)$$

Donde  $q_{T_p}(x_{T_p}) = \bar{\pi}_{T_p}(x_{T_p})$  y  $q_n(x_n|x_{T_p:n-1}) = f(x_n|x_{n-1})$  ( $f$  es la densidad de probabilidad asociada a la transición de  $x_{n-1}$  a  $x_n$ ). Luego, para las predicciones, se puede aproximar la distribución  $\pi_n(x_{T_p:n})$  por medio de la medida empírica (ecuación 2.32):

$$\pi_n(x_{T_p:n}) \approx \bar{\pi}_n(x_{T_p:n}) = \frac{1}{N} \sum_{i=1}^N \delta_{\bar{X}_{T_p:n}^i}(x_{T_p:n})$$

### 2.3.2. Pronóstico Mediante Filtro de Partículas: Regularización

Dentro de la literatura existen diversos métodos que permiten enfrentar el problema de pronóstico. Un algoritmo que también enfrenta este desafío es la generación de predicciones a largo plazo basandose en el algoritmo de filtro de partículas regularizado [25]. En lugar de recalcular los pesos de las partículas, se propone una representación de la incertidumbre en los instantes futuros mediante una aproximación discreta, como se observa en la ecuación 2.36.

$$\hat{p}(\hat{x}_{T_p+n}|\hat{x}_{T_p+n-1}) = \sum_{i=1}^N w_{T_p}^i K_h(\hat{x}_{T_p+n} - \mathbb{E}[\hat{x}_{T_p+n}|\hat{x}_{T_p+n-1}]) \quad (2.36)$$

Además, se definen:

$$K_h = \frac{1}{h_{n_x}} K\left(\frac{x}{h}\right), \quad h_{opt} = A \cdot N^{\frac{1}{n_x+4}}, \quad A = \left(8c_{n_x}^{-1}(n_x + 4)(2\sqrt{\pi})^{n_x}\right)^{\frac{1}{n_x+4}} \quad (2.37)$$

$$K(x) = \begin{cases} \frac{n_x+2}{2c_{n_x}}(1 - \|x\|^2), & \text{si } \|x\| \leq 1 \\ 0, & \sim \end{cases} \quad (2.38)$$

Donde  $K(\cdot)$  es una función densidad que corresponde a la pdf de una fuente de incertidumbre como una distribución Gaussiana o una versión del Kernel de Epanechnikov,  $\mathbb{E}[\cdot]$  la Esperanza,  $c_{n_x}$  es el volumen de una esfera unitaria en  $\mathbb{R}^{n_x}$  y  $n_x$  la dimensión del vector de estado.

De esta forma, la información acerca de la distribución del estado para los instantes futuros está dada solo por la posición de las partículas, independiente del valor de los pesos. Es importante destacar que esta metodología asume que los ruidos de proceso no estan correlacionados. En resumen, el algoritmo de regularización para generar predicciones a largo plazo es el Algoritmo 7:

### 2.3.3. Pronóstico Mediante Filtro de Partículas con propagación de incertidumbre via sigma-points

Inspirado en el algoritmo *Unscented Transform Approximation Clustering* (UTAC) [19], el pronóstico basado en filtro de partículas con propagación de incertidumbre via sigma-points [2] es un algoritmo que intenta mejorar los esquemas de pronóstico basados en PF al caracterizar apropiadamente las colas de la pdf asociada al vector de estado del sistema de

---

**Algorithm 7:** Metodología basada en Filtro de Partícula regularizado para pronóstico

---

- 1 Aplicar el procedimiento de remuestreo modificado de la transformación inversa.  
 $i = 1, \dots, N, w_{T_p+n}^i = N^{-1}$
  - 2 Calcular  $\hat{S}_{T_p+n}$ , matriz de covarianza empírica de  $\{E[\hat{x}_{T_p+n}|\hat{x}_{T_p+n-1}], w_{T_p+n-1}^i\}_{i=1}^N$
  - 3 Calcular  $\hat{D}_{T_p+n}$  de manera que  $\hat{D}_{T_p} + n\hat{D}^T T_p + n = \hat{S}_{T_p+n}$
  - 4 Para  $i = 1, \dots, N$  Obtener  $\varepsilon^i \sim K$ , de la distribución de Epanechnikov y asignar  
 $\hat{x}_{T_p+n}^{i*} = \hat{x}_{T_p+n}^i + h_{T_p+n}^{opt} \hat{D}_{T_p+n} \varepsilon^i$
- 

interés. Para esto, se tiene acceso a la estimación del estado del sistema en el instante  $k = T_p$ , mediante un filtro de partículas. Se utiliza la propagación consecutiva de sigma-points y el cómputo de modelos de mezcla de Gaussianas (GMMs- del inglés *Gaussian Mixture Models*), como se indica en el Algoritmo 8:

---

**Algorithm 8:** Filtro de Partículas con propagación de incertidumbre via sigma-points

---

- 1 Paso 1: Transición de estado con Sigma-Points.
  - 2 Se aproxima la pdf del ruido de proceso con  $N_h$  componentes de una GMM  
 $w_k \sim \sum_{l=1}^{N_h} \gamma_k^{(l)} \mathcal{N}(\mu_k^{(l)}, \Sigma_k^{(l)})$
  - 3 Luego, se representa cada componente de la GMM por un conjunto de  $2n_x + 1$  sigma-points, así se obtiene la siguiente aproximación:
  - 4  $\sum_{l=1}^{N_h} \gamma_k^{(l)} \mathcal{N}(x_k; \mu_k^{(l)}, \Sigma_k^{(l)}) \approx \sum_{l=1}^{N_h} \sum_{j=1}^{2n_x+1} \gamma_k^{(l,j)} \mathcal{W}_k^{(l,j)} \delta_{\mathcal{X}_k^{(l,j)}}(x_k)$
  - 5 Entonces, si se tiene una población de  $N_k$  partículas para el instante  $k$ , se obtiene una aproximación para la distribución de la transición del estado:
  - 6  $p(x_k | y_{1:T_p}) \approx \sum_{i^*=1}^{N_k N_h (2n_x+1)} \tilde{W}_k^{(i^*)} \delta_{\tilde{x}_k^{(i^*)}}(x_k)$
  - 7 Paso 2: Ajustar una GMM
  - 8 Tomando en cuenta que el Paso 1 aumenta la población de partículas, ahora se utiliza el algoritmo Weighted Expectation Maximization para ajustar una GMM con  $M_k$  componentes ( $GMM_k$ ):
  - 9  $GMM_k(x_k) = \sum_{j^*=1}^{M_k} \alpha_{WEM_k}^{(j^*)} \mathcal{N}(x_k; \mu_{WEM_k}^{(j^*)}, \Sigma_{WEM_k}^{(j^*)})$
  - 10 Paso 3: Simplificación utilizando Sigma-Points
  - 11 Dado que se obtiene una representación del estado por  $GMM_k$  con  $M_k$  componentes, se vuelven a introducir sigma-points para representar  $GMM_k$  con  $M_k(2n_x + 1)$  partículas, de esta forma el tamaño de la población queda definido por:
  - 12  $N_{k+1} = M_k(2n_x + 1)$
  - 13 Finalmente, repitiendo los Pasos del 1 al 3 se puede obtener la predicción del estado del sistema para un horizonte arbitrario.
- 

El algoritmo Weighted Expectation Maximization es un método iterativo que permite encontrar estimadores de máxima verosimilitud de los parámetros de un modelo estadístico asociado a datos incompletos [22]. Para el punto 6,  $\tilde{x}_k^{(i^*)}(x_k) = f(x_{k-1}^{(i)}, u_{k-1}, \mathcal{X}_{k-1}^{(l,j)})$  y  $\tilde{W}_k^{(i^*)} = W_{k-1}^{(i)} \gamma_{k-1}^{(l)} \mathcal{W}_{k-1}^{(l,j)}$  -donde  $W_k$  es el peso asociado a las partículas  $N_k$ -. Además, a  $i^*$  se le asigna una combinación única de  $i, j, l$ .

## 2.4. Lazo de Corrección Externa

En general, cuando se realiza estimación por medio de un modelo en espacio-estado, hay parámetros de dicho modelo que son desconocidos y, por tanto, afectan negativamente el desempeño de la estimación del estado pues entregan información errórea o poco certera. En este sentido, lo ideal sería poder realizar la estimación del estado del sistema en conjunto con un proceso de aprendizaje relacionado a los parámetros que se desconocen del modelo. Un método que enfrenta este desafío, por medio de la extensión del vector de estados para incorporar aquellos parámetros desconocidos, es la *evolución artificial* [26].

Por otra parte están los lazos de corrección externa (OFCL- del inglés *Outer Feedback Correction Loops*) [20]. Estos son una clase de algoritmos cuya finalidad es manipular los hiperparámetros que caracterizan los modelos en espacio de estados. El objetivo de estos algoritmos es permitir flexibilidad a los hiperparámetros que definen al modelo y, de esta forma, incorporar aquella información como un aprendizaje.

## 2.5. Medidas de Información

La Teoría de la Información estudia la cuantificación, almacenamiento y comunicación de la información. Inicialmente, esta teoría fue introducida por Claude E. Shannon [30]. El objetivo en ese entonces era encontrar límites fundamentales en el procesamiento de señales y en comprensión de datos.

Actualmente, la teoría de la información ha encontrado aplicación en muchas otras áreas incluyendo [11], [15]; inferencia estadística, criptografía, evolución, neurobiología, computación cuántica, reconocimiento de patrones, detección de plagas, dentro de muchas otras.

Tomando en cuenta que, para cada instante de tiempo, los algoritmos de estimación y pronóstico de fallas aproximan la distribución de probabilidad del estado del sistema. Entonces, se introducen las siguientes medidas de información.

### 2.5.1. Entropía

Un concepto fundamental dentro de la teoría de la información -que permite medir la incertidumbre asociada a una variable aleatoria- es la entropía [11]:

**Definición 2.2** La entropía  $H(X)$  de una variable aleatoria discreta  $X$ , con alfabeto  $\chi$  y función de probabilidad de masa  $p(x) = Pr\{X = x\}$ ,  $x \in \chi$  se define por:

$$H(X) = - \sum_{x \in \chi} p(x) \log p(x)$$

Es importante notar que la entropía es un funcional de la distribución de  $X$ , por lo que

no depende en los valores actuales tomados por la variable aleatoria  $X$ , sino solo en las probabilidades.

## 2.5.2. Divergencia Kullback-Lieber

La entropía de una variable aleatoria es una medida de la incerteza de la variable aleatoria; es una medida de la cantidad de información requerida en promedio para describir la variable aleatoria. La entropía relativa o distancia de Kullback-Lieber  $D(p||q)$  es una medida de la ineficiencia asociada a asumir que la distribución es  $q$  cuando la distribución real es  $p$  [11]:

**Definición 2.3** *La entropía relativa o distancia de Kullback-Lieber entre dos probabilidades de masa  $p(x)$  y  $q(x)$  se define como:*

$$D(p||q) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)}$$

La definición 2.3 utiliza la convención que  $0 \log_0^0 = 0$ ,  $0 \log_q^0 = 0$  y  $p \log_0^p = \infty$ . Se debe tener en cuenta que esta medida no es una métrica real, pues no es simétrica y no satisface la desigualdad triangular.

## 2.5.3. Estimación no-paramétrica de Medidas de Información

Si  $X$  es un vector aleatorio tomando valores en  $\mathbb{R}^d$ ,  $f(x)$  y  $g(x)$  son funciones de densidades de probabilidad del vector aleatorio. Entonces, se define la entropía diferencial y la divergencia de Kullback-Lieber [30], [33], [18]:

**Definición 2.4** *Entropía Diferencial:*

$$H(f) = - \int f(x) \log(f(x)) dx = -E\{\log(f(x))\}$$

**Definición 2.5** *Divergencia de Kullback-Lieber:*

$$D_{KL}(f||g) = \int f(x) \log\left(\frac{f(x)}{g(x)}\right) dx = E\left\{\log\left(\frac{f(x)}{g(x)}\right)\right\}$$

Se asume que  $H(f)$  y  $D_{KL}(f||g)$  están bien definidas y son finitas.

La dificultad surge cuando se quieren estimar estas cantidades, pero sin conocer exactamente las funciones de densidades de probabilidades asociadas a la variable aleatoria. Si se tiene acceso a muestras  $X_1, \dots, X_n$  del vector aleatorio entonces se puede obtener una

aproximación discreta y estimar la entropía [33] y la divergencia [18], por las ecuaciones 2.39 y 2.40 respectivamente:

$$H(f) \approx H(\hat{f}) = - \sum_{i=1}^n \hat{f}(X_i) \log(\hat{f}(X_i)) \quad (2.39)$$

$$D_{KL}(f||g) \approx D_{KL}(\hat{f}||\hat{g}) = \sum_{i=1}^n \hat{f}(X_i) \log\left(\frac{\hat{f}(X_i)}{\hat{g}(X_i)}\right) \quad (2.40)$$

Donde  $\hat{f}$  y  $\hat{g}$  son estimaciones de las densidades de interés, que pueden obtenerse por diferentes métodos de reconstrucción de densidades, desde una distribución empírica discreta.

Un método que permite obtener estas densidades estimadas es la estimación de densidad de Kernel, como se explica en la siguiente sección.

## 2.6. Estimación de densidad de Kernel

En estadística, el concepto de función de densidad de probabilidad es crucial. Su objetivo, principalmente, radica en condensar información acerca del comportamiento de una variable aleatoria.

Introducido Originalmente en 1962, por Emanuel Parzen [27], la Estimación de Densidad de Kernel (KDE - del Inglés *Kernel Density Estimation*) [17] es un método no-paramétrico que permite estimar la función de densidad de probabilidad de una variable aleatoria, por medio de una cantidad finita de muestras. Formalmente, dada una secuencia de variables aleatorias i.i.d. (del inglés *independent identically distributed*)  $\{X_1, X_2, \dots, X_n\}$ , con función de densidad de probabilidad  $f(x)$  (p.d.f - del Inglés *probability density function*) desconocida, pero común, entonces, se define el estimador  $\hat{f}(x)$  de  $f(x)$ :

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x - X_i}{h}\right) \quad (2.41)$$

Donde  $K(\cdot)$  es una función Kernel -no negativa, que integra uno y tiene media cero- y  $h > 0$  es un parámetro suavizante llamado *bandwidth* o, en español, ancho de banda. En general existe un amplio rango de funciones Kernel ha ser utilizadas: triangular, uniforme, Epanechnikov, normal, dentro de otras.

### 2.6.1. Selección del parámetro de suavizado

En miras a construir una estimación de la densidad de los datos observados es necesario escoger un parámetros de ancho de banda  $h$ . En general, una medida de la efectividad al estimar  $f$  por medio de  $\hat{f}$  se obtiene mediante el error cuadrático medio integrado (MISE) [5]:

$$MISE(\hat{f}) = E\left\{ \int [\hat{f}(y) - f(y)]^2 dy \right\} \quad (2.42)$$



2.42 se puede aproximar por la ecuación 2.43:

$$MISE(\hat{f}) \approx \frac{1}{4}h^4\sigma_K^4 \int f''(y)dy + \frac{1}{nh}\alpha(K) \quad (2.43)$$

Donde  $\alpha(K) = \int K^2(z)dz$  y  $\sigma_K^4 = \int z^2K(z)dz$  denota la varianza de la función Kernel. Considerando la aproximación de 2.43 se puede mostrar que el valor óptimo de  $h$  que minimiza el MISE en un sentido asintótico es [9]:

$$h_{opt} = \left\{ \frac{\gamma(K)}{\beta(f)n} \right\}^{\frac{1}{5}}$$

Donde  $\gamma(K) = \alpha(K)/\sigma_K^4$ , y  $\beta(f) = \int f''(y)^2dy$ . De esta forma, es claro que no es posible encontrar el ancho de banda óptimo, pues es necesario saber la pdf que se quiere estimar, y si ésta se tuviera entonces no existiría la necesidad de estimarla. Sin embargo, es posible obtener información acerca de como decae  $h$  con el número de muestras, propocionalmente a  $n^{1/5}$ , además es posible cuantificar el efecto de la curvatura de  $f$  por medio del factor  $\beta(f)$

### Ancho de Banda óptimo para Kernel Normal

En el caso particular cuando  $f$  es una densidad normal, entonces se puede aproximar  $h$  mediante la simple fórmula 2.44:

$$h = \left( \frac{4}{3n} \right)^{\frac{1}{5}} \sigma \quad (2.44)$$

Donde  $\sigma$  refiere a la desviación standard de la distribución. Si bien el supuesto es asumir la Gaussianidad de la función  $f$  a estimar, en general ésta resulta una aproximación apropiada para distribuciones unimodales [9].

## 2.7. Baterías de Ion-Litio

Para que cualquier sistemas energizado por baterías funcione correctamente es necesario garantizar protección y condiciones óptimas de operación en la batería. Como respuesta a este desafío nace el concepto de sistemas de gestión de batería (BMS - del inglés Battery Management Systems) [7], estos tienen por objetivo alcanzar aquellas condiciones de operación en la batería, por medio del seguimiento de ciertas variables que capturan el estado actual de la ESD. Algunas de ellas son; Estado de Carga (*SoC*) [7], Estado de Salud (SoH - del inglés State-of-Health)[7], vida útil remanente (RUL - del inglés *Remaining Useful Life*) [26]. Lamentablemente, en general, no es posible medir en línea estas variables directamente, razón por la cual los sistemas BMS deben incorporar rutinas de estimación y pronóstico en tiempo real para poder coordinar el sistema en conjunto, y así lograr su objetivo.

Para resolver el problema de la estimación y pronóstico de estas variables existe una amplia variedad de métodos en la literatura [12]. Estos permiten modelar la batería, tanto en aplicaciones fuera de línea; modelos electroquímicos, por ejemplo- como implementaciones

en línea que permiten representar la batería; representaciones de circuito abierto (OCV - del inglés *Open Circuit Voltage*). Sin embargo, estos métodos muestran ser poco apropiados para estimación en tiempo real, pues necesitan de largos periodos de descanso para las baterías. Por otra parte, existen métodos que utilizan equipamiento especializado para medir ciertas variables de la batería; *Electrochemical Impedance Spectroscopy* (EIS). Pero este tipo de equipamiento es muy costoso, por lo que es poco factible su implementación en la práctica.

Considerando lo expuesto anteriormente, un caso de particular interés en relación a la implementación de algoritmos de estimación y pronóstico esta asociado al estado de carga en baterías. En este escenario el desafío es obtener algoritmos simplificados que funcionen en tiempo real, restringiendo el gasto energético en que incurrir e intentando mantener los niveles de exactitud y precisión de la solución no simplificada.

### 2.7.1. Representación en Espacio-Estado

En relación a los diferentes métodos que existen para realizar estimación, el enfoque Bayesiano muestra ventajas al ser capaz de incorporar información a priori sobre la dinámica del sistema directamente por medio del modelo, además de ser capaz de introducir las incertidumbres asociadas a dichas representaciones. También es posible determinar un modelo de observaciones, relacionado a la representación en espacio-estado y, de esta forma, permitir la convergencia de los estimadores a través del cómputo de la verosimilitud de datos medidos secuencialmente.

El siguiente modelo para la batería [28], es una representación en espacio-estado de celdas de ion-litio:

- **Modelo de Transición de Estados**

$$x_1(k+1) = x_1(k) + w_1(k) \quad (2.45)$$

$$x_2(k+1) = x_2(k) - \frac{\hat{v}(x_1(k), x_2(k), u(k)) \cdot u(k) \cdot T_s}{E_{crit}} + w_2(k) \quad (2.46)$$

- **Modelo de Observación**

$$y(k) = \hat{v}(x_1(k), x_2(k), u(k)) + \eta(k) \quad (2.47)$$

con

$$\begin{aligned} \hat{v}(x_1(k), x_2(k), u(k)) = & v_L + (v_0 - v_L) \cdot e^{\gamma \cdot (x_2(k)-1)} + \dots \\ & \alpha \cdot v_L \cdot (x_2(k) - 1) + (1 - \alpha) \cdot v_L \cdot (e^{-\beta} - e^{-\beta \cdot \sqrt{x_2(k)}}) \dots \\ & - u(k) \cdot x_1(k) \end{aligned} \quad (2.48)$$

Donde  $w_1(k) \sim \mathcal{N}(0, \sigma_1)$  y  $w_2(k) \sim \mathcal{N}(0, \sigma_2)$  corresponden a ruidos aditivos Gaussianos, mientras que  $\eta(k) \sim \mathcal{N}(0, \sigma_{obs})$  representa la incertidumbre en las mediciones. El tiempo de muestreo es  $T_s[\text{sec}]$  y  $u(k) = i(k)[A]$  es la corriente de descarga, variable que se considera entrada al sistema. Finalmente  $y(k) = v(k)[V]$  es el voltaje de la batería y esta variable se

considera la salida del sistema. El estado  $x_1$  representa la impedancia interna de la batería, mientras que  $x_2$  representa el estado de carga de la batería. Finalmente, todos los demás parámetros de la batería pueden determinarse según el procedimiento descrito en [28], [12].

# Capítulo 3

## Metodología de Comparación para algoritmos basados en SMC

En general, el desafío de comparar algoritmos que poseen un comportamiento estocástico intrínscico puede ser abordado desde diferentes perspectivas. Si bien algunos análisis más conservadores proponen considerar el peor escenario posible para cuantificar el desempeño -o la pérdida de desempeño- asociado a algún algoritmo [8], esta perspectiva no entrega necesariamente la comparación más adecuada para la interpretación de resultados, de hecho el peor caso posible entrega un resultado que no tiene posibilidad de ocurrir. En este sentido, se puede decidir por un esquema más flexible que permita definir el peor escenario posible de algún algoritmo en probabilidad. De esta forma se puede estudiar el desempeño del algoritmo acercándose arbitrariamente al peor escenario posibles, pero siempre considerando que el peor caso tiene probabilidad nula de ocurrencia, por lo tanto es poco realista utilizarlo; ¿Qué ocurre si un algoritmo  $D_1$  tiene un peor desempeño que otro  $D_2$  en cuanto al peor caso posible, mientras que  $D_1$  tiene mejor desempeño que  $D_2$  el 99,99% de las veces?.

Otro punto importante al momento de realizar comparaciones entre algoritmos, o aproximaciones, que entregan descripciones estadísticas de eventos son las medidas que se van a utilizar para compararlos. Un primer enfoque puede sugerir contrastar los primeros momentos de estas distribuciones [32]. Pero si se considera que el hecho de contrastar diferentes aproximaciones estadísticas de un evento, con respecto a una descripción referencial del evento, es un problema que puede interpretarse desde las comunicaciones; el decodificar un mensaje ideal entregado por un emisor (algoritmo de referencia), mediante una aproximación de aquel mensaje (salida del algoritmo simplificado), entonces tiene sentido utilizar medidas de información para contrastar el desempeño de cada algoritmo según cómo aproxima al mensaje ideal. Es por esta razón que la metodología que se explica en este capítulo propone integrar medidas de la información como formas naturales de medir la discrepancia que entregan distintas representaciones de algún evento, dependiendo del algoritmo que se utilice.

En este capítulo de la Tesis, se explica y propone un método que permite comparar diferentes aproximaciones al desafío de estimar la evolución de un sistema dinámico, por medio de una representación estadística. En primera instancia se recuerda el problema que resuelve el análisis PACC y se trae al contexto de métodos secuenciales de Monte Carlo. Luego, se

explica y definen medidas de la Información y cómo pueden ser aplicadas a algoritmos que entregan una descripción estadística de la evolución temporal de un sistema y cómo estas medidas pueden ser utilizadas como funciones de pérdida para asociarlas a un análisis PACC de los algoritmos. Finalmente, se explica cómo aproximar las medidas de la información, considerando la imposibilidad de calcularlas directamente según las descripciones que entrega cada algoritmo, y se presenta un esquema de comparación.

### 3.1. Análisis PACC en Métodos Secuenciales de Monte Carlo

Como se explica en la Sección 2.1, la metodología PACC es una herramienta útil al momento de comparar algoritmos; permite estimar la pérdida en performance que se obtiene producto de modificar o simplificar algún algoritmo.

En este sentido, se puede utilizar la metodología PACC para comparar diferentes aproximaciones al problema que soluciona Monte Carlo Secuencial, y así contrastar el desempeño de diferentes algoritmos.

Recordando el problema que aborda PACC; la estimación de  $p_\gamma$ , donde  $p_\gamma$  es la probabilidad de que la pérdida de performance sea satisfecha a cierto nivel  $\gamma$ :

$$Pr\{|p_\gamma - \hat{p}_N| < \varepsilon\} > (1 - \delta) \quad (3.1)$$

Donde, considerando el escenario de comparar algoritmos que aproximan la evolución de un sistema mediante una descripción estadística:

- $\pi_n(x_{1:n})$  = Distribución objetivo. Esta descripción representa al algoritmo de referencia.
- $\hat{\pi}_n(x_{1:n})$  = Aproximación de la distribución objetivo. Esta descripción representa a un algoritmo simplificado.
- $p_\gamma = Pr(u(\pi_n(x_{1:n}), \hat{\pi}_n(x_{1:n})) < \gamma)$  donde  $u$  es una función de costo (loss function), y depende del problema que se está tratando, como se explica en la siguiente Sección 3.2. Además,  $\bar{\gamma}$  es el menor valor de  $\gamma$  tal que  $u(\pi_n(x_{1:n}), \hat{\pi}_n(x_{1:n})) < \bar{\gamma}$  se cumple  $\forall x_{1:n} \in \mathbb{R}^n$
- $\hat{p}_N = \frac{1}{N} \sum I(\pi_n(x_{1:n}), \hat{\pi}_n(x_{1:n}))$  Donde

$$I(\pi_n(x_{1:n}), \hat{\pi}_n(x_{1:n})) = \begin{cases} 1 & \text{Si } u(\pi_n(x_{1:n}), \hat{\pi}_n(x_{1:n})) < \gamma \\ 0 & \text{Si } u(\pi_n(x_{1:n}), \hat{\pi}_n(x_{1:n})) > \gamma \end{cases} \quad (3.2)$$

De esta manera, y recordando lo que se explica en la Sección 2.1, es posible utilizar la metodología PACC para comparar diferentes algoritmos que aproximan la distribución objetivo  $\pi_n(x_{1:n})$ . Entonces, al determinar  $\hat{\gamma}$  como el mínimo valor de  $\gamma$  tal que  $\hat{p}_N(\hat{\gamma}) = 1$ ,  $\forall \gamma \geq \hat{\gamma}$ . Luego  $\hat{\gamma}$  (máximo error empírico) es una aproximación de  $\bar{\gamma}$  (máximo error teórico) que depende del número de muestras  $N_C$ , y se puede utilizar  $N_C$  según la cota de Chernoff;  $N_C \geq \frac{\ln \frac{2}{2\varepsilon^2}}$ . Considerando lo anterior, el valor que se obtiene para  $\hat{\gamma}$  es un indicador de la exactitud computacional para el algoritmo simplificado. Y como se explica más adelante, si se utilizan medidas de información como *loss functions*, entonces el máximo error empírico

puede asociarse, por ejemplo, a pérdida de información en la que incurre cierto algoritmo, o a analizar el manejo de la incertidumbre según distintas aproximaciones.

Si  $D_1$  (algoritmo simplificado 1) -que entrega  $\hat{\pi}_n^1(x_{1:n})$ - obtiene una pérdida en performance  $\hat{\gamma}_1$  y  $D_2$  (algoritmo simplificado 2) -que entrega  $\hat{\pi}_n^2(x_{1:n})$ - una pérdida en performance  $\hat{\gamma}_2$ , entonces:

La exactitud computacional de  $D_1$  es mayor que la de  $D_2 \iff \hat{\gamma}_1 < \hat{\gamma}_2$ .

## 3.2. Medidas de Información en Métodos secuenciales de Monte Carlo

En relación a los algoritmos basados en métodos secuenciales de Monte Carlo, es importante recordar que estos generan una estimación del estado del sistema mediante la aproximación de una distribución de probabilidades. En consecuencia, para cada instante de tiempo, se obtiene una colección de muestras con pesos - *i.e.* partículas - que representan la distribución objetivo.

Llevando esto al problema de comunicación, se puede pensar en una fuente ideal, donde el mensaje que ésta fuente emite es la distribución objetivo  $\pi_n(x_{1:n})$ , mientras que los algoritmos basados en métodos secuenciales de Monte Carlo son receptores de aquel mensaje que decodifican  $\hat{\pi}_n(x_{1:n})$ . En este sentido, las diferentes simplificaciones que utilizan estos algoritmos para tratar de aproximar la distribución objetivo se pueden conceptualizar como el error que corrompe al mensaje durante la transmisión.

Al pensar el problema desde esta perspectiva, medidas de información como la Divergencia y Entropía permiten cuantificar la pérdida en información que se produce al simplificar estos algoritmos. Además, si se utilizan estas medidas como *loss functions* en conjunto con la metodología PACC (Sección 2.1), entonces es posible cuantificar en probabilidad la pérdida en performance de los algoritmos, asociándolo a pérdida de información. En particular:

Consideremos las siguiente definiciones:

- $\pi_n(x_n) :=$  La distribución marginal objetivo asociada al emisor ideal que define el estado del sistema en el instante  $n$ .
- $\hat{\pi}_n(x_n) :=$  La distribución marginal que se obtiene por medio del algoritmo simplificado para el estado de carga, en el instante  $n$ .
- $T :=$  Cantidad total de instantes en los que se realiza estimación o predicción del estado del sistema,  $n \in 1, \dots, T$ .

Se introducen las siguientes medidas para cuantificar la pérdida en el desempeño de los algoritmos:

**Definición 3.1** *Pérdida promedio de información:*

$$\bar{D}_{KL}(\pi_{1:T}(x_{1:T}) || \hat{\pi}_{1:T}(x_{1:T})) = \frac{1}{T} \sum_{k=1}^T D_{KL}(\pi_k(x_k) || \hat{\pi}_k(x_k))$$

Por medio de *Pérdida promedio de información* se cuantifica la cantidad de información perdida al aproximar la distribución objetivo, por alguna versión simplificada de ella. Con respecto a la entropía, parece interesante observar la evolución de  $H$  durante el proceso de estimación y predicción y comparar tanto la diferencia en entropía, como la diferencia de evolución de la entropía para ambos algoritmos y, de esta manera, medir la incerteza que caracteriza la evolución de la estimación y predicción según la aproximación que se implemente.

**Definición 3.2** *Incertidumbre promedio:*

$$\bar{H}(\hat{\pi}_{1:T}(x_{1:T})) = \frac{1}{T} \sum_{k=1}^T H(\hat{\pi}_k(x_k))$$

Además de medir la incertidumbre asociada a la distribución, es interesante medir la diferencia absoluta entre la incertidumbre asociada a cada aproximación y la distribución objetivo. De esta forma, se puede incorporar la *diferencia absoluta en entropía* como una forma de cuantificar la calidad, en cuanto a manejo de la incertidumbre, de las diferentes aproximaciones.

**Definición 3.3** *Diferencia Absoluta en Incertidumbre promedio:*

$$\Delta(\bar{H}(\pi_{1:T}(x_{1:T}), \hat{\pi}_{1:T}(x_{1:T}))) = |\bar{H}(\pi_{1:T}(x_{1:T})) - \bar{H}(\hat{\pi}_{1:T}(x_{1:T}))|$$

### 3.3. Estimación de Medidas de la Información en SMC

En general, la distribución objetivo en el instante  $n$  -  $\pi_n(x_n)$  - es un función de densidad de probabilidad continua que representa al estado del sistema. En este sentido, si se tiene acceso a una representación continua de las distribuciones marginales  $\pi_n(x_n)$  y  $\hat{\pi}_n(x_n)$  se pueden calcular las medidas de información:

$$H(\pi_n(x_n)) = - \int \pi_n(x_n) \log(\pi_n(x_n)) dx_n \quad (3.3)$$

$$D_{KL}(\pi_n(x_n) || \hat{\pi}_n(x_n)) = \int \pi_n(x_n) \log\left(\frac{\pi_n(x_n)}{\hat{\pi}_n(x_n)}\right) dx_n \quad (3.4)$$

Sin embargo, la mayoría de las veces no se tiene acceso a  $\pi_n(x_n)$ , sólo a aproximaciones de esta distribución basadas en pesos y partículas. En este escenario, no es posible calcular directamente 3.3 ni 3.4. Frente a esta dificultad se propone, en primer lugar, aproximar la distribución  $\pi_n(x_n)$  por medio de  $\pi_n^*(x_n)$ , ésta distribución -que también se obtiene mediante una aproximación de SMC- se asume como *Ground Truth* y será la referencia para comparar contra otras aproximaciones de SMC, es decir  $\pi_n^*(x_n)$  se asume como el mensaje ideal de la fuente, mientras que  $\hat{\pi}_n(x_n)$  (aproximaciones más simplificadas de  $\pi_n(x_n)$ ) representa el

mensaje que capta el receptor, y lo que se intenta medir es la discrepancia entre aquellas representaciones.

Considerando que se asume una aproximación de  $\pi_n(x_n)$  que representan el *Ground Truth* 3.5, y otras aproximaciones de SMC mediante diferentes métodos con simplificaciones más significativas 3.6:

$$\pi_n(x_n) \approx \pi_n^*(x_n) = \sum_{i=1}^{N1} W_n^{*i} \delta_{X_n^{*i}(x_n)} \quad (3.5)$$

$$\pi_n(x_n) \approx \hat{\pi}_n(x_n) = \sum_{i=1}^{N2} W_n^i \delta_{X_n^i(x_n)} \quad (3.6)$$

Al ser estas representaciones discretas, entonces, en una primera instancia se puede intentar aproximar la divergencia por 3.7:

$$D_{KL}(\pi_n(x_n) || \hat{\pi}_n(x_n)) \approx D_{KL}(\pi_n^*(x_n) || \hat{\pi}_n(x_n)) = \sum_{i=1}^n \pi_n^*(X_n^{*i}) \log \left( \frac{\pi_n^*(X_n^{*i})}{\hat{\pi}_n(X_n^i)} \right) \quad (3.7)$$

Esta aproximación presenta dos problemas. Por un lado, las partículas asociadas a  $\pi_n^*(x_n)$  son  $\{X_n^{*1}, \dots, X_n^{*N1}\}$  mientras que para  $\hat{\pi}_n(x_n)$  son  $\{X_n^1, \dots, X_n^{N2}\}$ . Como consecuencia de las diferentes partículas que representan estas distribuciones, no es posible determinar el super índice de la sumatoria si  $N1 \neq N2$ . Además, en el mejor de los casos, cuando  $N1 = N2$ , no se puede asegurar que  $X_n^{*i} = X_n^i, \forall i \in \{1, \dots, n\}$ , por lo que no es directo el cálculo del término  $\log \left( \frac{\pi_n^*(X_n^{*i})}{\hat{\pi}_n(X_n^i)} \right)$ . Considerando estas dificultades se propone, antes de realizar la estimación no-paramétrica de la entropía y la divergencia, un paso intermedio que permita reconstruir pdf continuas de las representaciones con partículas de  $\pi_n^*(x_n)$  y  $\hat{\pi}_n(x_n)$ , y luego muestras de ellas escogiendo  $\{\tilde{X}_n^1, \dots, \tilde{X}_n^M\}$  para evaluar las pdf y, en consecuencia, sea posible estimar la divergencia y entropía. Entonces se construyen las siguientes aproximaciones por medio de KDE para  $\pi_n^*(x_n)$  y  $\hat{\pi}_n(x_n)$ , respectivamente 3.8 y 3.9:

$$\pi_n^*(x_n) \approx \tilde{\pi}_n^*(x_n) = \frac{1}{nh} \sum_{i=1}^n K \left( \frac{x_n - X_n^{*i}}{h} \right) \quad (3.8)$$

$$\hat{\pi}_n(x_n) \approx \tilde{\pi}_n(x_n) = \frac{1}{nh} \sum_{i=1}^n K \left( \frac{x_n - X_n^i}{h} \right) \quad (3.9)$$

De esta forma ahora es posible determinar un rango de interés para evaluar las pdf estimadas de la distribución objetivo, escogiendo las muestras  $\{\tilde{X}_n^1, \dots, \tilde{X}_n^M\}$ . Lo que permite evitar las dificultades explicadas anteriormente en cuanto a las diferencia entre  $\{X_n^{*1}, \dots, X_n^{*N1}\}$  y  $\{X_n^1, \dots, X_n^{N2}\}$ . En este contexto,  $M$  es el número de muestras, y es un parámetro que se puede modificar según el costo computacional de calcular estas aproximaciones y la resolución con la que se quieren evaluar. Además, permite evaluar la zona de interés donde se concentra la información para la variable  $x_n$ . De esta manera, se estima la divergencia 3.10 y la entropía 3.11, 3.12:



$$D_{KL}(\pi_n(x_n)||\hat{\pi}_n(x_n)) \approx D_{KL}(\tilde{\pi}_n^*(x_n)||\tilde{\pi}_n(x_n)) = \sum_{i=1}^M \tilde{\pi}_n^*(\tilde{X}_n^i) \log\left(\frac{\tilde{\pi}_n^*(\tilde{X}_n^i)}{\tilde{\pi}_n(\tilde{X}_n^i)}\right) \quad (3.10)$$

$$H(\hat{\pi}_n(x_n)) \approx H(\tilde{\pi}_n(x_n)) = - \sum_{i=1}^M \tilde{\pi}_n(\tilde{X}_n^i) \log(\tilde{\pi}_n(\tilde{X}_n^i)) \quad (3.11)$$

$$H(\pi_n^*(x_n)) \approx H(\tilde{\pi}_n^*(x_n)) = - \sum_{i=1}^M \tilde{\pi}_n^*(\tilde{X}_n^i) \log(\tilde{\pi}_n^*(\tilde{X}_n^i)) \quad (3.12)$$

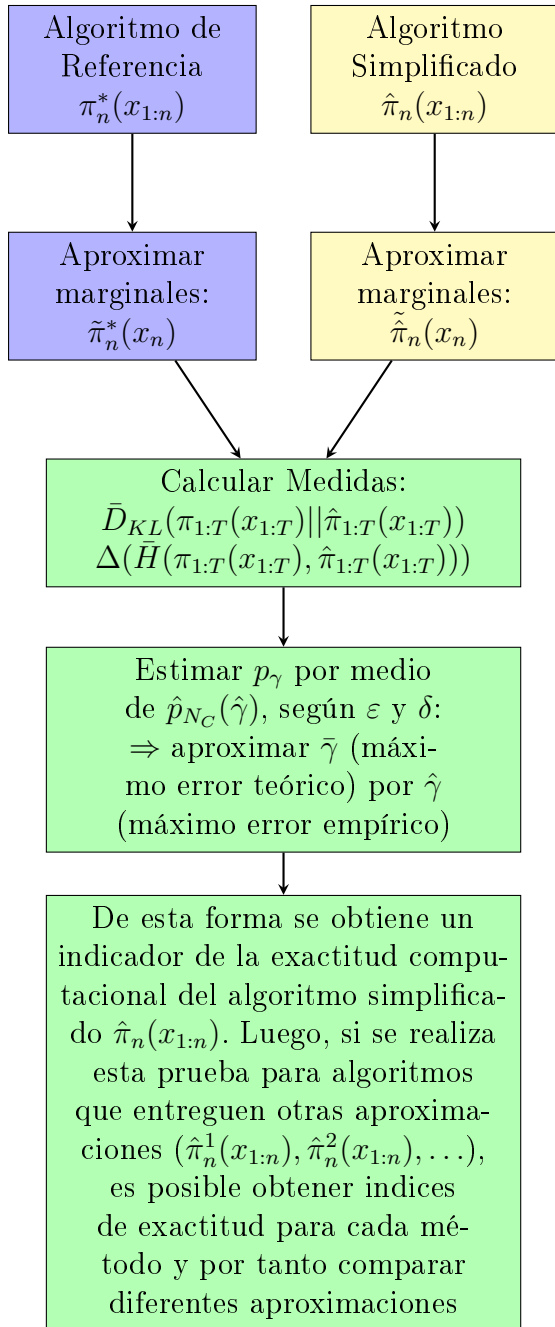
Además, evaluando la pdf en la misma cantidad de puntos y en las mismas posiciones, se puede contrastar de una manera más justa la concentración de información que entregan distintas aproximación.

### 3.4. Esquema Comparativo

Continuando con el esquema comparativo que se propone, éste cuenta con una primera instancia, donde se obtiene una realización -según el algoritmo de referencia- de la evolución del sistema dinámico; se considera sólo una realización de este algoritmo pues se supone que este representa el *Ground Truth* de la evolución del sistema. Luego se obtienen  $N_C$  realizaciones, según el algoritmos simplificado, de la evolución del sistema. Considerando estas realizaciones y utilizando KDE, es posible reconstruir las distribuciones marginales del algoritmo de referencia, como también las distribuciones marginales del algoritmo simplificado para cada una de sus realizaciones. De esta manera, se pueden calcular aproximaciones no-paramétricas de la entropía y divergencia, lo que permite el cálculo de las funciones de pérdida; *Diferencia absoluta en Incertidumbre promedio* y *Pérdida promedio de Información*.

Hasta el momento, el cálculo de las funciones de pérdida entre el algoritmo de referencia y una de las realizaciones del algoritmo simplificado, entregan una realización de cada función de pérdida. Entonces, al disponer de  $N_C$  realizaciones del algoritmo simplificado, se pueden obtener  $N_C$  realizaciones de la función de pérdida y, en consecuencia, es posible aproximar la probabilidad de máximo error teórico ( $p_\gamma$ ) a algún nivel  $\bar{\gamma}$ , por la probabilidad empírica  $\hat{p}_{N_C}$ . Donde  $\hat{p}_{N_C}$  depende del máximo error empírico  $\hat{\gamma}$ , quien, finalmente, es una aproximación del máximo error teórico  $\bar{\gamma}$ , por lo que entrega un indicador del desempeño, o exactitud, asociado al algoritmo simplificado.

### 3.4.1. Procedimiento de Comparación



Obtener realización del Algoritmo de referencia y del algoritmo simplificado

Aproximar mediante KDE. Así es posible estimar no-paramétricamente  $H$  y  $D_{KL}$

Calcular las funciones de pérdida:  $u(\tilde{\pi}_n^*(x_{1:n}), \tilde{\pi}_n(x_{1:n}))$

Obtener  $N_C$  realizaciones del algoritmo simplificado  $\tilde{\pi}_n(x_{1:n})$ . Así calcular  $N_C$  realizaciones de  $u(\tilde{\pi}_n^*(x_{1:n}), \tilde{\pi}_n(x_{1:n}))$

## 3.5. Comentarios Sobre el Esquema Comparativo

Considerando la metodología propuesta, es importante tener en cuenta los siguientes aspectos:

- Ya que no es posible tener acceso a la distribución objetivo  $\pi_n(x_{1:n})$ , es necesario definir un *Ground Truth*;  $\pi_n^*(x_{1:n})$ . En este sentido, para esta metodología, es importante la elección del algoritmo de referencia, pues basándose en la calidad de éste resultado es que serán contrastadas diferentes aproximaciones por otros métodos.
- Por otra parte, este método presenta varias ventajas con respecto a métodos anteriores que han sido aplicados para contrastar algoritmos. En primer lugar, presenta un marco estadístico apropiado para comparar algoritmos que presentan descripciones estadísticas de la evolución de algún proceso, pues estos algoritmos entregan diferentes resultados aún cuando dispongan de los mismos datos observados y mismas condiciones iniciales, por lo que parece sensato considerar una metodología que entregue una aproximación estadística del mayor error en el que incurre algún algoritmo. Finalmente, lo importante es asegurar el buen funcionamiento de los algoritmos en probabilidad (con niveles de confianza y precisión arbitrariamente escogidos) y no frente al cien por ciento de los casos; lo que implica un análisis muy estricto y poco realista, además del desafío teórico. En segundo lugar, el incorporar medidas de información, a diferencia de analizar los primeros momentos en las distribuciones marginales de cada algoritmo, permite capturar mayor información estadística en cuanto a la discrepancia o similitud que presentan diferentes algoritmos (es decir, distintas aproximaciones estadísticas para la evolución de un sistema dinámico), como por ejemplo la cantidad de información que se pierde entre un algoritmo y otro, a diferencia de calcular la discrepancia en la esperanza. O la capacidad de contrastar la discrepancia en la entropía para diferentes aproximaciones, en lugar de medir solamente la evolución de la varianza al comparar.
- También es interesante destacar que por medio de la metodología propuesta se puede cuantificar el efecto que produce modificar distintas variables o parámetros que definen a cierto algoritmo simplificado, por ejemplo, el número de partículas o la distribución inicial del estado. Cómo también el efecto de modificar subrutinas del algoritmo, como lazos de corrección externa, esquemas de resampling, etc.

## Capítulo 4

# Implementación de Esquema Comparativo en Algoritmos de Estimación y Pronóstico: Baterías de Ion-Litio

Un elemento clave para la apropiada implementación de sistemas de administración de pronóstico y salud (PHM) es la correcta certificación de rutinas que predicen y estiman el estado del sistema de interés [1]. En este sentido, durante largo tiempo se han dedicado grandes esfuerzos a la creación de algoritmos que permitan estimar y predecir la evolución de sistemas dinámicos. Sin embargo, el trabajo que se ha dedicado a definir un marco adecuado para comparar los diferentes enfoques aún es insuficiente.

Por otra parte, la comunidad de Prognostics and Health Management (PHM) ha escogido a los algoritmos basados en SMC como los algoritmos de facto del estado-del-arte en esta materia [21], pues ofrecen una alternativa eficiente frente a la simulación de Monte Carlo en la implementación de esquemas de pronóstico en tiempo-real. Sin embargo, aún quedan muchas interrogantes por responder al momento de implementar un PF y al mesurar su desempeño, más aún al intentar contrastar PF con otros métodos que también buscan solucionar el mismo desafío de diagnóstico y pronóstico.

En este capítulo de la Tesis, se implementa la metodología propuesta para medir el desempeño asociado a distintas aproximaciones que permiten enfrentar la problemática de estimar y predecir el estado de un sistema dinámico. Para el caso de estudio se utilizan datos de baterías de Ion-Litio y el objetivo es, en primer lugar, la definición e implementación de un algoritmo de referencia para ambos casos; estimación y pronóstico. Luego, se destinan esfuerzos al desafío de medir -incluyendo medidas que cuestionan el paradigma actual, el cual no incorpora una completa descripción estadística de los estados- el desempeño de distintas estrategias, tanto para el problema de diagnóstico, como para el de pronóstico.

## 4.1. Metodología comparativa en Estimación

### 4.1.1. Datos y Parámetros del Modelo

Los valores de corriente y voltaje utilizados corresponden a la caracterización del uso de un vehículo eléctrico dentro de la ciudad (*Federal Urban Driving Schedule*), apropiadamente escalado al caso de una sola celda de ion-litio [13].

En la Figura 4.1 se puede observar tanto el perfil de corriente, como el de voltaje asociado al set de datos. Tomando en cuenta el modelo para baterías presentado en la Subsección

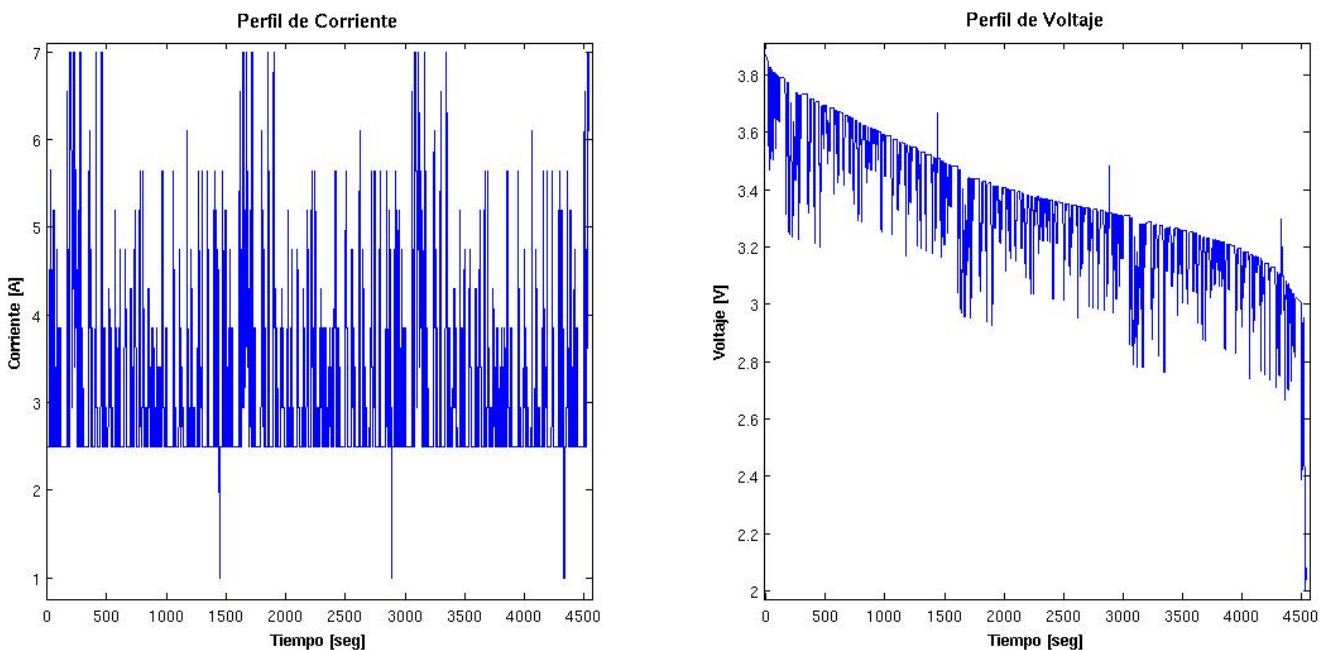


Figura 4.1: Perfiles voltaje y corriente FUDS

2.7.1, y el procedimiento de [28] para estimar los parámetros, se obtienen la caracterización del modelo de la batería según los valores de la Tabla 4.1.

### 4.1.2. Algoritmo de Referencia para Estimación

Como se ha explicado hasta el momento, en general, no se tiene acceso a la distribución objetivo  $\pi_n(x_{1:n})$ . Este es el caso también para el desafío de estimar el estado de carga en baterías de ion-litio. Dada esta situación, surge de manera natural la interrogante sobre cómo definir o construir una aproximación de  $\pi_n(x_{1:n}) \approx \pi_n^*(x_{1:n})$  tal que pueda utilizarse como marco de referencia para realizar un estudio comparativo sobre distintas aproximaciones de aquella distribución. Si se recuerda lo expuesto en la Subsección 2.2.5, el problema que enfrenta el suavizado en SMC, es el de estimar la distribución del estado en un tiempo específico dadas todas las observaciones hasta un *futuro*. Esto es equivalente a incorporar toda la información

Tabla 4.1: Valores parámetros del modelo en espacio-estado para datos FUDS

Parámetro	Descripción	Valor
$E_{crt}$	Parámetro modelo de batería	46858
$T_s$	Parámetro modelo de batería	1
$v_L$	Parámetro modelo de batería	3.9974
$v_0$	Parámetro modelo de batería	4.144
$\alpha$	Parámetro modelo de batería	0.1469
$\beta$	Parámetro modelo de batería	17
$\gamma$	Parámetro modelo de batería	10.4954
$w_1(k)$	Ruido de proceso: estado $x_1$	$\sim \mathcal{N}(0, 0.0015)$
$w_2(k)$	Ruido de proceso: estado $x_2$	$\sim \mathcal{N}(0, 0.0055)$
$\eta(k)$	Ruido de observación	$\sim \mathcal{N}(0, 0.067)$
$r_0$	Resistencia interna experimental	0.1

a la que se tiene acceso del sistema (observaciones) para determinar la distribución *a posteriori* del estado del sistema en todos los instantes de tiempo. Asumiendo que la capacidad de incorporar mayor información para predecir se traduce en una distribución *a posteriori* más precisa sobre el estado del sistema, y considerando también la capacidad de las técnicas de Smoothing para incorporar toda esta información y luego realizar la estimación, es que se propone e implementa el algoritmo Forward Filter-Backward Smoothing para ser utilizado como el algoritmo de referencia al realizar las comparaciones.

En la primera etapa; *Forward Filter*, el algoritmo realiza el proceso de filtrado por medio de una rutina SIR (Algoritmo 4). Un aspecto relevante a considerar es el número de partículas, ya que la varianza del error de aproximación decrece a una tasa de  $\mathcal{O}(1/N)$  [14]. Considerando lo anterior, es que se decide realizar la etapa de filtrado con  $N = 10000$  partículas.

Otra factor a considerar, es la posibilidad de incluir una rutina de OFCL en la etapa de filtrado. En particular, se utiliza la rutina que se observa en el Algoritmo 9.

---

**Algorithm 9:** Outer Feedback Correction Loops

---

- 1 Si  $k_{min} < k < k_{max}$  :
  - 2  $e_{acum} = e_{acum} + |e_{obs}|$
  - 3 Si  $e_{acum} \leq e_{Th}$  :
  - 4  $std(w_1(k)) = \max(p_1 std(w_1(k)), std_1)$
  - 5  $std(w_2(k)) = \max(p_2 std(w_2(k)), std_2)$
  - 6 De otra forma :
  - 7  $e_{acum} = 0$
  - 8  $std(w_1(k)) = q_1 std(w_1(k))$
  - 9  $std(w_2(k)) = q_2 std(w_2(k))$
- 

Donde  $k_{min}$  y  $k_{max}$  indican los instantes durante los que se ejecuta la rutina de OFCL,  $e_{obs}$  es la diferencia entre la medición de la observación real del proceso y la esperada por el algoritmo; error de observación.  $e_{acum}$  cuantifica el error acumulado de los errores de observación  $e_{obs}$ , y se inicializa con valor cero.  $e_{Th}$  es el criterio para tomar la decisión sobre cómo modificar los ruidos de proceso.  $std_1$  y  $std_2$  son las cotas inferiores sobre las cuales no

es posible continuar disminuyendo los ruidos de procesos. Finalmente,  $p_1$  y  $p_2$  son constantes con valor entre cero y uno, mientras que  $q_1$  y  $q_2$  son constantes mayores que uno. Los valores específicos que toma la rutina de OFCL depende del algoritmo que se implementará. En particular, para la etapa de estimación mediante *Forward Filter* los valores son:

- $k_{min} = 200$ ,  $k_{max} = 1000$  y  $e_{Th} = 0.15$
- $p_1 = 0.99$ ,  $p_2 = 0.99$ ,  $q_1 = 1.10$  y  $q_2 = 1.01$
- $std_1 = 10^{-4}$  y  $std_2 = 10^{-4}$

En la segunda etapa; *Backward Smoothing*, el algoritmo realiza un suavizado de la estimación por medio de simulaciones en reversa (*Backward Simulation*). Es decir, ya que por medio de la etapa de *Forward Filter* se obtiene  $\hat{p}(x_{1:T}|y_{1:T})$ , entonces es posible muestrear una realización  $\hat{x}_{1:T} = (\hat{x}_1, \hat{x}_2, \dots, \hat{x}_T)$  del proceso, como se explica en el algoritmo 5. La rutina de muestreo en reversa es aplicada 10000, por lo que se obtienen 10000 realizaciones de  $\hat{x}_{1:T}$  que capturan la descripción estadística de la evolución del proceso tomando en cuenta toda las observaciones.

### 4.1.3. Algoritmos a Comparar

Con respecto a los algoritmos simplificados que se someten a la comparación, es importante destacar que son versiones modificadas de algoritmos que han venido desarrollandose hace años por estudiantes del departamento de Ingeniería Eléctrica de la Universidad de Chile que han trabajado en el desafío de la estimación y pronóstico del estado de carga en baterías de Ion-litio, bajo la guía del profesor Marcos Orchard. Es por esta razón que se escogen los algoritmos que se explican a continuación, con la intención de someterlos a un análisis comparativo que permita cuantificar la calidad de cada aproximación.

Además, se debe destacar que todos los algoritmos implementados consideran la misma modelación en espacio de estado para la batería. Lo que cambia es la cantidad de partículas o sigma points que representan la distribución del estado, cómo se escogen estas partículas y cómo se calculan los pesos asociados, como se describe en la Sección 2.2 para cada una de las distintas aproximaciones.

### Filtro de Partículas

Este algoritmo es una aproximación de SMC mediante una rutina de SIR (Algoritmo 4) con  $N = 50$  partículas. La implementación de este algoritmo incluye la rutina de OFCL (Algoritmo 9), con los siguiente parámetros:

- $k_{min} = 200$ ,  $k_{max} = 1000$  y  $e_{Th} = 0.15$
- $p_1 = 0.99$ ,  $p_2 = 0.99$ ,  $q_1 = 1.10$  y  $q_2 = 1.01$
- $std_1 = 10^{-4}$  y  $std_2 = 10^{-4}$

## Unscented Kalman Filter

Una observación importante es que UKF no es filtro de partículas (algoritmo SMC). En efecto, ésta es una descripción más simple y restrictiva del proceso. Este algoritmo realiza una aproximación de la evolución del sistema mediante el Algoritmo 2. Considerando que el estado del sistema es de dimensión dos, entonces se escogen  $N = 5$  sigma points.

Considerando que ésta es una aproximación más simple de la descripción estadística de la evolución del proceso, toma sentido evaluar el desempeño de ella por medio de la metodología propuesta. Para esta aproximación se implementan dos versiones; una sin rutina de OFCL y una que si incluye ésta rutina. Para la versión de UKF que contiene la rutina de OFCL (Algoritmo 9), se utilizan los siguiente parámetros:

- $k_{min} = 5$ ,  $k_{max} = 1000$  y  $e_{Th} = 0.15$
- $p_1 = 0.99$ ,  $p_2 = 0.98$ ,  $q_1 = 1.10$  y  $p_2 = 1.01$
- $std_1 = 10^{-5}$  y  $std_2 = 10^{-5}$

La implementación original se encuentra en [32].

## Condiciones Iniciales

Es importante mencionar que tanto al algoritmo *Forward Filter-Backward Smoothing*, como a las otras tres aproximaciones se le entrega la misma condición inicial para el estado del sistema:

- $x_1 \sim \mathcal{N}(0.10, 0.05)$
- $x_2 \sim \mathcal{N}(0.85, 0.10)$

### 4.1.4. Resultados de la comparación

Para realizar la comparación entre los 3 algoritmos señalados y el *Ground Truth*, se deben obtener  $N_C$  realizaciones de la evolución del sistema por cada algoritmo simplificado. Donde se puede escoger  $N_C$  según la cota de Chernoff 2.4. En consecuencia, si se quiere estimar la probabilidad del máximo error teórico con precisión  $\varepsilon = 0.05$  y confianza  $(1 - \delta) = 0.99$ , i.e.:

$$Pr\{|p_\gamma - \hat{p}_{N_C}| < 0.05\} > 0.99$$

Entonces se deben obtener  $N_C = 1060$  realizaciones.

Si se toma el 90% de los datos para realizar la estimación (comenzando en el instante que la batería esta cargada completamente) y se realiza la estimación del estado del sistema, entonces se puede obtener una descripción estadística de la evolución del *SoC* para la batería según las distintas aproximaciones a comparar. En la Figura 4.2 se observa una realización de la evolución de la Esperanza para el *SoC*, según los 3 algoritmos descritos anteriormente y el *Ground Truth*.



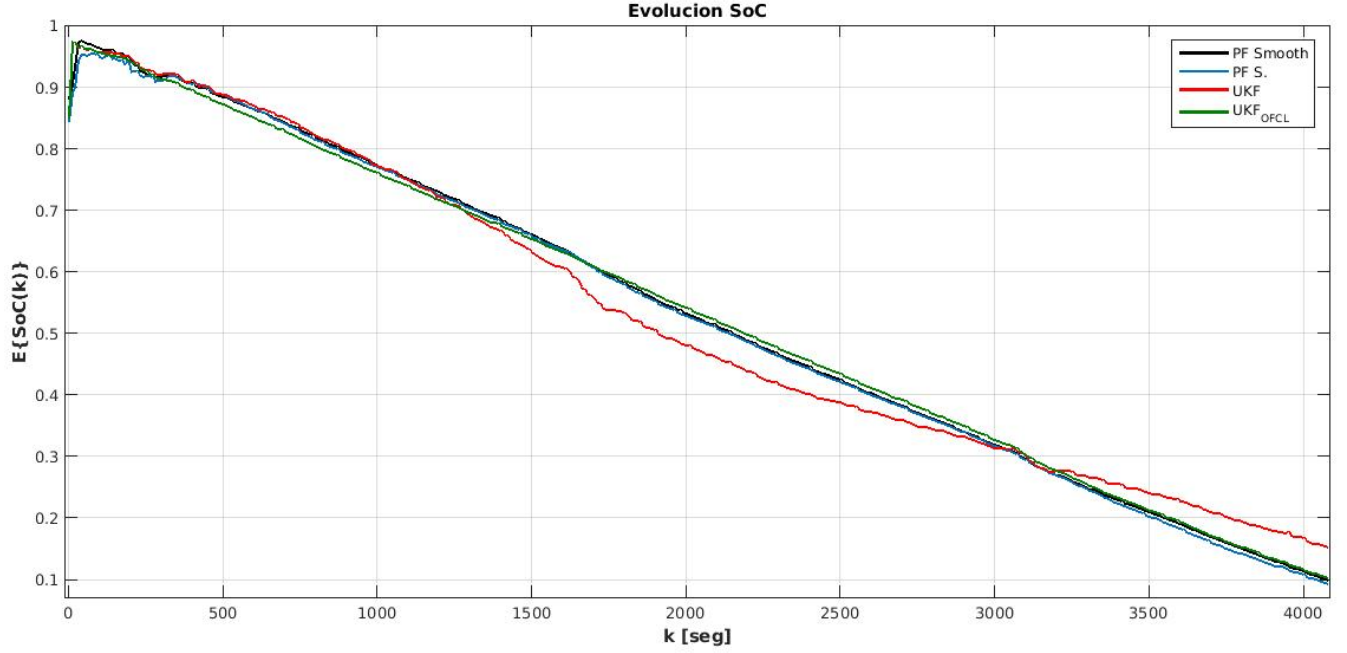


Figura 4.2: Evolución de la esperanza del SoC, según distintas aproximaciones.

## Reconstrucción de la densidad de probabilidad del estado

En la Figura 4.2 se observa la evolución de la Esperanza del SoC, pero si se escoge un instante cualquiera de la estimación, lo que cada algoritmo entrega es una representación de la distribución del estado mediante partículas y pesos. En la Figura 4.3 se pueden observar las partículas y pesos asociados de los 3 algoritmos a comparar, además se observa la reconstrucción en el intervalo  $[0, 1]$  -por medio de KDE, asumiendo un kernel Gaussiano y el ancho de banda óptimo para éste tipo de kernel, Sección 2.6- de las pdfs asociadas al estado del  $SoC$  en este instante.

De la Figura 4.3 se puede observar el problema de calcular directamente la divergencia de Kullback-Liebr entre estas distintas aproximaciones considerando la representación con pesos y partículas. Es por esta razón que, antes de calcular las medidas de información, se procede a reconstruir estas distribuciones y, de esta forma, es posible realizar una estimación no paramétrica de la entropía y la divergencia para este instante de tiempo y para cada algoritmo, como se explica en la Sección 3.3.

En particular, luego de obtener las aproximaciones para un instante de tiempo de cada pdf según KDE ( $\tilde{\pi}_k^*(x_k^{SoC})$  asociada al *Ground Truth*,  $\tilde{\pi}_k^{PF}(x_k^{SoC})$  para el filtro de partículas,  $\tilde{\pi}_k^{UKF}(x_k^{SoC})$  y  $\tilde{\pi}_k^{UKF_{ofcl}}(x_k^{SoC})$  para el algoritmo UKF y UKF con rutina OFCL, respectivamente), entonces se procede a escoger las muestras  $\{\tilde{X}_k^1, \dots, \tilde{X}_k^M\}$  que permiten evaluar la reconstrucción de estas pdf de forma que sea posible calcular no-paramétricamente  $H$  y  $D_{KL}$ .

Con respecto a la reconstrucción de la pdf para cada algoritmo, debido al alto costo computacional que requiere reconstruir la pdf del estado para cada instante de tiempo, para

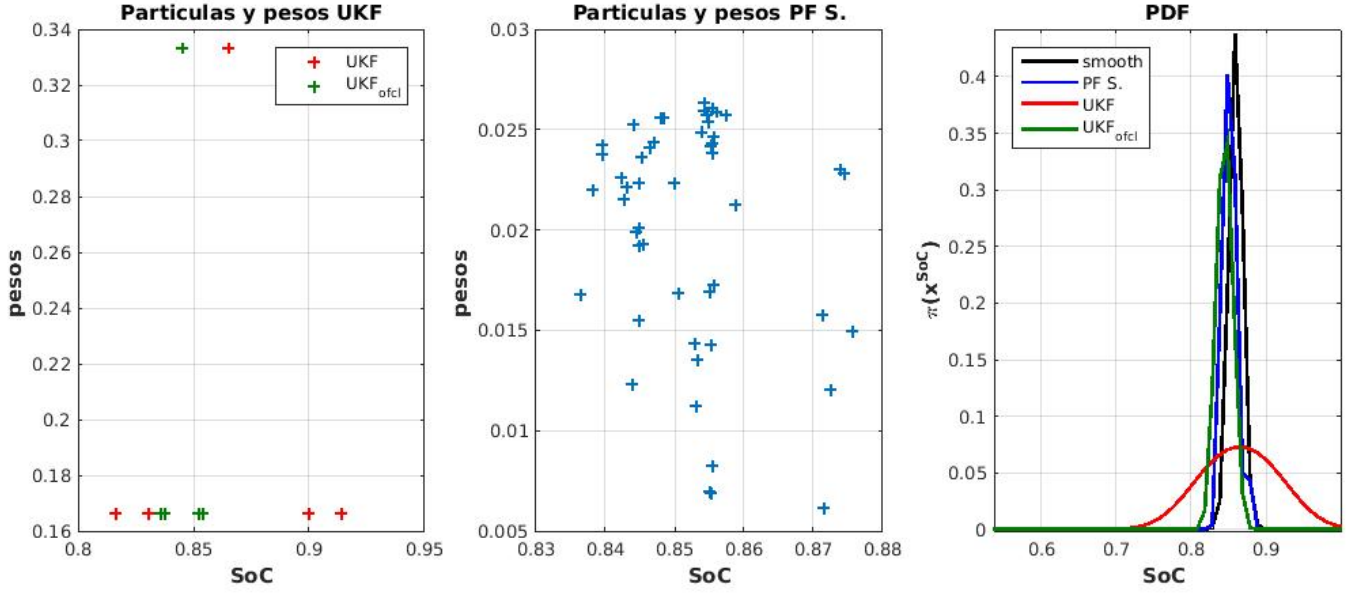


Figura 4.3: Representación mediante pesos y partículas de la distribución del  $SoC$  y respectiva reconstrucción de pdfs mediante KDE. instante  $k = 621$ .

cada algoritmo y además realiza esto 1060 por cada algoritmo simplificado, es que se decide realizar la reconstrucción cada 10 instantes; es decir,  $Ts = 10[\text{seg}]$  para la reconstrucción del  $SoC$  mediante KDE.

## Medidas de Información

En este momento, se dispone -para cada algoritmo- de la pdf aproximada en cada instante de tiempo  $k$ :

- $Ground\ Truth = \tilde{\pi}_k^*(x_k^{SoC})$
- Filtro de partículas =  $\tilde{\pi}_k^{PF}(x_k^{SoC})$
- UKF =  $\tilde{\pi}_k^{UKF}(x_k^{SoC})$
- UKF con OFCL =  $\tilde{\pi}_k^{UKF_{ofcl}}(x_k^{SoC})$

Para evaluar las pdfs en cada instante de tiempo se escogen 100 muestras equidistantes en el intervalo donde esta definido el  $SoC$   $[0, 1]$ . De esta forma  $M = 100$  y  $\{\tilde{X}_k^1, \dots, \tilde{X}_k^M\} = \{0, \frac{1}{99}, \frac{2}{99}, \dots, \frac{98}{99}, 1\}$ . Es decir, siempre se esta evaluando la concentración de información que entrega cada algoritmo en el intervalo  $[0, 1]$ , escogiendo las muestras adecuadas para poder evaluar cada pdf en las mismas posiciones de manera que sea posible estimar no parametricamente las medidas de información, como se muestra en las ecuaciones para la divergencia 4.1 y la entropía 4.2, 4.3:

$$D_{KL}(\pi_n(x_n) || \hat{\pi}_n(x_n)) \approx D_{KL}(\tilde{\pi}_n^*(x_n) || \tilde{\pi}_n(x_n)) = \sum_{i=1}^{100} \tilde{\pi}_n^* \left( \frac{i-1}{99} \right) \log \left( \frac{\tilde{\pi}_n^* \left( \frac{i-1}{99} \right)}{\tilde{\pi}_n \left( \frac{i-1}{99} \right)} \right) \quad (4.1)$$

$$H(\hat{\pi}_n(x_n)) \approx H(\tilde{\pi}_n(x_n)) = - \sum_{i=1}^{100} \tilde{\pi}_n\left(\frac{i-1}{99}\right) \log(\tilde{\pi}_n\left(\frac{i-1}{99}\right)) \quad (4.2)$$

$$H(\pi_n^*(x_n)) \approx H(\tilde{\pi}_n^*(x_n)) = - \sum_{i=1}^{100} \tilde{\pi}_n^*\left(\frac{i-1}{99}\right) \log(\tilde{\pi}_n^*\left(\frac{i-1}{99}\right)) \quad (4.3)$$

Considerando las aproximaciones descritas, en la Figura 4.4 se puede observar el valor estimado para la entropía en cada instante de tiempo, para cada algoritmo. Además, se observa como la aproximación de la entropía captura la evolución de la incertidumbre en el proceso, lo que puede interpretarse como *aprendizaje*. En este sentido, todos los algoritmos, excepto UKF, presentan una disminución de la incertidumbre en el tiempo, mientras que UKF muestra un incremento. Esto ocurre, pues la distribución inicial que se entrega presenta menor incertidumbre que la que utiliza UKF para representar el estado del sistema. En efecto, al incorporar la rutina de OFCL en UKF se observa una disminución en la incertidumbre, que también puede interpretarse como mayor aprendizaje, ya que la rutina de OFCL permite incorporar información sobre algunos parámetros del sistema que modela el proceso de descarga; en particular, las rutinas de OFCL implementadas modifican los ruidos de proceso. Por otra parte, en la Figura 4.5 se observa la estimación de la divergencia en cada instante de tiempo, es decir la evolución de la pérdida de información que presenta alguno de los algoritmos aproximados con respecto a la referencia, para la misma realización del proceso de descarga que se observa en la Figura 4.4.

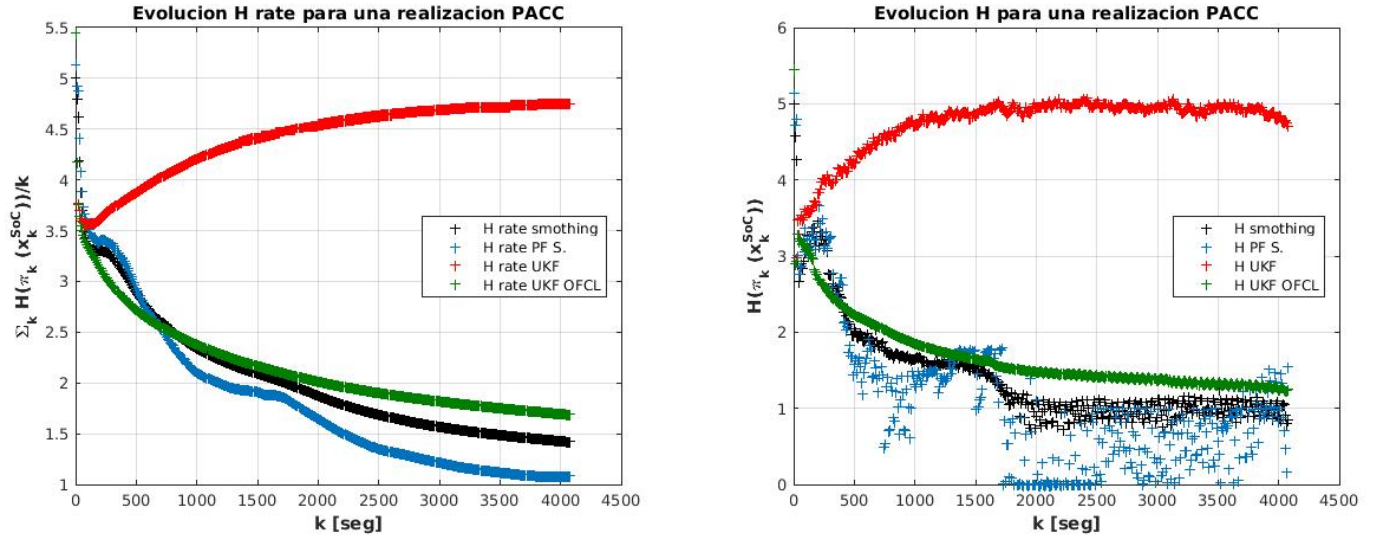


Figura 4.4: Evolución de la entropía para una realización del proceso de estimación, para cada algoritmo, por medio de estimación no paramétrica.  $T_s = 10[\text{seg}]$

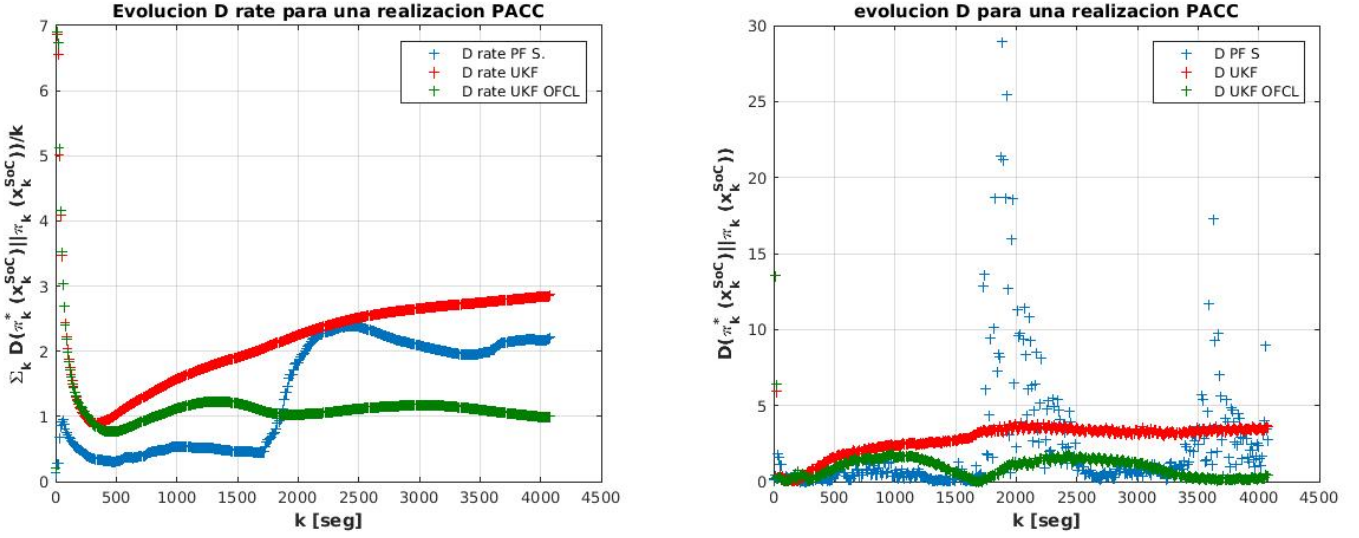


Figura 4.5: Evolución de la Divergencia para una realización del proceso de estimación, para cada algoritmo, por medio de estimación no paramétrica.  $T_s = 10[\text{seg}]$

Considerando una realización del proceso de estimación por cada algoritmo, y la aproximación de medidas de la información para cada instante de tiempo, se puede calcular una realización de las *loss functions* para cada algoritmo. Por ejemplo, la pérdida promedio de información (Definición 3.1) para cada algoritmo en la realización de la Figura 4.5 es:

- $\bar{D}_{KL}(\tilde{\pi}_{1:T}^*(x_{1:T}^{SoC}) || \tilde{\pi}_{1:T}^{PF}(x_{1:T}^{SoC})) \approx 2,1 \frac{[\text{bits}]}{[\text{seg}]}$
- $\bar{D}_{KL}(\tilde{\pi}_{1:T}^*(x_{1:T}^{SoC}) || \tilde{\pi}_{1:T}^{UKF}(x_{1:T}^{SoC})) \approx 2,9 \frac{[\text{bits}]}{[\text{seg}]}$
- $\bar{D}_{KL}(\tilde{\pi}_{1:T}^*(x_{1:T}^{SoC}) || \tilde{\pi}_{1:T}^{UKF_{ofcl}}(x_{1:T}^{SoC})) \approx 1 \frac{[\text{bits}]}{[\text{seg}]}$

Luego, para aproximar la máxima probabilidad de error es necesario calcular estas medidas  $N_C = 1060$  veces.

## Aproximación del máximo error teórico en probabilidad

Para contrastar el desempeño de los algoritmos de estimación en diferentes rangos de operación se los somete a cuatro pruebas con diferentes cantidades de datos. El objetivo es, dado un rango de operación, calcular las aproximaciones no paramétricas de las medidas de la información para cada instante y realización de los algoritmos simplificado y contrastarlos con el algoritmo de referencia. De esta manera, se obtiene por cada realización del algoritmo simplificado un valor para cada función de pérdida. Luego de obtener  $N_C$  realizaciones de la función de pérdida por cada algoritmo simplificado, se puede aproximar la probabilidad  $p_\gamma$ , por la probabilidad  $\hat{p}_N(\hat{\gamma})$ , y para que se cumpla que  $\hat{p}_N(\hat{\gamma}) = 1, \forall \gamma \geq \hat{\gamma}$  se escoge  $\hat{\gamma}$  como el máximo valor que obtiene el algoritmo simplificado para alguna realización de la función de pérdida tomando en cuenta todas las realizaciones.

Primeramente se realiza una estimación considerando el 90% de los datos; comenzando en

el momento que la batería esta completamente cargada (100 %) y hasta el momento en el que queda 10 % de carga aproximadamente. Esta prueba incluye un amplio rango de operación para la batería. Los resultados asociados al máximo valor que obtiene cada algoritmo, luego de las 1060 realizaciones para las funciones de pérdida, se puede observar en la Tabla 4.2.

Tabla 4.2: Máximo error empírico tomando el 90 % de los datos. 1060 realizaciones

Máximo error Empírico 90 %	FF-BS	PF	UKF	UKF OFCL
$\bar{H}(\tilde{\pi}_{1:T}) \frac{[bits]}{[seg]}$	1.42	1.82	4.74	1.69
$\bar{D}_{KL}(\tilde{\pi}_{1:T}^*    \tilde{\pi}_{1:T}) \frac{[bits]}{[seg]}$	0	44.87	2.86	0.99
$\Delta(\bar{H}(\tilde{\pi}_{1:T}^*, \tilde{\pi}_{1:T})) \frac{[bits]}{[seg]}$	0	0.82 ( $\bar{H}(\tilde{\pi}^*) > \bar{H}(\tilde{\pi})$ )	3.32 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )	0.26 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )

Considerando la no-linealidad del proceso y como cada aproximación logra capturarla, puede ocurrir que algún algoritmo presente un buen desempeño sólo hasta cierto instante de la estimación. Por esta razón se procede a realizar el mismo proceso de estimación, pero considerando menor cantidad de datos para cuantificar el desempeño. Se debe recordar que siempre se comienza estimando con la batería cargada completamente. En la Tabla 4.3 se observan los resultados al considerar 70 % de los datos, mientras que en la Tabla 4.4 se utiliza el 50 % y en la Tabla 4.5 el 30 %.

Tabla 4.3: Máximo error empírico tomando el 70 % de los datos. 1060 realizaciones

Máximo error Empírico 70 %	FF-BS	PF	UKF	UKF OFCL
$\bar{H}(\tilde{\pi}_{1:T}) \frac{[bits]}{[seg]}$	1.54	2.09	4.70	1.8
$\bar{D}_{KL}(\tilde{\pi}_{1:T}^*    \tilde{\pi}_{1:T}) \frac{[bits]}{[seg]}$	0	43.33	2.7	1.18
$\Delta(\bar{H}(\tilde{\pi}_{1:T}^*, \tilde{\pi}_{1:T})) \frac{[bits]}{[seg]}$	0	0.83 ( $\bar{H}(\tilde{\pi}^*) > \bar{H}(\tilde{\pi})$ )	3.15 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )	0.25 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )

Tabla 4.4: Máximo error empírico tomando el 50 % de los datos. 1060 realizaciones

Máximo error Empírico 50 %	FF-BS	PF	UKF	UKF OFCL
$\bar{H}(\tilde{\pi}_{1:T}) \frac{[bits]}{[seg]}$	1.77	2.27	4.59	1.96
$\bar{D}_{KL}(\tilde{\pi}_{1:T}^*    \tilde{\pi}_{1:T}) \frac{[bits]}{[seg]}$	0	44.07	2.42	1.08
$\Delta(\bar{H}(\tilde{\pi}_{1:T}^*, \tilde{\pi}_{1:T})) \frac{[bits]}{[seg]}$	0	0.93 ( $\bar{H}(\tilde{\pi}^*) > \bar{H}(\tilde{\pi})$ )	2.82 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )	0.18 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )

Tabla 4.5: Máximo error empírico tomando el 30 % de los datos. 1060 realizaciones

Máximo error Empírico 30 %	FF-BS	PF	UKF	UKF OFCL
$\bar{H}(\tilde{\pi}_{1:T}) \frac{[bits]}{[seg]}$	2.15	2.87	4.37	2.22
$\bar{D}_{KL}(\tilde{\pi}_{1:T}^*    \tilde{\pi}_{1:T}) \frac{[bits]}{[seg]}$	0	40.52	1.84	1.24
$\Delta(\bar{H}(\tilde{\pi}_{1:T}^*, \tilde{\pi}_{1:T})) \frac{[bits]}{[seg]}$	0	0.87 ( $\bar{H}(\tilde{\pi}^*) > \bar{H}(\tilde{\pi})$ )	2.21 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )	0.06 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )

Finalmente, con respecto a los resultados que se obtienen al incorporar medidas de información dentro del análisis PACC, cabe destacar:

- La función de *Pérdida promedio de información* ( $\bar{D}_{KL}(\tilde{\pi}_{1:T}^* || \tilde{\pi}_{1:T})$ ), es una medida que indica la cantidad de información promedio que se pierde al representar la distribución objetivo que entrega el algoritmo de referencia  $\tilde{\pi}_{1:T}^*(x_{1:T})$  por medio de la distribución aproximada  $\tilde{\pi}_{1:T}(x_{1:T})$ . En este sentido, la distribución  $\tilde{\pi}_{1:T}^{UKF_{ofcl}}(x_{1:T}^{SoC})$  es la que obtiene el menor valor estimado para el máximo error teórico. Según esta medida,  $\hat{\gamma}_{D_{KL}}^{UKF_{ofcl}} < \hat{\gamma}_{D_{KL}}^{UKF} < \hat{\gamma}_{D_{KL}}^{PF}$  se cumple en todos los rangos comparados. Lo que permite concluir que el algoritmo UKF con la rutina OFCL es el que obtiene la mayor exactitud computacional, considerando la cantidad de información promedio que se pierde. Además de presentar ventajas con respecto al algoritmo PF al considerar el tiempo de procesamiento, como se puede observar en el Anexo A.
- Otra conclusión al realizar el análisis y considerar la *Pérdida promedio de información* ( $\bar{D}_{KL}(\tilde{\pi}_{1:T}^* || \tilde{\pi}_{1:T})$ ) y la *Diferencia absoluta en incertidumbre promedio* ( $\Delta(\bar{H}(\pi_{1:T}, \hat{\pi}_{1:T}))$ ), es el efecto positivo que genera la rutina de OFCL en el algoritmo UKF. Esta rutina es capaz de reducir la incertidumbre asociada al estado del sistema, pero sin representar menos incertidumbre que el algoritmo de referencia, por lo que incluir la rutina tiene un efecto positivo en ambas funciones de pérdida y, por lo tanto, en la exactitud computacional del algoritmo.
- Por otro lado, si se considera solamente la función de pérdida *Diferencia absoluta en incertidumbre promedio* ( $\Delta(\bar{H}(\tilde{\pi}_{1:T}^*, \tilde{\pi}_{1:T}))$ ); que representa una medida en cuanto al manejo de la incertidumbre por los algoritmos. Entonces, en todos los rangos de operación se cumple que  $\hat{\gamma}_{\Delta(\bar{H})}^{UKF_{ofcl}} < \hat{\gamma}_{\Delta(\bar{H})}^{PF} < \hat{\gamma}_{\Delta(\bar{H})}^{UKF}$ . Esto puede llevar a conclusiones precipitadas sobre la exactitud computacional si se toma en cuenta sólo esta medida, pues se puede concluir que el manejo de la incertidumbre por el algoritmo PF es más adecuado -en cuanto a la *Diferencia absoluta en incertidumbre promedio*- que el del algoritmo UKF. Pero se debe tener en cuenta que para el algoritmo PF el máximo valor para la *Diferencia absoluta en incertidumbre promedio* se obtiene cuando la entropía promedio del algoritmo PF es menor que la del algoritmo Forward Filter-Backward Smoothing, mientras que esto nunca ocurre para UKF. El hecho de que el máximo valor de *Diferencia absoluta en incertidumbre promedio* para PF se alcance cuando su entropía promedio es menor que la del algoritmo de referencia, se debe considerar como una pérdida de información importante de la aproximación, pues se está representando el estado del sistema con menor incertidumbre que el algoritmo referencial. La medida *Pérdida pro-*

*medio de información* es capaz de incorporar esta pérdida en información penalizando más fuertemente una representación con menor entropía que la distribución objetivo.

- En relación al punto anterior, es interesante analizar la relación que existe entre el alto valor en la *Pérdida promedio de información* del filtro de partícula y como se relaciona con una representación incompleta de la incertidumbre del proceso. De hecho, para diferentes realizaciones de PF, en algunos instantes la entropía es 0, lo que implica que se concentra toda la información del proceso en un solo punto. Es por este motivo, que se sugiere la implementación de nuevas rutinas y estrategias que permitan atacar el problema de empobrecimiento -y no solo la degeneración- en la población de partículas, generando mayor diversidad en la representación del estado.

## 4.2. Metodología comparativa en Pronóstico

Para el escenario del pronóstico se tiene acceso a los mismos datos que para la estimación (FUDS, Figura 4.1). Además, se utilizan los mismos valores de los parámetros para el modelo en espacio de estado (Tabla 4.1).

### 4.2.1. Algoritmo de Referencia para Pronóstico

De manera análoga al escenario de comparación en estimación, para el pronóstico también es necesario definir un *Ground Truth* que será la referencia contra la cual se comparan los algoritmos simplificados. En este sentido, y como se explica en la Sección 2.3, las simulaciones de Monte Carlo ofrecen una solución al desafío del pronóstico a un alto costo computacional.

Para el caso de la estimación se dispone de un modelo dinámico de la forma:

$$x_{k+1} = f(x_k) + w_k \quad (4.4)$$

Donde  $x \in \mathbb{R}^2$ .  $x_2$  en la ecuación 2.46 ( $x^{SoC}$ ) es el estado que representa la evolución del *SoC*, mientras que  $x_1$  en la ecuación 2.45 (desde ahora se le llamará  $x^Z$ ) se interpreta como la impedancia interna del modelo, pero es un parámetro desconocido que sigue un proceso de aprendizaje según la evolución artificial [26]. Es importante destacar que los ruidos de proceso no tienen una interpretación física directa, y su objetivo principal es dotar al estado de diversidad y de esta forma permitir el aprendizaje. Por esta razón, no es directo incluir los mismos ruidos de proceso que se utilizan en estimación, en la rutina de pronóstico. Pues esto conlleva a incluir incertidumbre en la predicción que no es propia del sistema modelado. En particular, el incluir el ruido de  $x^Z$  en el pronóstico no es correcto, pues al ser ésta una variable asociada a un proceso de evolución artificial, que sólo aprende durante la estimación, entonces lo más sensato para realizar pronóstico es tomar el último valor “aprendido” de  $x^Z$  durante la estimación y utilizar este valor para continuar con el pronóstico, de otra manera se inyecta incertidumbre al sistema que no es propia de él, sino de la incapacidad de modelar correctamente la impedancia interna.

Tomando en cuenta lo expuesto anteriormente, y considerando que se tiene un *Ground Truth* para el *SoC* durante la estimación. Entonces, el objetivo es obtener un *Ground Truth*

para el pronóstico disponiendo de una secuencia de funciones de probabilidad -que se obtienen mediante el algoritmo de Forward Filter-Backward Smoothing- ordenada  $\{\pi_0^*(x_0), \pi_1^*(x_1), \dots, \pi_{T_p}^*(x_{T_p})\}$  (con  $T_p$  el tiempo donde se comienza a ejecutar el algoritmo de pronóstico y  $x_k = [x_k^Z, x_k^{SoC}]^T$ ). Para lograr esto se debe tener en cuenta lo siguiente:

- Considerando que es incorrecto incorporar el ruido de  $x^Z$  ( $w_1$  en la ecuación 2.45. Desde ahora  $w_Z$ ) en el pronóstico, y considerando también que se tiene una estimación para  $x^Z$  contemplando todos los datos (Forward Filter-Backward Smoothing). Entonces, para el pronóstico se escoge  $x^Z(k) = \mathbb{E}\{\pi_k^*(x_k^Z)\} \forall k \geq T_p$ .
- Ya que se quiere obtener una referencia para el pronóstico, la variable exógena del modelo ( $u$  en la ecuación 2.46), se escoge igual al valor de corriente real medida, es decir  $u(k) = y(k) \forall k \geq T_p$ . Se debe recordar que en el esquema de pronóstico, aún cuando se entrega la variable exógena, no se modifican los pesos de las partículas; Algoritmo 6. El hecho de entregar la variable exógena igual al valor real de corriente es con la intención de generar una referencia adecuada, pero es incorrecto interpretar el valor de la variable exógena en el pronóstico como una observación.
- Finalmente, tomando en consideración que el ruido de proceso para el pronóstico del  $SoC$  no es el mismo que se utiliza en la estimación, se propone la siguiente metodología para determinarlo. Tomando en cuenta que se tiene una colección de distribuciones para el estado de carga con anterioridad al pronóstico  $\{\pi_0^*(x_0^{SoC}), \pi_1^*(x_1^{SoC}), \dots, \pi_{T_p}^*(x_{T_p}^{SoC})\}$ , entonces se construye un modelo dinámico de la forma:

$$\pi_{k+1}^*(x_{k+1}^{SoC}) = f(\pi_k^*(x_k^{SoC})) + w_k \quad (4.5)$$

Donde  $f$  se mantiene de la ecuación 2.46 y solo se modifica el ruido. Entonces, si  $x_k^{SoC}$  y  $w_k$  son variables aleatorias independientes, la secuencia debe respetar la ecuación 4.6

$$Var\{\pi_{k+1}^*(x_{k+1}^{SoC})\} = Var\{f(\pi_k^*(x_k^{SoC}))\} + Var\{w_k\} \quad (4.6)$$

Por lo tanto, si el ruido  $w_k$  es i.i.d., la varianza debe cumplir con la ecuación 4.7:

$$\begin{aligned} Var\{\pi_1^*(x_1^{SoC})\} &= Var\{f(\pi_0^*(x_0^{SoC}))\} + Var\{w_k\} \\ Var\{\pi_2^*(x_2^{SoC})\} &= Var\{f(\pi_1^*(x_1^{SoC}))\} + Var\{w_k\} \\ &\vdots \\ Var\{\pi_{T_p}^*(x_{T_p}^{SoC})\} &= Var\{f(\pi_{T_p-1}^*(x_{T_p-1}^{SoC}))\} + Var\{w_k\} \end{aligned} \quad (4.7)$$

Lo anterior justifica la idea de definir la varianza del ruido  $w_k$  según la ecuación 4.8:

$$Var\{w_k\} = \frac{1}{M} \sum_{i=T_p-M-1}^{T_p} [Var\{\pi_i^*(x_i^{SoC})\} - Var\{f(\pi_{i-1}^*(x_{i-1}^{SoC}))\}] \quad (4.8)$$

Donde  $M$  son la cantidad de instantes que se escogen -anteriores al pronóstico- para aproximar la varianza de  $w_k$ , y para este trabajo se utilizan los 100 instantes anteriores, así  $M = 100$ .

Finalmente, tomando en cuenta todas las consideraciones mencionadas, se aplica el Algoritmo 6, utilizando 50000 partículas.



## 4.2.2. Algoritmos a Comparar

Con respecto a los algoritmos simplificados para el pronóstico, se modifican versiones ya existentes de algoritmos de pronóstico para el estado de carga de baterías de Ion-Litio, que se han desarrollado por estudiantes de Ingeniería Eléctrica de la Universidad de Chile, bajo la guía del profesor Marcos Orchard. Todos los algoritmos consideran la misma función de transición de estado y de observación asociadas a la modelación en espacio de estado.

Con el fin de comparar diferentes modificaciones en cada uno de los métodos, se toman  $N_C = 1060$  realizaciones de cada algoritmo, por cada perfil de corriente. El objetivo es medir como afecta la calidad de los resultados al incluir distintos perfiles de corriente para generar la predicción del estado en el tiempo. Específicamente, se utilizan tres perfiles de corriente:

- **Corriente Real:** Considerando que se tiene acceso a los datos reales de corriente, entonces se puede entregar el valor real de la corriente a cada algoritmo y realizar el pronóstico. El objetivo de este perfil es incluir la información real del proceso, para luego contrastar ésta aproximación con perfiles de corriente que intentan representar la evolución de ésta sin tener acceso a su valor real.
- **Corriente Promedio:** Una de las aproximaciones más simples que se pueden generar para realizar el pronóstico es tomar la corriente hasta el instante del pronóstico, promediarla, y utilizar este valor para construir la predicción.  
Es decir  $u(k) = \frac{1}{T_p} \sum_{j=1}^{T_p} I(j), \forall k \geq T_p$
- **Corriente Cadena de Markov:** Propuesto originalmente en [23], el objetivo es construir un perfil de corriente de descarga por medio de una realización de una cadena de Markov homogénea de dos estados, los cuales son representados por transiciones aleatorias entre niveles altos y bajos de corriente. De esta manera, se puede medir el efecto de generar un perfil de corriente que capture características estadísticas de la evolución de la corriente hasta antes del pronóstico, y luego utilizar ésta caracterización para determinar los niveles y las probabilidades de transición asociadas a la cadena de Markov.

Finalmente, ya que el pronóstico intenta predecir el momento en que fallará el sistema en el futuro, entonces para cada realización de la rutina de pronóstico se puede construir la probabilidad de masa asociada al EoL (del inglés *End of Life*), ésta pmf caracteriza la probabilidad de que el estado de carga de la batería este bajo cierto umbral. Para calcular la pmf se considera la deficiencia de EoL según [2]. Además, la *hazard zone*, para este caso, se define:

$$p(\text{failure}|x_k) = \begin{cases} 1 & \text{Si } SoC(k) \leq 20\% \\ 0 & \sim \end{cases} \quad (4.9)$$

## Filtro de Particulas Regularizado

Implementado originalmente en [13], el esquema de pronóstico por medio de PF regularizado esta resumido en el Algoritmo 7. En esta implementación se utilizan 50 partículas. Además, para medir el desempeño de este esquema se considera la realización del PF tomando los tres perfiles de corriente explicados anteriormente.

## Filtro de Particulas con propagación de incertidumbre via sigma-points

El algoritmo basado en filtro de partículas con propagación de incertidumbre via sigma-points, originalmente, fue implementado para el caso de baterías en [2]. Esta aproximación sigue los pasos descritos por el Algoritmo 8 y, al igual que con el PF regularizado, se somete a este algoritmos a los tres perfiles de corriente mencionados, para evaluar el efecto que tienen en la calidad de los resultados.

Se utilizan  $N_k = 90$  partículas,  $M_k = 10$  componentes de la GMM y  $\frac{N_k}{M_k} = 9$  sigma-points.

## Condiciones Iniciales

Para cada realización del pronóstico todos los algoritmos comparten la misma condición inicial. Ésta se construye definiendo una distribución normal utilizando el *Ground Truth* que se obtiene en la estimación, en el instante donde comienza la rutina de pronóstico. Entonces, si comienza el pronóstico en el instante  $k = T_p$ , la condición inicial es:

$$\pi_{T_p}(x_{T_p}) \sim \mathcal{N}(\mathbb{E}\{\pi_{T_p}^*(x_{T_p}^*)\}, \text{Var}\{\pi_{T_p}^*(x_{T_p}^*)\})$$

Donde  $\pi_{T_p}^*(x_{T_p}^*)$  se tiene de la etapa de estimación, al generar el resultado por *Forward Filter-Backward Smoothing*.

### 4.2.3. Resultados de la comparación

De la misma forma que para la estimación, al momento de aproximar el máximo error teórico en las rutinas de pronóstico, se deben obtener  $N_C$  realizaciones del pronóstico por cada algoritmo simplificado. De esta manera, si se escoge  $N_C = 1060$ , se puede aproximar la probabilidad de máximo error teórico con precisión  $\varepsilon = 0,05$  y confianza  $(1 - \delta) = 0,99$ .

Si se toma la distribución *a posteriori* que entrega el algoritmo de referencia para la estimación (FF-BS) en el instante  $k = T_p$  y se utiliza como condición inicial para el pronóstico, entonces se pueden generar trayectorias de la evolución del estado del sistema. En particular, en la Figura 4.6 se observa la evolución de la esperanza del *SoC* para el algoritmo PF, considerando los tres perfiles de corriente descritos y comenzando la predicción desde el 70% de los datos en adelante. De la misma forma, en la Figura 4.7 se observa la realización del mismo experimento, pero ejecutando el algoritmo PFSG para generar la predicción.

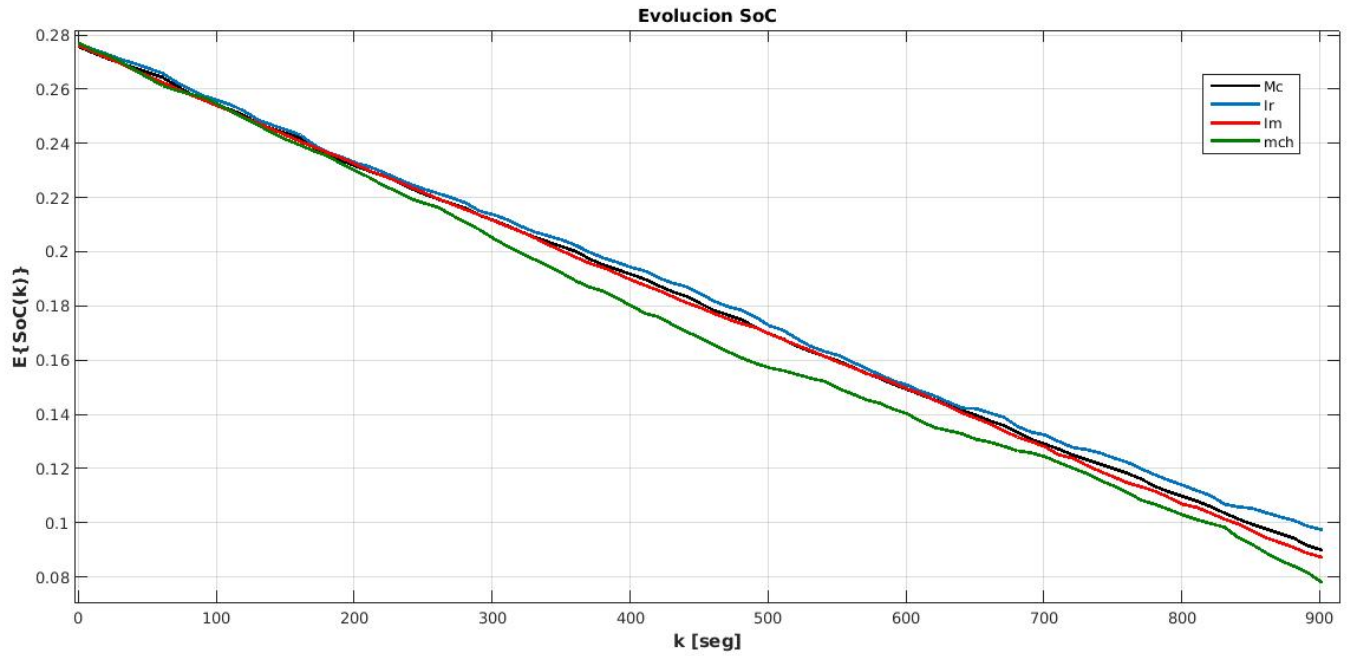


Figura 4.6: Evolucion de la Esperanza del  $SoC$  para el algoritmo PF, considerando tres perfiles de corriente distintos:  $Ir$  = Corriente Real ,  $Im$  =Corriente promedio y  $mch$  = corriente Cadena de Markov.

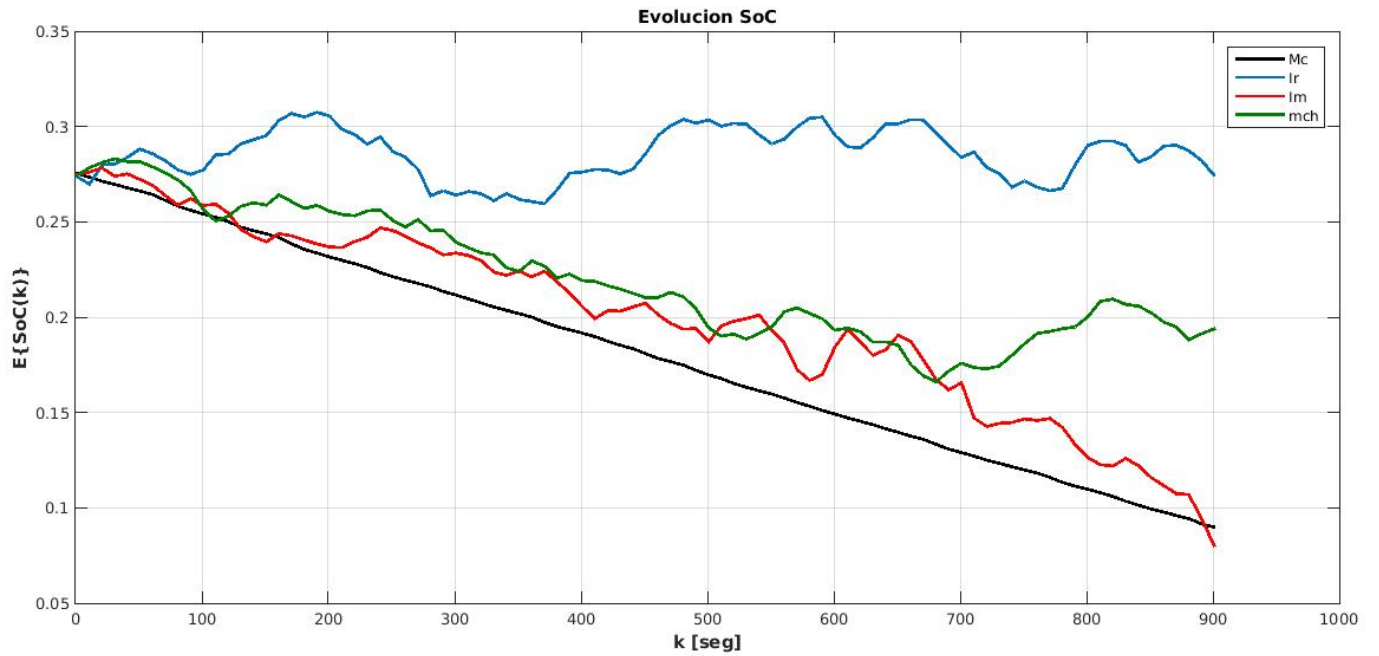


Figura 4.7: Evolucion de la Esperanza del  $SoC$  para el algoritmo PFSG, considerando tres perfiles de corriente distintos:  $Ir$  = Corriente Real ,  $Im$  =Corriente promedio y  $mch$  = corriente Cadena de Markov.

## Reconstrucción de la densidad de probabilidad del estado

Entonces, si se escoge una realización del pronóstico por cada algoritmo, y se analiza un instante particular, se puede reconstruir la distribución de probabilidades asociada al estado, por medio de KDE (para este caso también se asume kernel Gaussiano y el ancho de banda óptimo, como se explica en la Sección 2.6). Además, por cada realización del pronóstico se obtiene una probabilidad de masa asociada al EoL, para el caso de filtro de partículas regularizado y filtro de partículas con propagación de incertidumbre via sigma-points, respectivamente como se ilustra en las Figuras 4.8 y 4.9.

## Medidas de Información

Tomando la descripción por partículas y pesos que genera cada algoritmo para la predicción de la evolución del sistema, se construyen las pdfs en cada instante de tiempo  $k \geq T_p$ :

- *Ground Truth* =  $\tilde{\pi}_k^*(x_k^{SoC})$
- Filtro de partículas regularizado =  $\tilde{\pi}_k^{PF}(x_k^{SoC})$
- Filtro de partículas con propagación de incertidumbre via sigma-points =  $\tilde{\pi}_k^{PFSG}(x_k^{SoC})$

Luego, para estimar las medidas de información en el pronóstico, se utilizan las mismas ecuaciones descritas para el caso de la estimación; divergencia 4.1 y entropías 4.2, 4.3.

De esta manera, se aproxima la entropía como muestra la Figura 4.10 para el filtro de partículas regularizado y la Figura 4.11 para el filtro de partículas con propagación de incertidumbre via sigma-points. Vale destacar que, a diferencia del caso de estimación, al momento de predecir la evolución del estado de carga se observa un constante aumento en la entropía; al no tener acceso a datos, no es posible modificar la distribución *a priori* del estado y, por tanto, incorporar nueva información del proceso, lo que conlleva un aumento de la incertidumbre a medida que el tiempo avanza. Con respecto al efecto de la condición inicial -a diferencia de la estimación- en la rutina de pronóstico siempre se observará un aumento de la entropía independiente de la condición inicial, siempre que la distribución en  $k = T_p$  no describa la máxima cantidad de incertidumbre posible asociada al estado del sistema. Por otra parte, con respecto a la divergencia, en la Figura 4.12 se observa la evolución de esta medida para el algoritmo de filtro de partículas regularizado -considerando la misma realización que para la Figura 4.10-, mientras que la Figura 4.13 muestra la misma medida para la rutina PFSG, utilizando la misma realización del pronóstico que el que se observa en la estimación de la entropía de la Figura 4.11.

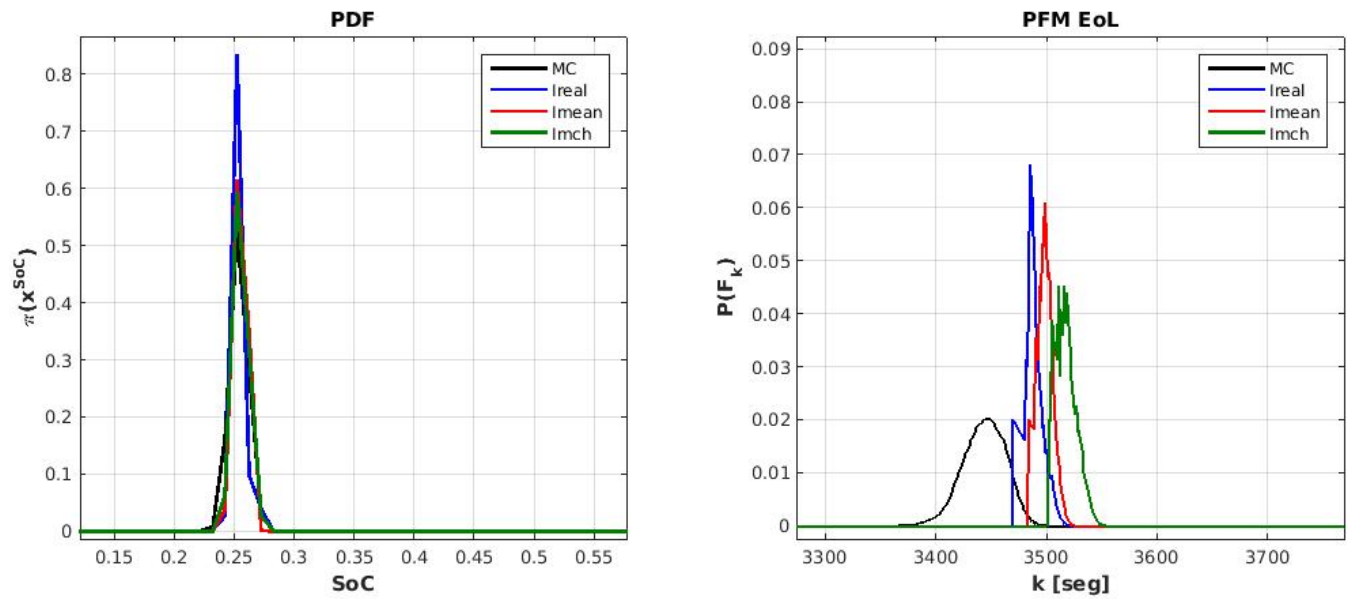


Figura 4.8: Algoritmo PF. Izquierda: Reconstrucción de pdfs mediante KDE, para instante  $k = 3275$ . Derecha: EoL (20%) según cada perfil de corriente.

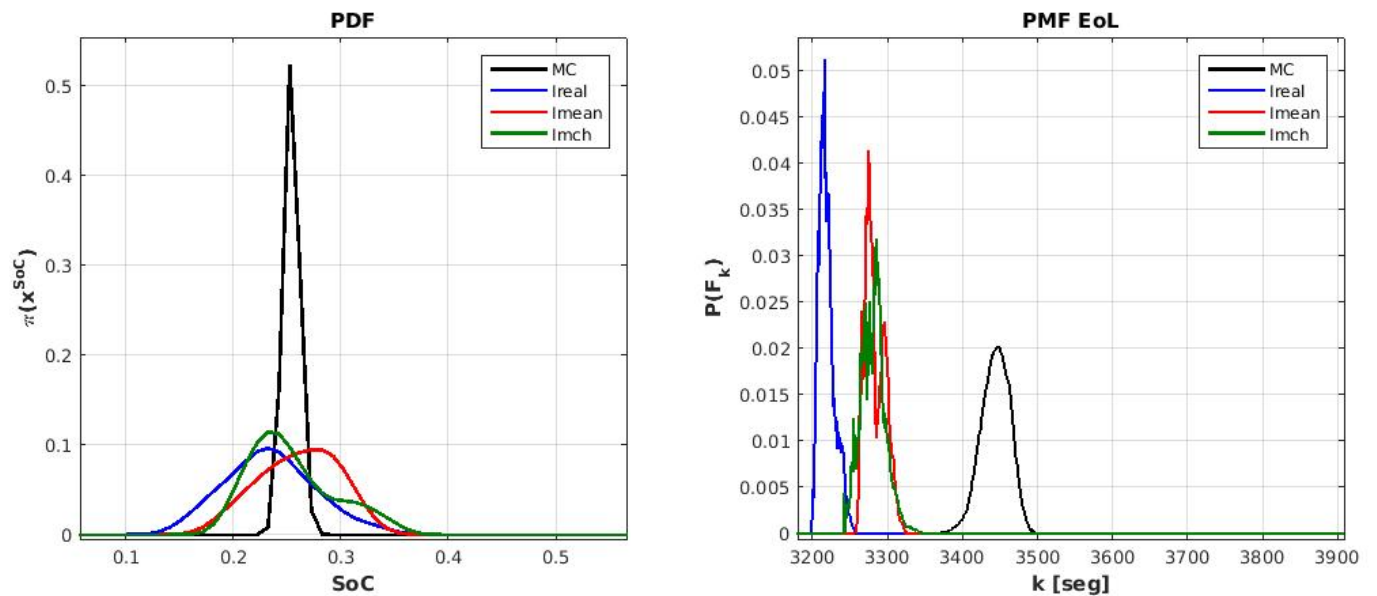


Figura 4.9: Algoritmo PFSG. Izquierda: Reconstrucción de pdfs mediante KDE, para instante  $k = 3275$ . Derecha: EoL (20%) según cada perfil de corriente.

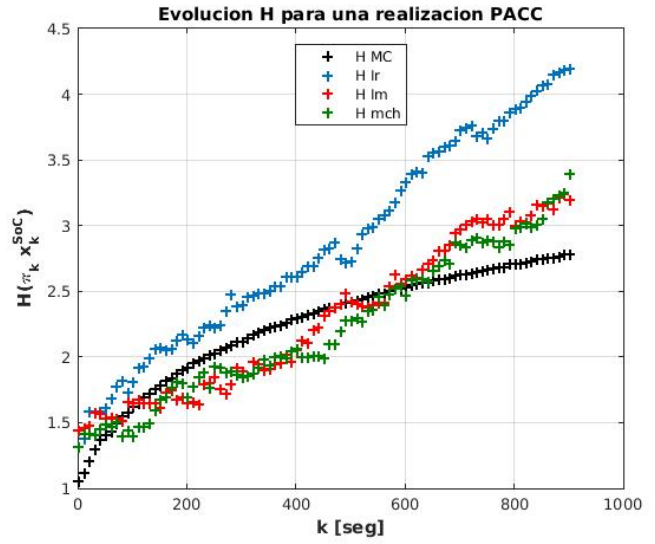
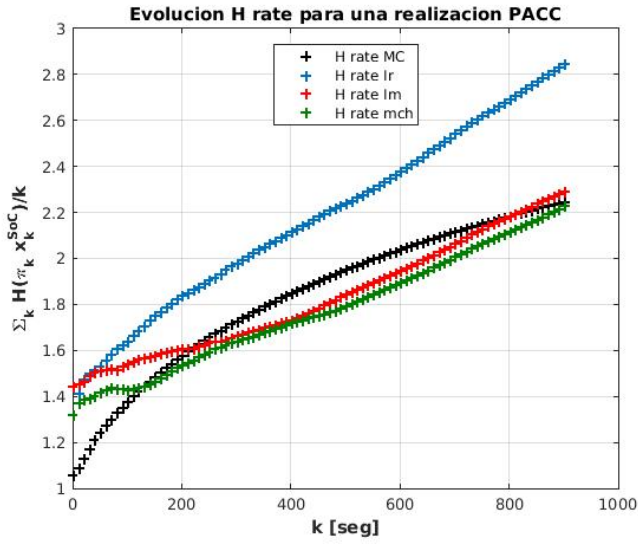


Figura 4.10: Evolucion de la Entropia, según cada perfil de corriente para el algoritmo PF; una realización del pronóstico.  $T_s = 10[\text{seg}]$

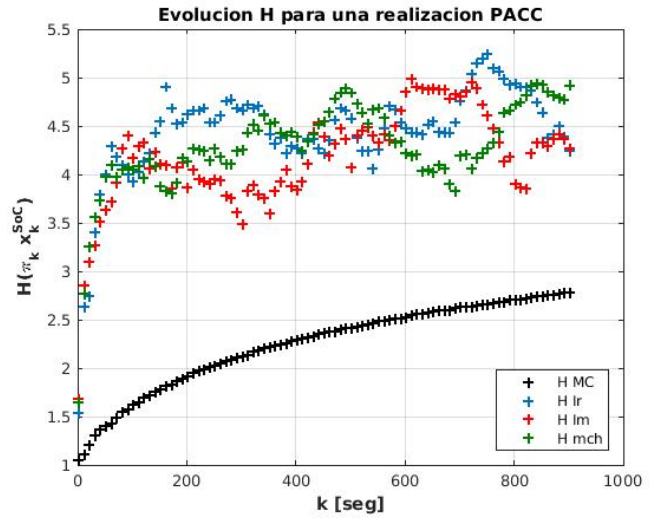
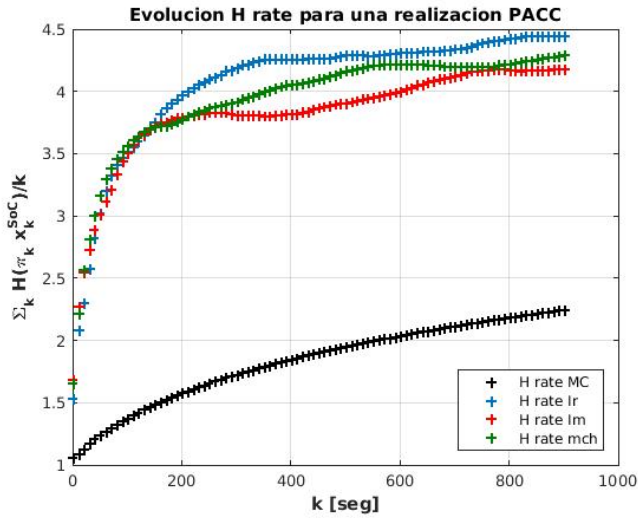


Figura 4.11: Evolucion de la Entropia, según cada perfil de corriente para el algoritmo PFSG; una realización del pronóstico.  $T_s = 10[\text{seg}]$

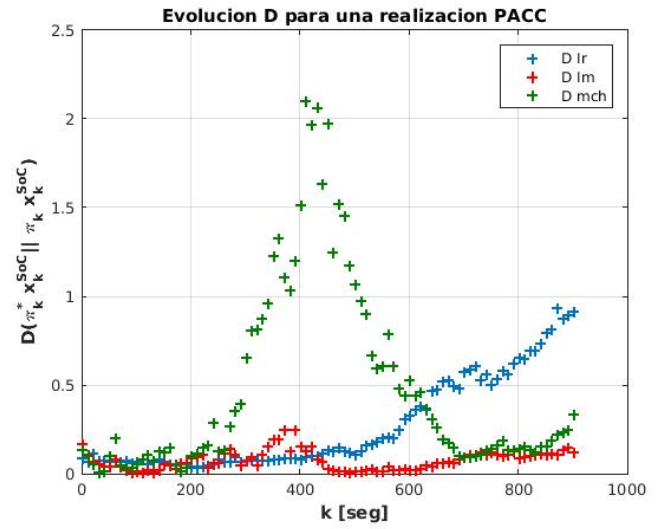
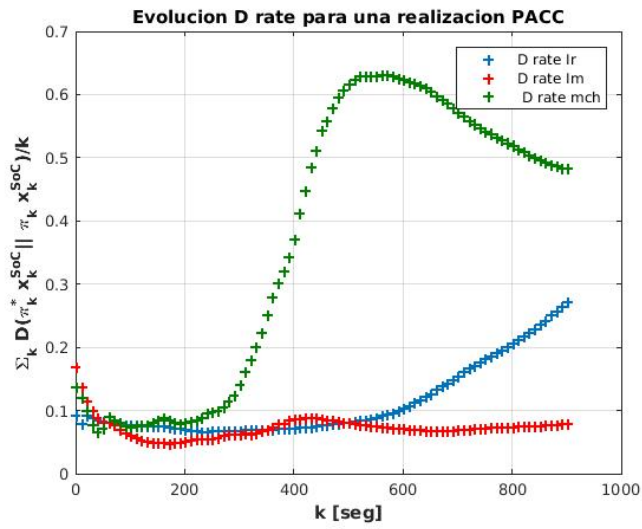


Figura 4.12: Evolución de la Divergencia según cada perfil de corriente para el algoritmo PF; una realización del pronóstico.  $T_s = 10[seg]$

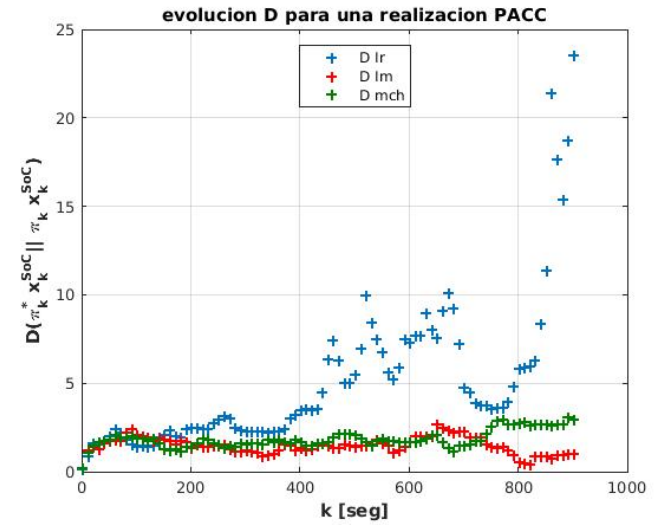
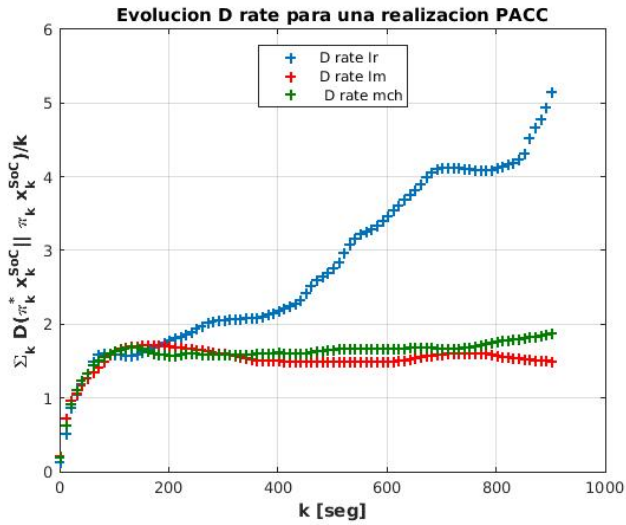


Figura 4.13: Evolución de la Divergencia según cada perfil de corriente para el algoritmo PFSG; una realización del pronóstico.  $T_s = 10[seg]$

## Aproximación del máximo error teórico en probabilidad

Entonces, para contrastar el desempeño de los algoritmos, se generan las  $N_c = 1060$  realizaciones de los diferentes esquemas simplificados, según los distintos perfiles de corriente. La Tabla 4.6 muestra los resultados de la comparación, cuando se implementan las rutinas de pronóstico desde el 70 % de los datos en adelante.

Para contrastar el desempeño de los algoritmos de pronóstico bajo distintos rangos de operación se utilizan tres pruebas con diferentes cantidades de datos. El objetivo es, análogamente al escenario de la estimación, dado un rango de operación, calcular las aproximaciones no paramétricas de las medidas de información para cada instante y realización de los algoritmos simplificado y contrastarlos con el algoritmo de referencia. Los resultados asociados a comenzar el pronóstico desde el 50 % de los datos en adelante se muestran en la Tabla 4.7, mientras que el mismo experimento, pero comenzando desde el 30 % en adelante se observa en la Tabla 4.8.

Tabla 4.6: Máximo error empírico comenzando el pronóstico desde el 70 % de los datos. 1060 realizaciones

Máximo error Empírico 70 %	G.T	$I_{real}$	$I_{promedio}$	$I_{M.Ch}$
$\bar{H}(\tilde{\pi}_{1:T}^{PF}) \frac{[bits]}{[seg]}$	2.24	2.98	3.13	2.9
$\bar{H}(\tilde{\pi}_{1:T}^{PFSG}) \frac{[bits]}{[seg]}$	2.24	5.71	5.53	5.71
$\bar{D}_{KL}(\tilde{\pi}_{1:T}^*    \tilde{\pi}_{1:T}^{PF}) \frac{[bits]}{[seg]}$	0	3.19	3.67	13.49
$\bar{D}_{KL}(\tilde{\pi}_{1:T}^*    \tilde{\pi}_{1:T}^{PFSG}) \frac{[bits]}{[seg]}$	0	21.91	11.95	24.7
$\Delta(\bar{H}(\tilde{\pi}_{1:T}^*, \tilde{\pi}_{1:T}^{PF})) \frac{[bits]}{[seg]}$	0	1.08 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )	1.15 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )	1.34 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )
$\Delta(\bar{H}(\tilde{\pi}_{1:T}^*, \tilde{\pi}_{1:T}^{PFSG})) \frac{[bits]}{[seg]}$	0	3.47 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )	3.28 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )	3.46 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )
$D_{KL}(ToF^*    T\hat{o}F^{PF}) [bits]$	0	33.84	32.59	36.74
$D_{KL}(ToF^*    T\hat{o}F^{PFSG}) [bits]$	0	45.7	45.7	45.7



Tabla 4.7: Máximo error empírico comenzando el pronóstico desde el 50 % de los datos. 1060 realizaciones

Máximo error Empírico 50 %	G.T	$I_{real}$	$I_{promedio}$	$I_{M.Ch}$
$\bar{H}(\tilde{\pi}_{1:T}^{PF}) \frac{[bits]}{[seg]}$	2.51	3.92	3.91	3.81
$\bar{H}(\tilde{\pi}_{1:T}^{PFSG}) \frac{[bits]}{[seg]}$	2.51	5.92	6.07	6.046
$\bar{D}_{KL}(\tilde{\pi}_{1:T}^*    \tilde{\pi}_{1:T}^{PF}) \frac{[bits]}{[seg]}$	0	2.26	2.59	3.23
$\bar{D}_{KL}(\tilde{\pi}_{1:T}^*    \tilde{\pi}_{1:T}^{PFSG}) \frac{[bits]}{[seg]}$	0	35.95	37.26	31.8
$\Delta(\bar{H}(\tilde{\pi}_{1:T}^*, \tilde{\pi}_{1:T}^{PF})) \frac{[bits]}{[seg]}$	0	1.41 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )	1.4 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )	1.3 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )
$\Delta(\bar{H}(\tilde{\pi}_{1:T}^*, \tilde{\pi}_{1:T}^{PFSG})) \frac{[bits]}{[seg]}$	0	3.4 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )	3.55 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )	3.53 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )
$D_{KL}(ToF^*    T\hat{o}F^{PF}) [bits]$	0	2.8	2.63	3.37
$D_{KL}(ToF^*    T\hat{o}F^{PFSG}) [bits]$	0	44.95	44.95	44.95

Tabla 4.8: Máximo error empírico comenzando el pronóstico desde el 30 % de los datos. 1060 realizaciones

Máximo error Empírico 30 %	G.T	$I_{real}$	$I_{promedio}$	$I_{M.Ch}$
$\bar{H}(\tilde{\pi}_{1:T}^{PF}) \frac{[bits]}{[seg]}$	3.58	5.03	5.06	5.05
$\bar{H}(\tilde{\pi}_{1:T}^{PFSG}) \frac{[bits]}{[seg]}$	3.58	6.13	5.62	6.02
$\bar{D}_{KL}(\tilde{\pi}_{1:T}^*    \tilde{\pi}_{1:T}^{PF}) \frac{[bits]}{[seg]}$	0	2.19	2.76	2.06
$\bar{D}_{KL}(\tilde{\pi}_{1:T}^*    \tilde{\pi}_{1:T}^{PFSG}) \frac{[bits]}{[seg]}$	0	36.19	28.07	35.86
$\Delta(\bar{H}(\tilde{\pi}_{1:T}^*, \tilde{\pi}_{1:T}^{PF})) \frac{[bits]}{[seg]}$	0	1.45 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )	1.47 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )	1.43 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )
$\Delta(\bar{H}(\tilde{\pi}_{1:T}^*, \tilde{\pi}_{1:T}^{PFSG})) \frac{[bits]}{[seg]}$	0	2.55 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )	2.03 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )	2.43 ( $\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi})$ )
$D_{KL}(ToF^*    T\hat{o}F^{PF}) [bits]$	0	6.12	5.99	6.2
$D_{KL}(ToF^*    T\hat{o}F^{PFSG}) [bits]$	0	44.15	44.15	44.15

De los resultados que se obtienen aplicando la metodología propuesta al escenario de pronóstico, se desprende lo siguiente:

- A diferencia del caso asociado a estimación, para el pronóstico todos los algoritmos simplificados obtienen el máximo de *Diferencia absoluta en incertidumbre* con  $(\bar{H}(\tilde{\pi}^*) < \bar{H}(\tilde{\pi}))$ . Es decir, los algoritmos simplificados representan la predicción del estado con mayor incertidumbre que la referencia. Esto es consistente considerando que los algoritmos simplificados son una aproximación más simple de la evolución del estado del sistema que la que entrega el algoritmo de referencia.
- En general, el filtro de partículas regularizado obtiene un máximo error empírico menor que el que obtiene el filtro de partículas con propagación de incertidumbre via sigma-points. Considerando las tres medidas propuestas; *Diferencia absoluta en incertidumbre*, *Pérdida promedio de información* y *Pérdida de información en la pmf del EoL*. Entonces, como  $\hat{\gamma}^{PF} < \hat{\gamma}^{PFSG} \iff$  la exactitud computacional de PF es mayor que la de PFSG, se concluye que el algoritmo PF es más exacto computacionalmente que el algoritmo PFSG, tomando las medidas de información propuesta para realizar la comparación.
- Tomando en cuenta la forma en la que están implementados los algoritmos hasta este momento, introducir en el pronóstico la corriente promedio como variable exógena es una simplificación que implica un menor costo computacional sin comprometer el desempeño del algoritmo (Anexo A). Sin embargo, el valor de la corriente y la impedancia interna están relacionados. Es decir, distintos perfiles de corriente, inducen diferentes valores para la impedancia. Esta relación aún no está bien definida, como se puede observar en las ecuaciones de transición de estado (particularmente, ecuación 2.45), y por esto es que se utiliza un valor constante para la impedancia interna en el pronóstico y así no incorporar incertidumbre que no es propia del proceso. En este sentido, se sugiere dedicar esfuerzo a definir apropiadamente la relación que existe entre la impedancia interna y la corriente, y así poder introducir valores de impedancia consistentes con el perfil de la variable exógena tanto para el pronóstico, como para la estimación, lo que significará incorporar mayor información sobre la representación del proceso y, por tanto, una caracterización adecuada de la evolución de los parámetros del modelo, que se concreta en un resultado que incluye un significado físico del sistema que aún no está definido apropiadamente.
- Con respecto al desempeño que obtiene el Algoritmo PFSG, cabe mencionar dos aspectos interesantes a tener en cuenta para mejorar su desempeño. A diferencia del Algoritmo PF regularizado, PFSG asume el ruido de proceso al momento de realizar el pronóstico, sin que este ruido tenga un significado físico directo asociado al sistema. En este sentido, dedicar esfuerzos a la correcta modelación del ruido de proceso a utilizar en el pronóstico posiblemente tenga un impacto positivo en el desempeño del Algoritmo PFSG. En segundo lugar, el objetivo de PFSG es caracterizar apropiadamente las colas de las pdfs asociadas al vector de estado. En consecuencia, efectivamente su estructura le permite representar de mejor manera las colas de las pdfs, pero esto no significa necesariamente una adecuada representación de la distribución de probabilidades del estado de sistema, ni de la distribución asociada al EoL. Por lo tanto, si las medidas a utilizar para contrastar el desempeño de los algoritmos consideran la totalidad de la descripción estadística que entregan estas rutinas, entonces este algoritmo puede evaluarse

con un desempeño pobre. Mientras que si se utilizan otras medidas para cuantificar su desempeño, por ejemplo medidas que centren su atención en las colas de las pdfs (como la medida  $JITP_{\alpha\%}$ ), entonces pueden cambiar radicalmente las conclusiones con respecto al desempeño de PFSG.

### 4.3. Comentarios sobre la metodología

Es importante mencionar que el marco comparativo es bastante flexible en relación a las funciones de pérdida que se pueden incorporar. En este sentido, diferentes funciones de pérdida pueden ser más apropiadas para medir ciertas cualidades interesantes de los algoritmos, y de esta manera se pueden determinar funciones de pérdida específicas para medir el efecto de subrutinas de los algoritmos. Por ejemplo, al medir la entropía se pueden tomar decisiones en cuanto al efecto que tiene en el desempeño de los algoritmos distintas subrutinas de resampling y diferentes condiciones para ejecutarlas, o el número de partículas necesario para incorporar la incertidumbre mínima. En este sentido, también es posible realizar pruebas donde cada vez se incluyan condiciones iniciales distintas en los algoritmos simplificados, de esta manera se puede cuantificar la sensibilidad que tienen con respecto a una inicialización incorrecta en el estado del sistema.

La manera en la que se pueden aproximar las medidas de información no es única. En este sentido, se sugiere el desarrollo, estudio e implementación de nuevas técnicas que permitan estimar medidas de información en el enfoque de la inferencia Bayesiana. Este punto es crucial, pues hasta el momento las medidas que son utilizadas generalmente para medir el desempeño de este tipo de algoritmos (MSE, por ejemplo), no incorpora una adecuada descripción de la caracterización estadística que se asume para la evolución del sistema.

En relación al párrafo anterior, parece extramadamente útil el desarrollo e incorporación de una aproximación de la información mutua como función de pérdida para cuantificar la pérdida de desempeño en los algoritmos. Puede ser interesante explorar una manera de modificar la *Adjusted Mutual Information*[24], para que sea aplicable al momento de comparar diferentes aproximaciones a la evolución del sistema en estudio.

Si bien el desafío original de la metodología propuesta es comparar algoritmos de estimación que aproximan la solución de SMC, el método se puede extender para generar comparaciones de algoritmos que generen la estimación de otra manera. En particular, otra forma en la que puede ser útil el marco comparativo es al momento de determinar las funciones que definen el modelo en espacio de estado asociado a SMC. Por ejemplo, si se obtiene una aproximación de la distribución *a posteriori* del estado de un sistema, por alguna metodología distinta del enfoque secuencial de Monte Carlo, y se utiliza este resultado como la distribución objetivo, entonces se puede definir un problema de optimización, donde el objetivo es minimizar el máximo error teórico estimado según PACC (utilizando alguna medida de la información, como la divergencia), sujeto a un set de funciones que permitan representar al sistema en el enfoque SMC asociado y a un rango para los hiperparámetros de aquellas funciones. Así se puede estudiar el efecto de utilizar distintas funciones para representar la evolución del sistema en el respectivo enfoque SMC, y los parámetros de aquellas funciones.

# Conclusión

La metodología presentada permite, por una parte, generar un marco adecuado que facilita el cálculo del máximo error en probabilidad dada una medida, y por tanto permite concluir en cuanto a la precisión computacional de los algoritmos. Por otra parte, se cuestiona el paradigma de comparar momentos de la distribución para evaluar el desempeño, al incorporar la caracterización completa de la distribución al momento de cuantificar el desempeño. En este sentido, se debe poner atención en cuanto al desarrollo de nuevas técnicas que permitan incorporar medidas de información dentro de la descripción y comparación de algoritmos que realicen estimación y predicción del estado de sistemas.

En cuanto al *Ground Truth* para la estimación, se implementa una aproximación de Monte Carlo secuencial que considera toda la información del proceso, para generar una estimación off-line. Sin embargo, se pueden proponer otros métodos como referencia para la comparación, pero, desde un principio, el objetivo de esta Tesis fue generar una referencia sujeta a un modelo en espacio de estado. Ocurre de manera similar para el escenario de pronóstico, donde se asume el modelo y se genera una referencia tomando el *Ground Truth* que describe la estimación y utilizando Monte Carlo secuencial, con el valor real de la corriente, para representar la evolución del sistema. En ambas referencias, se considera el hecho de que la varianza del error de aproximación decrece a una tasa de  $\mathcal{O}(1/N)$ , con  $N$  el número de partículas. Es importante destacar que la construcción de un *Ground Truth* es un elemento crucial para un análisis correcto del desempeño de distintos algoritmos, según la metodología propuesta.

También es importante destacar la posibilidad de aplicar la metodología propuesta a un amplio rango de problemas, comenzando por el hecho de que el filtrado bayesiano mediante filtro de partículas es un caso particular de los algoritmos basados en Monte Carlo Secuencial. Por otro lado, también existen diferentes enfoques que permiten resolver el problema de estimar y predecir la evolución de la distribución de probabilidades asociada al estado de un sistema, donde la metodología propuesta puede ser utilizada para comparar estos enfoques con distintas aproximaciones de SMC u otros métodos.

Es interesante el hecho de que las medidas de información tienen una alta aplicabilidad en cuanto a la forma en la que se asume la descripción del mundo desde que se adopta la perspectiva de la mecánica cuántica. Siendo consistentes con la descripción y medición de fenómenos físicos. En relación a esto último, se debe recordar que en última instancia estos métodos intentan representar un fenómeno que ocurre en la naturaleza -como por ejemplo la medición de la corriente- y que, si se asume que la representación que entrega la

mecánica cuántica de la realidad es asertada, y que el enfoque probabilístico que asumen los métodos secuenciales de Monte Carlo es consistente con ésta representación (que los sistemas se describen por una superposición de estados que coexisten) entonces tiene sentido incorporar medidas de información en algoritmos que intentan capturar una representación probabilística de la realidad/estado de un sistema, antes que una visión determinística del asunto, donde si tiene sentido aplicar medidas que no logran capturar la descripción estadística del estado de un sistema; como ocurre al utilizar sólo los primeros momentos de la distribución.

# Capítulo 5

## Bibliografía

- [1] Edward Balaban Kai Goebel Bhaskar Saha Sankalita Saha Abhinav Saxena, Jose Celaya and Mark Schwabacher. Metrics for evaluating performance of prognostic techniques. *International Conferenc: Prognostics and Health Management*, 2008.
- [2] David Acuña. *Manejo de incertidumbre ante pérdida parcial o total de datos en algoritmos basado en métodos secuenciales de Monte Carlo y nueva definición de probabilidad de falla en el contexto de monitoreo en línea*. Tesis de Magister. Departamento de Ingeniería Eléctrica, Universidad de Chile, Chile, 2016.
- [3] Cesare Alippi. A probably approximately correct framework to estimate performance degradation in embedded systems. *IEEE transactions on computer-aided design of integreted circuits and systems*, 21, 2002.
- [4] Cesare Alippi. Randomized algorithms: A system-level, poly-time analysis of robust computation. *IEEE transactions on computers*, 51, 2002.
- [5] Nando de Freitas Arnaud Doucet and Neil Gordon. *Sequential Monte Carlo Methods in Practice*. Springer, United States, 2001.
- [6] Ristic B. Arumlampalam S., Gordon N. *Beyond the Kalman Filter: particle filters for trackong applications*. Artech House, 2004.
- [7] C. Sankavaram B. Pattipati and K. Pattipati. System identification and estimation framework for pivotal automotive battery management system characteristics. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 2011.
- [8] Er-Wei Bai. Worst-case properties of the uniform distribution and randomized algorithms for robustness analysis. *American Control Conference*, 1997.
- [9] Adrian Bowman and Adelchi Azzalini. *Applied Smoothing Techniques for Data Analysis*. Oxford University Press, United States, 1997.

- [10] Herman Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *The Annals of Mathematical Statistics*, 23, 1952.
- [11] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, United States, 1991.
- [12] M. E. Orchard R. S. Rabié M. A. Cerda B. E. Olivares J. F. Silva P. A. Espinoza D. A. Pola, H. F. Navarrete and A. Pérez. Particle-filtering-based discharge time prognosis for lithium-ion batteries with a statistical characterization of use profiles. *IEEE Transactions on Reliability*, 2015.
- [13] M. Orchard R. Rabié M. Cerda B. Olivares J.Silva P. Espinoza y P. Aramis D. Pola, H. Navarrete. Particle-filtering-based discharge time prognosis for lithium-ion batteries with statical characterization of use profiles. *IEEE Transactions on Reliability*, 2014.
- [14] Arnaud Doucet and Adam M. Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. 2008.
- [15] Robert G Gallager. *Digital Communication*. Cambridge, United States, 2008.
- [16] Ramon Van Handel. *Hidden Markov Models*. 2008.
- [17] Bruce E. Hansen. Lecture notes on nonparametrics. *University of Wisconsin*, 2009.
- [18] L. Györfi EC. van der Meulen J. Beirlant, E. J. Dudewicz. Nonparametric entropy estimation: An overview. *University of Wisconsin*, 2001.
- [19] H.Greenspan J. Goldberger and J. Dreyfuss. Simplifying mixture models using unscented transform. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2008.
- [20] F. Tobar M. Orchard and G. Vachtsevanos. Outer feedback correction loops in particle filtering-based prognosis algorithms: Statical performance comparision. *Studies in Informatics and Controls*, 2009.
- [21] K. Goebel B. Saha M. Orchard, G. Kacprzynski and G. Vachtsevanos. Advances in uncertainty representation and managment for particle filtering applied to prognostics. *Proc. Int. Conf. Prognostics and Health Managment*, 2008.
- [22] G. J. McLachlan and T. Krishnan. The em algorithm and extensions. *NJ: Wiley, Hoboken*, 2008.
- [23] Hugo Navarrete. *Caracterización Estadística del Perfil de Uso de Baterías para el Pronóstico del Estado-de-Carga*. Memoria de Título. Departamento de Ingeniería Eléctrica, Universidad de Chile, Chile, 2014.
- [24] Julien Epps Nguyen Xuan Vinh and James Bailey. Information theoretic measures for clustering comparison: Is a correction for chance necessary? *International Conference on Machine Learning*, 2009.



- [25] G. Orchard, M. Vachtsevanos. A particle-filtering approach for on-line fault diagnosis and failure prognosis. *Transactionn of the Institute of Measurement and Control*, 31, 2009.
- [26] M. Orchard and G. Vachtsevanos. A particle-filtering aproach for on-line fault diagnosis and failure prognosis. *Trans. Inst. Meas. Control*, 2009.
- [27] Emanuel Parzen. On the estimation of a probability density function and mode. *Annals of Mathematical Statics*, 33, 1962.
- [28] D. Pola. An improved prognosis strategy with temperature-dependent state space models for the analysis of the state-of-health and state-of-charge in lithium-ion batteries. *M.Sc. Tesis. Department of Electrical Engineering, Universidad de Chile*, 2014.
- [29] Maria Isabel Ribeiro. *Kalman and Extended Kalman Filters: Concept, Derivation and Properties*. Institute for Systems and Robotics. Instituto Superior Técnico, Portugal, 2004.
- [30] Claude E. Shannon. A mathematical theory of comunication. *The BELL System Technical Journal*, 27, 1948.
- [31] Arnaud Doucet Simon J. Godsill and Mike West. Monte carlo smoothing for nonlinear time series.
- [32] Carlos Tampier. *Análisis comparativo de técnicas avanzadas de estimación bayesiana aplicado al pronóstico del tiempo de descarga de celdas de Ion-Litio*. Facultad de Ciencias Físicas y Matemáticas. Universidad de Chile, Chile, 2014.
- [33] Aman Ullah. Uses of entropy and divergence measures for evalueting econometric approximations and inference. *Journal of Econometrics*, 107, 2002.
- [34] M. Vidyasagar. Statical learning theory and randomized algo- rithms for control. *IEEE Control Systems*, 18, 1998.
- [35] Eric A. Wan and Rudolph van der Merwe. The unscented kalman filter for nonlinear estimatio. *Oregon Graduated Institute of Science and Tecnology*.

# Anexo A

## Tiempos de Procesamiento de Algoritmos

Otro aspecto importante para evaluar el desempeño de los algoritmos de estimación y pronóstico, es el tiempo que les toma procesar la información, pues esta variable está relacionada con el costo computacional del algoritmo. Por esta razón, en la Figura A.1 se observa la relación que existe entre la cantidad de datos a procesar (considerando que el tiempo de muestreo del sistema es 1[seg], entonces 1 dato corresponde a 1[seg] en el eje *instantes estimación*) y el respectivo tiempo que le toma a cada algoritmo procesar aquella información.

Para establecer la relación entre el tiempo de procesamiento y los instantes a estimar, se toma el promedio de 100 realizaciones de la estimación para cada algoritmo, tomando los mismos 4 intervalos utilizados para comparar el desempeño de los algoritmos en el Capítulo 4 (30 %, 50 %, 70 % y 90 %) y luego se escoge la curva polinomial de grado 1 que mejor se ajusta a los datos en el sentido de mínimos cuadrados. En la Figura A.1 los círculos representan el tiempo promedio de procesamiento de las 100 realizaciones, la línea vertical dentro de ellos respresenta la desviación estandar en el tiempo de procesamiento y la línea punteada es la curva ajustada según mínimos cuadrados. Además, en la Tabla A.1 se detallan los coeficientes del ajuste polinomial.

Tabla A.1: Coeficientes polinomio grado 1 asociado a cada algoritmo para estimación

Algoritmo	Coef. $x^1$	Coef. $x^0$
PF	$2,38 \times 10^{-4}$	$1,685 \times 10^{-3}$
UKF	$1,269 \times 10^{-4}$	$1,5 \times 10^{-3}$
UKF OFCL	$1,343 \times 10^{-4}$	$1,065 \times 10^{-3}$

De manera análoga, para los algoritmos estudiados en la etapa de pronóstico, se obtienen 100 realizaciones del tiempo de procesamiento, considerando los intervalo 30 %, 50 % y 70 %; se obtiene el promedio y la desviación estandar, y luego se ajusta una curva polinomial de grado 2 en el sentido de mínimos cuadrados. En la Figura A.2 y la Tabla A.2 se observan

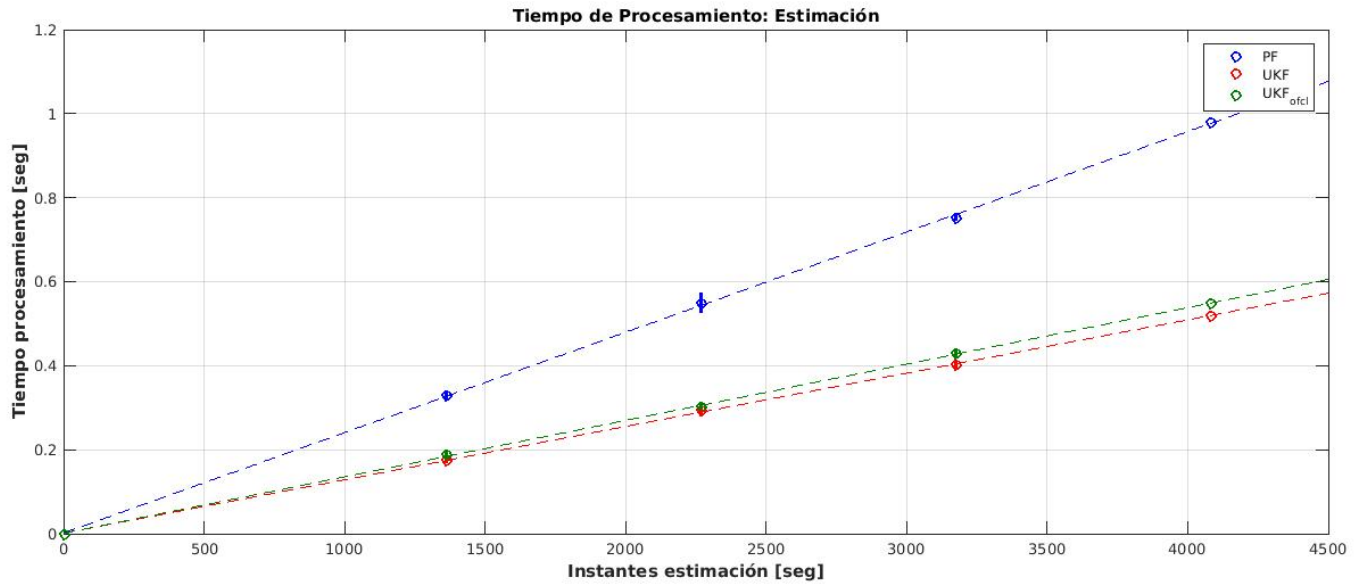


Figura A.1: Tiempo de procesamiento que demora el algoritmo en función de la cantidad total de instantes a estimar. Se consideran los 3 algoritmos utilizados en 4.1.3.

los resultados asociados al algoritmo de pronóstico filtro de partículas regularizado, mientras que en la Figura A.3 y Tabla A.3 se observa la relación que existe entre la cantidad de datos a procesar y el tiempo que le toma al algoritmo filtro de partícula con propagación de incertidumbre via sigma-points.

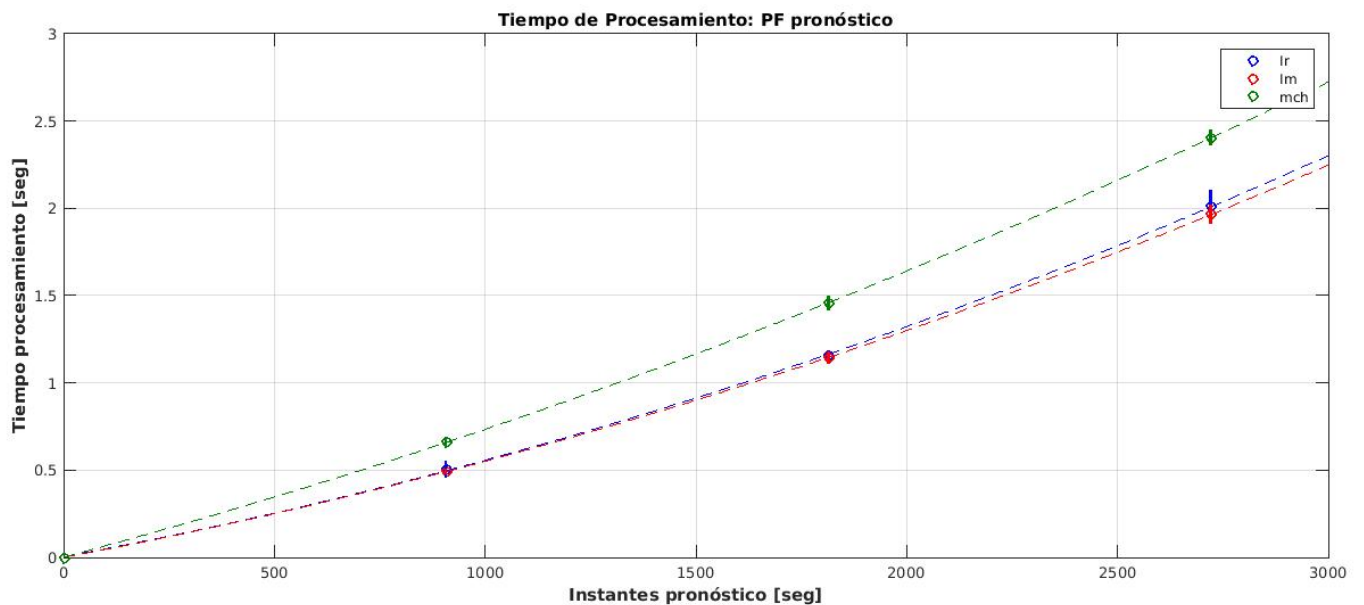


Figura A.2: Tiempo de procesamiento que demora el algoritmo en función de la cantidad total de instantes a pronosticar. Se considera el algoritmos PF Regularizado y los 3 perfiles de corrientes descritos en 4.2.2.

Tabla A.2: Coeficientes polinomio grado 2 asociado a PF Regularizado, considerando 3 perfiles de corriente distintos para pronóstico

Algoritmo	Coef. $x^2$	Coef. $x^1$	Coef. $x^0$
Corriente Real	$1,064 \times 10^{-7}$	$4,47 \times 10^{-4}$	$3,072 \times 10^{-3}$
Corriente Promedio	$9,937 \times 10^{-8}$	$4,51 \times 10^{-4}$	$2,561 \times 10^{-4}$
Corriente C.Markov	$8,742 \times 10^{-8}$	$6,451 \times 10^{-4}$	$1,168 \times 10^{-3}$

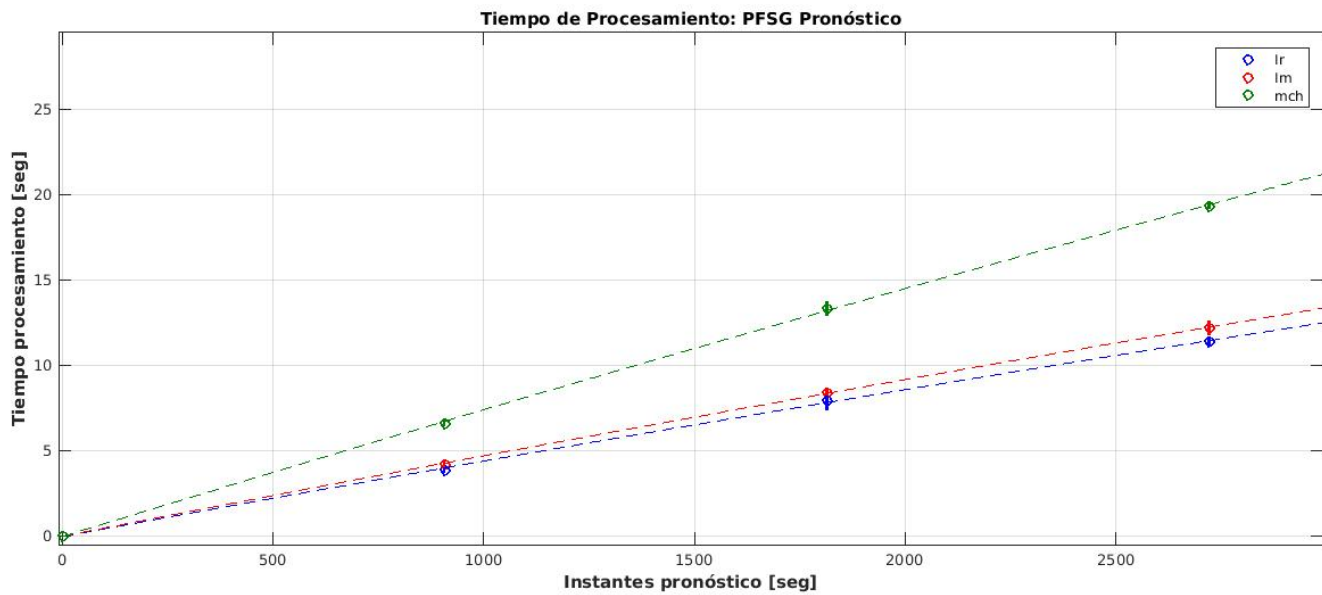


Figura A.3: Tiempo de procesamiento que demora el algoritmo en función de la cantidad total de instantes a pronosticar. Se considera el algoritmos PFSG y los 3 perfiles de corrientes descritos en 4.2.2.

Tabla A.3: Coeficientes polinomio grado 2 asociado a PFSG, considerando 3 perfiles de corriente distintos para pronóstico

Algoritmo	Coef. $x^2$	Coef. $x^1$	Coef. $x^0$
Corriente Real	$-1,187 \times 10^{-7}$	$4,544 \times 10^{-3}$	$-3,776 \times 10^{-2}$
Corriente Promedio	$-1,241 \times 10^{-7}$	$4,842 \times 10^{-3}$	$-2,168 \times 10^{-2}$
Corriente C.Markov	$-1,698 \times 10^{-7}$	$7,611 \times 10^{-3}$	$-4,537 \times 10^{-2}$