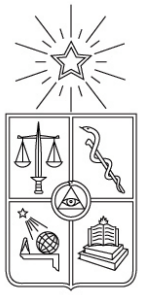


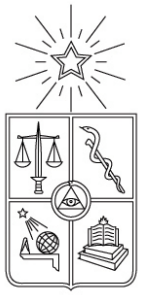
UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

APLICACIÓN DE MÉTODOS SECUENCIALES DE MONTE CARLO SENSIBLES AL RIESGO EN LA ESTIMACIÓN DE VIDA ÚTIL DE COMPONENTES

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELECTRICISTA

DANIEL ALEXIS NEIRA BRAVO

SANTIAGO DE CHILE
JUNIO 2011



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

APLICACIÓN DE MÉTODOS SECUENCIALES DE MONTE CARLO SENSIBLES AL RIESGO EN LA ESTIMACIÓN DE VIDA ÚTIL DE COMPONENTES

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELECTRICISTA

DANIEL ALEXIS NEIRA BRAVO

PROFESOR GUÍA:
MARCOS ORCHARD CONCHA

MIEMBROS DE LA COMISIÓN:
HÉCTOR AGUSTO ALEGRÍA
JORGE SILVA SÁNCHEZ

SANTIAGO DE CHILE
JUNIO 2011

RESUMEN DE LA MEMORIA
PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL ELECTRICISTA
POR: DANIEL ALEXIS NEIRA BRAVO
FECHA: 23 DE JUNIO DE 2011
PROF. GUÍA: MARCOS ORCHARD

APLICACIÓN DE MÉTODOS SECUENCIALES DE MONTE CARLO SENSIBLES AL RIESGO EN LA ESTIMACIÓN DE VIDA ÚTIL DE COMPONENTES

El presente trabajo de memoria se propone como objetivo principal el de analizar el desempeño de los filtros de partículas clásico y sensible al riesgo en el estudio de la vida útil de componentes mecánicos que se encuentren en falla.

Los filtros de partículas, también conocidos como métodos secuenciales de Monte Carlo, corresponden a técnicas de estimación y pronóstico que se sustentan en un marco teórico de trabajo capaz de tratar con problemas en tiempo real, no lineales y con ruidos de proceso y observación no necesariamente *Gaussianos*.

El desempeño de estos métodos de estimación y predicción será estudiado y validado mediante la utilización de datos de laboratorio, provenientes de pruebas realizadas a componentes reales de un helicóptero en falla. En particular, se utilizarán los datos originados a partir del crecimiento de una fractura en una placa de engranaje planetario, pieza que se encuentra presente en el rotor principal de la nave.

La prueba de laboratorio, realizada por terceros, consistió en fracturar artificialmente una placa de engranaje planetario para luego someterla a una carga variable, carga que en un instante de la prueba sufre un cambio. A partir del perfil de vibraciones originado se hizo una estimación ruidosa del crecimiento de la fractura, característica que finalmente es la que se utiliza en este trabajo.

Lo que se espera a partir del estudio de esta señal es evaluar cómo se desempeña el filtro de partículas sensible al riesgo, en comparación con el clásico, en vista del cambio introducido durante el registro de los datos y cómo ello influye en la estimación de la vida útil de la pieza mecánica.

Para implementar los algoritmos de los filtros se hacen diversas consideraciones de cara a disminuir la complejidad del código, proveer una aproximación más simple al problema y reducir la carga computacional asociada con su ejecución. La suposición más importante que se realiza da origen a que durante el proceso de pronóstico los pesos de las partículas se mantengan constantes.

En particular, la implementación de la etapa de predicción para el caso del filtro de partículas sensible al riesgo considera dos casos excluyentes. En el primero, las componentes del ruido de proceso ω_1 y ω_2 distribuyen como *Gaussianas* de media cero y se calcula el valor esperado de los estados. En el segundo caso, $\omega_1 \sim \mathcal{N}(0, 5 \cdot 10^{-4})$ y $\omega_2 = 0$, sin cálculo del valor esperado de los estados.

Las estimaciones de vida útil obtenidas a partir de los filtros de partículas implementados no fueron satisfactorias. Al no existir una actualización de pesos en la etapa de pronóstico, no fue posible corregir el error que se estaba cometiendo en la estimación. Así, la alerta de falla se activa con demasiada anticipación, subestimando el tiempo real de vida útil remanente del sistema.

En conformidad con lo anterior se propone como trabajo futuro la implementación de una técnica encargada de actualizar los pesos de las partículas durante la etapa de pronóstico, así como de una estrategia que sea capaz de evaluar en línea la calidad de las predicciones a largo plazo.

Se concluye a partir del trabajo realizado que para el caso de los filtros de partículas sensibles al riesgo implementados, en cuanto a estimación de vida útil y desempeño computacional, no ofrecen ventajas comparativas claras frente al filtro de partículas clásico.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Marco del trabajo y objetivos

1.1.1. Problemática

El presente trabajo se enmarca dentro de la problemática de estimación de la vida útil de un componente mecánico. En particular, se estudiará el crecimiento de una fractura axial en una placa de engranaje planetario de un helicóptero UH-60A de modo de estimar cuánto tiempo se puede seguir utilizando la nave antes de que se vuelva peligroso hacerlo.

La aparición de la fractura en la placa de engranaje cobra importancia dado que no es exclusiva al UH-60A. Lejos de ser un problema aislado, la aparición de esta fractura se convirtió en un problema de cuidado al verificar que no era específica a un tipo de helicóptero en particular, sino que se podía presentar en diversos modelos.

Debido a que la estimación de la vida útil remanente del sistema se realiza para instantes de tiempo futuros, esta se enfrenta a la problemática de la incertidumbre. El manejo de dicho elemento, sumado a la no linealidad que presentan algunos sistemas, deja espacio a constantes mejoras, por lo que la solución a este tipo de problema de ninguna manera se encuentra cerrada.

1.1.2. Estrategia a utilizar

Para realizar este estudio de vida útil se hará uso de un algoritmo de estimación Bayesiano denominado filtro de partículas, algoritmo que se basa en los métodos de Monte Carlo. Específicamente, un filtro de partículas corresponde a un método secuencial de Monte Carlo.

El filtro de partículas se encarga de generar una población de puntos denominados partículas (muestras del espacio de incógnitas), con pesos asociados que representan la probabilidad discreta de cada una de ellas [15]. Para estimar una señal, los pesos son actualizados convenientemente (para cada partícula) en base a un modelo de planta y a una medición.

El tema de cómo se actualizan las partículas es de especial relevancia en este trabajo. Se utilizarán

dos aproximaciones: la primera dice relación con el filtro de partículas clásico mientras que la segunda se basa en lo que se conoce como filtro de partículas sensible al riesgo.

Una vez finalizada la etapa de estimación se da inicio a la de pronóstico. En esta etapa no se tiene acceso a las mediciones de modo que no se puede verificar directamente la calidad de la predicción. La consecuencia de esto es que la incertidumbre en los valores pronosticados crece rápidamente con el tiempo.

Se estudiará hasta qué punto de la etapa de predicción los resultados surgidos de la utilización de los filtros es fiable y cuál de ellos se comporta de mejor manera. La calidad de la estimación en la vida útil remanente del sistema es consecuencia directa de este estudio.

Finalmente, lo que se busca con la implementación del filtro de partículas sensible al riesgo es estudiar cómo reacciona frente a un cambio en las condiciones del experimento del cual se obtienen las mediciones, en comparación con su par clásico. Se intentará responder qué filtro origina una mejor estimación de la vida útil remanente del sistema.

1.1.3. Objetivos

Este trabajo se expone como un estudio acerca de los efectos que tiene modificar la "función de densidad de importancia" q en el filtro de partículas, desde el punto de vista de exactitud y precisión de la estimación de la vida útil remanente. En particular, se estudiará la elección de una función q que privilegie el seguimiento de un estado crítico del problema para dar origen a un filtro de partículas sensible al riesgo. Con este tipo de filtro se busca poder responder adecuadamente a variaciones o perturbaciones inesperadas del problema y generar una buena aproximación de la vida útil remanente del helicóptero en estudio.

1.2. Organización del documento

Este documento se separa en 6 capítulos, un apartado de referencias bibliográficas y dos anexos. A continuación se presenta la descripción de cada uno de los elementos en los que está dividido el documento:

- **Capítulo 1: Introducción**. El presente capítulo: presentación del trabajo y una breve descripción del contenido del documento.
- **Capítulo 2: Revisión bibliográfica**. Síntesis de los aspectos teóricos que soportan el desarrollo de este trabajo.
- **Capítulo 3: Problema a abordar**. Descripción del problema a abordar: placa de engranaje planetario de un helicóptero UH-60A.
- **Capítulo 4: Implementación del sistema de estimación de Vida Útil Remanente**. Descripción del camino seguido para implementar los algoritmos.

- **Capítulo 5: Discusión de resultados**. Presentación de los resultados obtenidos y comentarios relevantes en torno a ellos.
- **Capítulo 6: Trabajo futuro y conclusiones**. Reflexión acerca de los resultados obtenidos y sugerencias para continuar con el trabajo desarrollado.
- **Referencias**. Detalle de las fuentes de información utilizadas en la confección y del informe y desarrollo del trabajo.
- **Anexos: tablas de datos y código de los programas**. Tablas con los datos numéricos surgidos de los experimentos realizados y el código fuente de los programas escritos para este trabajo.

Capítulo 2

Revisión bibliográfica

Dado un modelo dinámico no lineal y no *Gaussiano* en el espacio de estados, se define el proceso de filtraje no lineal como la estimación, en base a observaciones ruidosas, de al menos los dos primeros momentos de la distribución de probabilidad del vector de estados del modelo [5].

En general, el propósito del filtraje no lineal es, en base a las observaciones que se realizan en un sistema, estimar la probabilidad de que se mida un valor específico para el estado del sistema [5]. En tiempo discreto, esto se puede denotar como $p(x_{0:k}|y_{1:k})$ y se denomina "función de densidad de probabilidad posterior" o "*pdf*[1] posterior".

El problema de estimación se puede plantear tanto en tiempo continuo como discreto, pero en este documento sólo se abordará este último caso. Dado que las mediciones y el procesamiento de datos se realiza de manera digital, se considera de mayor utilidad plantear y resolver el problema directamente en tiempo discreto.

La definición matemática del problema a resolver es la siguiente [10]: sea $X = \{X_k, k \in \mathbb{N}\}$ un proceso estocástico discreto no observable caracterizado por una cadena de Markov de primer orden en $\mathbb{R}^{n_x}$ con condición inicial $p(x_0)$ y probabilidad de transición $p(x_k|x_{k-1})$. Si $\{\omega_k\}_{k\geq 0}$ es una secuencia de variables aleatorias independientes, entonces $p(x_k|x_{k-1})$ se puede describir por (2.1):

$$x_k = f_k(x_{k-1}, \omega_k) \tag{2.1}$$

Por otra parte, si las observaciones ruidosas $Y = \{Y_k, k \in \mathbb{N}\}$ son accesibles y condicionalmente independientes dado X, $p(y_k|x_k)$ se puede describir por (2.2) donde $\{\nu_k\}_{k\geq 0}$ es una secuencia de variables aleatorias independientes:

$$y_k = h_k(x_k, \nu_k) \tag{2.2}$$

En base a (2.1) y (2.2), el problema a resolver consiste en calcular x_k (el estado en el instante de tiempo k) dado que se conoce $y_{1:k}$ (las mediciones hasta dicho instante de tiempo k). Como el problema no es determinístico, para realizar lo anteriormente descrito se debe estimar $p(x_{0:k}|y_{1:k})$

[1]Sigla de "*Probability Density Function*" (función de densidad de probabilidad en inglés).

(*pdf* posterior), lo que desde el punto de vista *Bayesiano* constituye, tal como ya se mencionó, el problema de filtraje propiamente tal.

La estimación de $p(x_{0:k}|y_{1:k})$ se realiza de manera recursiva en dos etapas: predicción y actualización [2]. Para ello se considera que $p(x_0|y_0) = p(x_0)$ y se asume que $p(x_{k-1}|y_{1:k-1})$, la *pdf* requerida en el instante de tiempo $k-1$, está disponible.

La etapa de predicción involucra el cálculo de $p(x_k|y_{1:k-1})$, dado que se conoce $p(x_{k-1}|y_{1:k-1})$, por medio de la ecuación de Chapman-Kolmogorov [2]. Esta ecuación, presentada en (2.3), considera que (2.1) es un proceso de Markov de orden 1 y, por tanto, que se cumple $p(x_k|x_{k-1}, y_{1:k-1}) = p(x_k|x_{k-1})$.

$$p(x_k|y_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|y_{1:k-1})dx_{k-1} \tag{2.3}$$

En la etapa de actualización lo que se hace es calcular $p(x_k|y_{1:k})$ en base a (2.3) y a y_k, medición que ya está disponible en el instante de tiempo k. El cálculo se realiza mediante la expresión (2.4), donde $p(y_k|y_{1:k-1})$ es una constante de normalización que se computa según (2.5) [2].

$$p(x_k|y_{1:k}) = \frac{p(y_k|x_k)p(x_k|y_{1:k-1})}{p(y_k|y_{1:k-1})} \tag{2.4}$$

$$p(y_k|y_{1:k-1}) = \int p(y_k|x_k)p(x_k|y_{1:k-1})dx_k \tag{2.5}$$

Las expresiones (2.3) y (2.4) dan lugar a la solución *Bayesiana* óptima al problema de calcular recursivamente el valor exacto de la *pdf* posterior [2].

A pesar de contar con la solución ya expuesta, esta no siempre puede ser calculada. Es por ello que cuando bajo ciertas hipótesis y requisitos establecidos el cálculo sí puede ser realizado, se habla de un algoritmo óptimo (es decir, que ningún otro algoritmo se comporta de mejor manera que él bajo dichas condiciones). Por su parte, cuando las características del problema en estudio no permiten hacer un cálculo exacto de la solución *Bayesiana* óptima, sino sólo aproximarse numéricamente a ella, se tiene un algoritmo subóptimo.

A continuación se presentan algunos ejemplos de estos algoritmos, óptimos y subóptimos, con particular énfasis en los filtros de partículas, los cuales corresponden a un algoritmo del segundo tipo mencionado.

2.1. Algoritmos óptimos: filtro de Kalman

El filtro de Kalman asume que, en cada instante de tiempo, la *pdf* posterior es *Gaussiana*. De esta forma, si $p(x_{k-1}|y_{1:k-1})$ es una *Gaussiana*, se puede demostrar que $p(x_k|y_{1:k})$ también lo es siempre y cuando se cumplan las siguientes condiciones [6]:

- ω_{k-1} y ν_k provienen de distribuciones *Gaussianas* de parámetros conocidos.

- $f_k(x_{k-1}, \omega_{k-1})$ es conocida y lineal en x_{k-1} y ω_{k-1}.
- $h_k(x_k, \nu_k)$ es conocida y lineal en x_k y ν_k.

Si se satisfacen las condiciones antes descritas, entonces (2.1) y (2.2) se pueden escribir como (2.6) y (2.7), respectivamente [2]:

$$x_k = F_k x_{k-1} + \omega_{k-1} \tag{2.6}$$
$$y_k = H_k x_k + \nu_k \tag{2.7}$$

En el modelo anterior, F_k y H_k corresponden a matrices conocidas que pueden ser variantes en el tiempo.

Si se considera que ω_{k-1} y ν_k son estadísticamente independientes de media cero y con covarianzas variables en el tiempo Q_{k-1} y R_k, respectivamente, el algoritmo que describe el filtro de Kalman se puede representar mediante las siguientes relaciones recursivas [2]:

$$p(x_{k-1}|y_{1:k-1}) = \mathcal{N}(x_{k-1}; m_{k-1|k-1}, P_{k-1|k-1}) \tag{2.8}$$
$$p(x_k|y_{1:k-1}) = \mathcal{N}(x_k; m_{k|k-1}, P_{k|k-1}) \tag{2.9}$$
$$p(x_k|y_{1:k}) = \mathcal{N}(x_k; m_{k|k}, P_{k|k}) \tag{2.10}$$

En las expresiones (2.8), (2.9) y (2.10), $\mathcal{N}(x; m, P)$ denota una *Gaussiana* con argumento x, media m y covarianza P. A su vez, $m_{k|k-1}$, $P_{k|k-1}$, $m_{k|k}$ y $P_{k|k}$ se definen de la siguiente manera [2]:

$$m_{k|k-1} = F_k m_{k-1|k-1} \tag{2.11}$$
$$P_{k|k-1} = Q_{k-1} + F_k P_{k-1|k-1} F_k^t \tag{2.12}$$
$$m_{k|k} = m_{k|k-1} + K_k(y_k - H_k m_{k|k-1}) \tag{2.13}$$
$$P_{k|k} = P_{k|k-1} - K_k H_k P_{k|k-1} \tag{2.14}$$

Por último, también se definen las siguientes expresiones [2]:

$$S_k = H_k P_{k|k-1} H_k^t + R_k \tag{2.15}$$
$$K_k = P_{k|k-1} H_k^t S_k^{-1} \tag{2.16}$$

En las igualdades anteriores, (2.15) corresponde a la covarianza del término correctivo ($y_k - H_k m_{k|k-1}$) y (2.16) a la ganancia del filtro.

2.1.1. Otros algoritmos óptimos

Una alternativa al filtro de Kalman son los métodos basados en cuadrícula ("*Grid-based methods*"). Este tipo de algoritmos son óptimos si el espacio de estados es discreto y posee un número

finito de estados [2].

Si x^i_{k-1}, $i = 1, ..., N$ corresponde al estado en el tiempo $k-1$ y $w^i_{k-1|k-1}$ es la probabilidad condicional de dicho estado en base a las mediciones hasta el instante $k-1$, es decir, $\mathbb{P}(x_{k-1} = x^i_{k-1}|y_{1:k-1}) = w^i_{k-1|k-1}$, entonces la *pdf* posterior en $k-1$ se puede escribir como (2.17) [2]. Nótese que en dicha expresión, $\delta(\cdot)$ corresponde a la función delta de Dirac, función para la cual se cumple la igualdad (2.18) [18].

$$p(x_{k-1}|y_{1:k-1}) = \sum_{i=1}^{N} w^i_{k-1|k-1}\delta(x_{k-1} - x^i_{k-1}) \tag{2.17}$$

$$\int_{a-\epsilon}^{a+\epsilon} g(x)\delta(x-a) = g(a), \text{ para } \epsilon > 0 \tag{2.18}$$

La expresión (2.17), junto a (2.18), lleva a la deducción de (2.19), ecuación de predicción, y (2.20), ecuación de actualización. Ellas se obtienen de (2.3) y (2.5).

$$p(x_k|y_{1:k-1}) = \sum_{i=1}^{N} w^i_{k|k-1}\delta(x_k - x^i_k) \tag{2.19}$$

$$p(x_k|y_{1:k}) = \sum_{i=1}^{N} w^i_{k|k}\delta(x_k - x^i_k) \tag{2.20}$$

En (2.19) y (2.20), $w^i_{k|k-1}$ y $w^i_{k|k}$ se describen mediante las expresiones (2.21) y (2.22), respectivamente.

Para poder utilizar este algoritmo, además de cumplirse las hipótesis mencionadas al comienzo, se asume que $p(x^i_k|x^j_{k-1})$ y $p(y_k|x^j_k)$ son conocidas. Lo anterior no limita la forma que puedan tener estas distribuciones discretas de probabilidad [2].

$$w^i_{k|k-1} \triangleq \sum_{j=1}^{N} w^j_{k-1|k-1} p(x^i_k|x^j_{k-1}) \tag{2.21}$$

$$w^i_{k|k} \triangleq \frac{w^i_{k|k-1} p(y_k|x^i_k)}{\sum_{j=1}^{N} w^j_{k|k-1} p(y_k|x^j_k)} \tag{2.22}$$

2.2. Algoritmos subóptimos: filtro de partículas

Al abordar el problema de la implementación de un filtro de partículas, es de interés estimar la *pdf* posterior $p(x_{0:k}|y_{1:k})$, la distribución marginal $p(x_k|y_{1:k})$ y los valores esperados descritos en la expresión (2.23) para cualquier función $g_k : \mathbb{R}^{n_x} \to \mathbb{R}^{f_k}$ integrable con respecto a la *pdf* [10]:

$$I(f_k) = \mathbb{E}_{p(x_{0:k}|y_{1:k})}[g_k(x_{0:k})] \triangleq \int g_k(x_{0:k}) p(x_{0:k}|y_{1:k}) dx_{0:k} \tag{2.23}$$

Para implementar el filtro se siguen dos etapas [2]: predicción y actualización (filtraje). La etapa

de predicción estima la *pdf a priori* del estado. Esto se realiza utilizando el modelo de la planta (expresiones (2.1) y (2.2)) y el estado predicho en el instante anterior de tiempo, tal como se presentó al comienzo de este Capítulo, obteniendo la expresión (2.24).

$$p(x_{0:k}|y_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{0:k-1}|y_{1:k-1})dx_{0:k-1} \tag{2.24}$$

Por su parte, la actualización (filtraje) considera la observación ruidosa actual y_k, la *pdf a priori* $p(x_{0:k}|y_{1:k-1})$, la función de verosimilitud $p(x_k|y_k)$ y la ley de Bayes para calcular la *pdf* posterior (igualdad (2.25)). Para implementar (2.25) recursivamente se puede utilizar la igualdad (2.26), donde $p(y_k|y_{k-1}) = \int p(y_k|x_k)p(x_k|y_{1:k-1})dx_k$ [10].

$$\begin{aligned}
p(x_{0:k}|y_{1:k}) &= \frac{p(y_{1:k}|x_{0:k})p(x_{0:k})}{p(y_{1:k})} = \frac{p(y_k, y_{1:k-1}|x_{0:k})p(x_{0:k})}{p(y_k, y_{1:k-1})} \\
&= \frac{p(y_k|x_{0:k}, y_{1:k-1})p(y_{1:k-1}|x_{0:k})p(x_{0:k})}{p(y_k|y_{1:k-1})p(y_{1:k-1})} \\
&= \frac{p(y_k|x_{0:k}, y_{1:k-1})p(x_{0:k}|y_{1:k-1})p(y_{1:k-1})p(x_{0:k})}{p(y_k|y_{1:k-1})p(y_{1:k-1})p(x_{0:k})} \\
&= \frac{p(y_k|x_k)p(x_{0:k}|y_{1:k-1})}{p(y_k|y_{1:k-1})} \qquad (2.25) \\
&= \frac{p(y_k|x_k)p(x_k|x_{0:k-1}, y_{1:k-1})}{p(y_k|y_{1:k-1})}p(x_{0:k-1}|y_{1:k-1}) \\
&= \frac{p(y_k|x_k)p(x_k|x_{0:k-1})}{p(y_k|y_{1:k-1})}p(x_{0:k-1}|y_{1:k-1}) \qquad (2.26)
\end{aligned}$$

Como ya se comentó al comienzo del Capítulo, no siempre es posible calcular las integrales anteriores, de modo que es necesario encontrar una manera práctica de dar solución al problema planteado en (2.24) y (2.26). Aquí donde entran en juego los algoritmos subóptimos permitiendo aproximar numéricamente estos valores.

2.2.1. Método secuencial de Monte Carlo: filtro de partículas

Conocido también como "algoritmo de condensación" [9] y "supervivencia del más apto" [7], entre otros, el filtro de partículas corresponde a la utilización de un método de Monte Carlo de manera secuencial (SMC[2] por sus siglas en inglés).

Al implementar un filtro de partículas, lo que se desea hacer es estimar la evolución de un vector de estados en cada instante de tiempo en base a la información que proveen el modelo de la planta y las mediciones. En un caso real, el modelo de la planta puede ser impreciso y las mediciones pueden estar sujetas a ruido y a contratiempos en la calibración y precisión de los instrumentos, de modo que es necesario introducir incertidumbre en la descripción del problema, tal como se hizo en las

[2] *Sequential Monte Carlo.*

expresiones (2.1) y (2.2).

La introducción de incertidumbre lleva a que se estimen densidades de probabilidad en lugar de valores determinísticos para los estados. Numéricamente, lo que se hace es proponer una *pdf* candidata a partir de la cual se pretende aproximar la *pdf* posterior. Esto se realiza actualizando los pesos y posiciones de un conjunto de partículas (muestras del espacio de incógnitas) para cada instante de tiempo. Se espera que en cada actualización este conjunto de partículas sea capaz de describir con suficiente exactitud la *pdf* de los estados.

Se debe estar al tanto que, de hacer uso de un filtro de partículas, en cada iteración la población de partículas degenera[3], disminuyendo la calidad de la estimación. Esto se debe a que la varianza de los pesos sólo puede crecer con el tiempo [3]. De esta forma, cada cierto número (que puede ser variable) de ciclos del algoritmo, el conjunto de las partículas y sus pesos debe ser intervenido mediante un proceso denominado "remuestreo" ("*resampling*" en inglés) el cual se encarga de sortear esta indeseable característica del filtro.

En cuanto al planteamiento matemático del problema de seguimiento de una trayectoria haciendo uso de un filtro de partículas, este se puede expresar de la siguiente forma [10]. Sea $\{\pi_k(x_{0:k})\}_{k\geq 1}$ una secuencia de distribuciones de probabilidad, donde $\pi_k(x_{0:k})$ puede ser evaluada punto a punto hasta una constante de normalización. El objetivo del filtro de partículas es el de generar $N \gg 1$ partículas $\{w_k^{(i)}, x_{0:k}^{(i)}\}_{i=1:N}$, $w_k^{(i)} \geq 0, \forall t \geq 1$ que satisfacen (2.27) [1], expresión en la que φ_k es cualquier función π_k-integrable. Lo anterior permite obtener, de manera aproximada, muestras secuenciales de $\{\pi_k\}$ [10].

$$\sum_{i=1}^{N} w_k^{(i)} \varphi_k(x_{0:k}^{(i)}) \xrightarrow{N\to\infty} \int \varphi_k(x_{0:k})\pi_k(x_{0:k})dx_{0:k} \tag{2.27}$$

Para el caso particular de un filtro bayesiano, $\pi_k(x_{0:k}) = p(x_{0:k}|y_{1:k})$ (*pdf* posterior). Luego, en base a las expresiones (2.1) y (2.2), $\pi_k(x_{0:k})$ se puede escribir como (2.28) [4].

$$\pi_k(x_{0:k}) = p(x_0) \prod_{j=1}^{k} f_j(x_j|x_{j-1}) h_j(y_j|x_j) \tag{2.28}$$

En lo que sigue de esta subsección se presentará uno de los métodos que se pueden utilizar para resolver el problema planteado en (2.27), llamado "Esquema de muestreo por importancia", así como la estrategia que se sugiere emplear para hacer frente al problema de la degeneración de las partículas, que consiste en su "remuestreo".

[3]Que la población de partículas degenere quiere decir que su capacidad para aproximar la distribución buscada decae progresivamente. Sin una estrategia que se encargue de hacer frente a este problema, el fenómeno se refleja en el colapso de todos los pesos en una sola partícula.

Esquema de muestreo por importancia

Para aproximar $\pi_k(x_{0:k})$, la *pdf* posterior en el instante k, se hace uso de la información disponible en el instante de tiempo anterior. Así, si $\{x_{0:k-1}^{(i)}\}_{i=1:N}$ corresponde al conjunto de las N partículas con que se cuenta en $k-1$, distribuidas de acuerdo a $\pi_{k-1}(x_{0:k-1})$, entonces la *pdf* posterior requerida puede ser aproximada por la expresión (2.29).

$$\pi_k^N(x_{0:k-1}) = \frac{1}{N}\sum_{i=1}^{N}\delta(x_{0:k-1} - x_{0:k-1}^{(i)}) \tag{2.29}$$

Sin embargo, lo que realmente se desea obtener no es una aproximación de la *pdf* posterior con las partículas del instante $k-1$, sino estimarla mediante un nuevo conjunto de N partículas $\{\tilde{x}_{0:k}^{(i)}\}_{i=1:N}$ que distribuyan aproximadamente de acuerdo a $\pi_k(\tilde{x}_{0:k}^{(i)})$, lo que se logra mediante un kernel q_k propuesto [10]. Una vez que se ha generado este nuevo conjunto de partículas es posible aproximar $\pi_k(x_{0:k})$ por medio de la expresión (2.30) [1], donde $\{\omega_{0:k}^{(i)}\}_{i=1:N}$ tal que $w_{0:k}^{(i)} \propto w_{0:k}(\tilde{x}_{0:k}^{(i)})$ y $\sum_{i=1}^{N} w_{0:k}^{(i)} = 1$ son los pesos que denotan la relevancia de cada partícula en la estimación [10].

$$\pi_k^N(x_{0:k}) = \sum_{i=1}^{N} w_{0:k}^{(i)}\delta(x_{0:k} - x_{0:k}^{(i)}) \tag{2.30}$$

Como se desprende del párrafo anterior, la generación de nuevas partículas se realiza en base a una función de densidad propuesta denotada q_k y llamada "función de densidad de importancia" ("*Importance density function*"). Haciendo uso de esta función de densidad se puede extender $x_{0:k-1}$ para generar el nuevo conjunto de partículas por medio de $q_k(\tilde{x}_{0:k}|x_{0:k-1})$.

Idealmente, la nueva población de partículas debería aproximar fielmente $\pi_k(\tilde{x}_{0:k})$, pero esto no es así [10]. De hecho, la nueva población distribuye de acuerdo a (2.31) [1].

$$q_k(\tilde{x}_{0:k}) = \int_{X^k} q_k(\tilde{x}_{0:k}|x_{0:k-1})\pi_{k-1}(x_{0:k-1})dx_{0:k-1} \tag{2.31}$$

El problema que surge a partir de las diferencias existentes entre $\pi_k(\tilde{x}_{0:k})$ y $q_k(\tilde{x}_{0:k})$ es abordado por el muestreo por importancia asignando el valor de los N pesos de las partículas de acuerdo a la expresión (2.32) [10].

$$w(\tilde{x}_{0:k}) = \frac{\pi_k(\tilde{x}_{0:k})}{q_k(\tilde{x}_{0:k})} \tag{2.32}$$

Por su parte, en muchos casos se hace imposible evaluar (2.31) directamente, por lo que se sugiere utilizar otra expresión para ella. De este modo, si se asume que la función de densidad de importancia se puede escribir como (2.33), entonces la actualización de los pesos se puede describir mediante (2.34) [10].

$$q_k(\tilde{x}_{0:k}|x_{0:k-1}) = \delta(\tilde{x}_{0:k-1} - x_{0:k-1})q_k(\tilde{x}_k|x_{0:k-1}) \tag{2.33}$$

$$\begin{aligned} w(\tilde{x}_{0:k}) &= \frac{\pi_k(\tilde{x}_{0:k})}{q_k(\tilde{x}_{0:k})} = \frac{\pi_k(\tilde{x}_{0:k})}{\pi_{k-1}(x_{0:k-1})q_k(\tilde{x}_k|x_{0:k-1})} \\ &\propto \frac{\pi_{k-1}(x_{0:k-1})p(y_k|\tilde{x}_k)p(\tilde{x}_k|x_{0:k-1})}{\pi_{k-1}(x_{0:k-1})q_k(\tilde{x}_k|x_{0:k-1})} \\ &\propto \frac{p(y_k|\tilde{x}_k)p(\tilde{x}_k|x_{0:k-1})}{q_k(\tilde{x}_k|x_{0:k-1})} \end{aligned} \tag{2.34}$$

A partir de (2.34), nótese que si se impone $q_k(\tilde{x}_k|x_{0:k-1}) = p(\tilde{x}_k|x_{0:k-1}) = f_k(\tilde{x}_k|x_{k-1})$, la actualización de los pesos se puede calcular por medio de (2.35), dando origen al filtro de partículas denominado "secuencial por muestreo de importancia" (SIS[4] por sus siglas en inglés), el cual es muy simple en forma, pero de mal desempeño si no es implementado junto a una estrategia de "remuestreo" [10].

$$w(\tilde{x}_{0:k}) \propto p(y_k|\tilde{x}_k) = h_k(y_k|\tilde{x}_k) \tag{2.35}$$

Por otra parte, si además de cumplirse (2.33) se impone que q_k satisfaga la expresión (2.36), es decir, que acepte a $q_{k-1}(x_{0:k-1})$ como distribución marginal en $k-1$, entonces (2.32) puede reescribirse de tal forma que sea igual a (2.37) [10].

$$q_k(\tilde{x}_{0:k}) = q_{k-1}(x_{0:k-1})q_k(\tilde{x}_k|x_{0:k-1}) \tag{2.36}$$

$$\begin{aligned} w(\tilde{x}_{0:k}) &= \frac{\pi_k(\tilde{x}_{0:k})}{q_k(\tilde{x}_{0:k})} = \frac{\pi_k(\tilde{x}_{0:k})}{q_{k-1}(x_{0:k-1})q_k(\tilde{x}_k|x_{0:k-1})} \\ &\propto \frac{\pi_{k-1}(x_{0:k-1})}{q_{k-1}(x_{0:k-1})} \frac{p(y_k|\tilde{x}_k)p(\tilde{x}_k|x_{0:k-1})}{q_k(\tilde{x}_k|x_{0:k-1})} \\ &\propto w(x_{0:k-1}) \frac{p(y_k|\tilde{x}_k)p(\tilde{x}_k|x_{0:k-1})}{q_k(\tilde{x}_k|x_{0:k-1})} \end{aligned} \tag{2.37}$$

La expresión (2.37) es de gran utilidad a la hora de implementar el filtro pues provee una expresión recursiva capaz de encargarse de la actualización de los pesos de las partículas. Cabe mencionar que para obtener la igualdad basta simplemente con normalizar todos los pesos.

Nótese que, en una línea similar a la del filtro secuencial por muestreo de importancia, si $q_k(\tilde{x}_{0:k}|x_{0:k-1}) = p(\tilde{x}_k|x_{0:k-1}) = f_k(\tilde{x}_k|x_{k-1})$, entonces se cumple (2.38) y la actualización de los pesos puede ser calculada por medio de (2.39) [10].

$$w(\tilde{x}_{0:k}) \propto w(x_{0:k-1})p(y_k|\tilde{x}_k) = w(x_{0:k-1})h_k(y_k|\tilde{x}_k) \tag{2.38}$$

$$w(\tilde{x}_k^{(i)}) = w_{k-1}^{(i)} h_k(y_k|\tilde{x}_k^{(i)}),\ w_k^{(i)} = \frac{w(\tilde{x}_k^{(i)})}{\sum_{i=1}^{N} w(\tilde{x}_k^{(i)})} \tag{2.39}$$

Si al tipo de filtro de partículas recién expuesto se le añade una etapa de remuestreo, entonces

[4] *Sequential Importance Sampling.*

dará origen a lo que a lo largo del presente documento se denominará "filtro de partículas clásico" o simplemente "filtro clásico". Esta implementación del filtro de partículas es conocida en la literatura como filtro de partículas SIR (*Sequential Importance Sampling Resampling Particle Filter*).

Finalmente, nótese cómo la elección de la función de densidad de importancia q_k es relevante en el rendimiento del filtro de partículas al afectar directamente cómo se actualizan los pesos.

Remuestreo

Los filtros de partículas presentan un problema que, de no ser tratado adecuadamente, atenta considerablemente contra el desempeño del algoritmo. Este problema, denominado degeneración, radica en que luego de algunas iteraciones sólo una de las partículas concentrará un peso relevante, mientras que el de todas las demás será despreciable [2]. La degeneración es un problema que no se puede evitar dado que, por las características del algoritmo, la varianza de los pesos de las partículas sólo puede crecer en el tiempo [3]. No manejar correctamente este inconveniente se traduce en un desperdicio de cálculo computacional en partículas cuyo aporte es despreciable en la estimación [2], por lo que es altamente recomendable implementar estrategias encaminadas a evaluar correctamente el problema para minimizarlo.

En este contexto, el objetivo del remuestreo es el de reemplazar las partículas menos relevantes de la muestra manteniendo aquellas que sí son importantes [10]. Este proceso se ejecuta cada vez que se cumple con un criterio de nivel de degeneración en la muestra de partículas. Para evaluar este nivel de degeneración se observa el incremento de la varianza de los pesos de acuerdo a la expresión (2.40) [10], donde N corresponde al número de partículas utilizadas.

$$N_{\mathit{eff}} = \frac{N}{1 + \mathrm{Var}_{\pi(\cdot|y_{0:k})}(w_{0:k})} \tag{2.40}$$

Sin embargo, (2.40) no se puede evaluar directamente [2]. Es por esto que en la práctica se utiliza un estimador, $\hat{N}_{\mathit{eff}}$, que aproxima el valor deseado [10].

$$\hat{N}_{\mathit{eff}} = \frac{1}{\sum_{i=1}^{N}(w_k^{(i)})^2} \tag{2.41}$$

Para decidir si es necesario hacer un remuestreo se compara $\hat{N}_{\mathit{eff}}$ con N_T, una constante escogida previamente. Así, cada vez que $\hat{N}_{\mathit{eff}} \leq N_T$ se procede a realizar el remuestreo. El valor de N_t puede variar en cada problema, pero basta con asignarle un valor $N_T = 0.85N$ o $N_T = 0.9N$ para comenzar a analizar el desempeño del filtro.

Como se describe en [10], el remuestreo involucra la generación de un nuevo conjunto de partículas $\{\breve{x}_{0:k}^{(i)}\}_{i=1:N}$ a partir de (2.30). Esto se logra muestreando N veces (con reemplazo) tal que $\mathbb{P}(\breve{x}_{0:k}^{(i)} = x_{0:k}^{(j)}) = w_k^{(j)}$. La muestra resultante es i.i.d.[5] por lo que se igualan todos los pesos a $\frac{1}{N}$. De esta forma

[5]Independiente e idénticamente distribuida.

(2.30) se puede reescribir como (2.42).

$$\breve{\pi}_k^N(x_{0:k}) = \sum_{i=1}^{N} w_{0:k}^{(i)} \delta(x_{0:k} - \breve{x}_{0:k}^{(i)}) = \frac{1}{N} \sum_{i=1}^{N} \delta(x_{0:k} - \breve{x}_{0:k}^{(i)}) \tag{2.42}$$

En cuanto a las desventajas introducidas por el remuestreo destaca el "empobrecimiento de la muestra". Como se privilegia la permanencia de las partículas con alto peso, es posible que esto de origen a una conjunto poco diverso. El caso crítico se da cuando el ruido del proceso es pequeño, lo que puede provocar que todas las partículas colapsen a un solo punto en pocas iteraciones [2].

2.2.2. Otros algoritmos subóptimos

Junto a los filtros de partículas existen otros algoritmos que pueden ser utilizados en el tratamiento de sistemas no lineales. En esta subsección se presentan dos de ellos: el filtro de Kalman extendido ("*Extended Kalman filter*") y los métodos aproximados basados en cuadrículas ("*approximate grid-based methods*").

El primer caso, filtro de Kalman extendido, considera la linealización del modelo en torno a un punto de operación en cada instante de tiempo. La aproximación se realiza con el primer término de una expansión en serie de Taylor. Cabe hacer notar que el filtro de Kalman extendido asume que la *pdf* posterior es *Gaussiana*, de modo que si la *pdf* real no lo es, nunca se podrá describir correctamente [2].

Al utilizar el filtro de Kalman extendido, las expresiones (2.8), (2.9) y (2.10) dejan de ser igualdades y se transforman en aproximaciones, tal como se puede ver en (2.43), (2.44) y (2.45) [2].

$$p(x_{k-1}|y_{1:k-1}) \approx \mathcal{N}(x_{k-1}; m_{k-1|k-1}, P_{k-1|k-1}) \tag{2.43}$$

$$p(x_k|y_{1:k-1}) \approx \mathcal{N}(x_k; m_{k|k-1}, P_{k|k-1}) \tag{2.44}$$

$$p(x_k|y_{1:k}) \approx \mathcal{N}(x_k; m_{k|k}, P_{k|k}) \tag{2.45}$$

Las expresiones que describen a $m_{k|k-1}$, $P_{k|k-1}$, $m_{k|k}$ y $P_{k|k}$ se presentan en (2.46), (2.47), (2.48) y (2.49) [2]. Notar que, en forma, no difieren mayormente de las expresiones para el caso óptimo.

$$m_{k|k-1} = f_k(m_{k-1|k-1}) \tag{2.46}$$

$$P_{k|k-1} = Q_{k-1} + \hat{F}_k P_{k-1|k-1} \hat{F}_k^t \tag{2.47}$$

$$m_{k|k} = m_{k|k-1} + K_k(y_k - h_k(m_{k|k-1})) \tag{2.48}$$

$$P_{k|k} = P_{k|k-1} - K_k \hat{H}_k P_{k|k-1} \tag{2.49}$$

Sin embargo, las diferencias son relevantes. En particular, la linealización que incorpora el filtro de Kalman extendido en el modelo está representada en los términos $\hat{F}_k$ y $\hat{H}_k$ (aproximaciones locales de primer orden de las funciones no lineales $f_k(\cdot)$ y $h_k(\cdot)$, respectivamente), los cuales se exponen en

(2.50) y (2.51). A su vez, en (2.52) y (2.53) también se presentan las equivalencias de las igualdades (2.15) y (2.16) [2].

$$\hat{F}_k = \frac{df_k(x)}{dx}\Big|_{x=m_{k-1|k-1}} \tag{2.50}$$

$$\hat{H}_k = \frac{dh_k(x)}{dx}\Big|_{x=m_{k|k-1}} \tag{2.51}$$

$$S_k = \hat{H}_k P_{k|k-1} \hat{H}_k^t + R_k \tag{2.52}$$

$$K_k = P_{k|k-1} \hat{H}_k^t S_k^{-1} \tag{2.53}$$

Por su parte, en el caso de los métodos aproximados basados en cuadrículas, la *pdf* posterior se aproxima dividiendo un espacio de estados continuo en numerosas "celdas" $\{x_k^i : i = 1, \ldots, N_s\}$. Estas N_s celdas son utilizadas para aproximar la distribución, de modo que mientras más fina sea la división del espacio, mejor debería ser esta aproximación, pero a mayor costo computacional. Nótese que si el espacio de estados es infinito, este se debe truncar en algún punto para hacer uso de este algoritmo [2].

En cuanto a las expresiones matemáticas que describen al algoritmo, ellas son básicamente las mismas del método basado en cuadrícula, pero con aproximaciones. Así, (2.17) se transforma en (2.54), (2.19) en (2.55) y (2.20) en (2.56) [2].

$$p(x_{k-1}|y_{1:k-1}) \approx \sum_{i=1}^{N_s} w_{k-1|k-1}^i \delta(x_{k-1} - x_{k-1}^i) \tag{2.54}$$

$$p(x_k|y_{1:k-1}) \approx \sum_{i=1}^{N_s} w_{k|k-1}^i \delta(x_k - x_k^i) \tag{2.55}$$

$$p(x_k|y_{1:k}) \approx \sum_{i=1}^{N_s} w_{k|k}^i \delta(x_k - x_k^i) \tag{2.56}$$

Sin embargo, sí hay diferencias cuando se buscan las expresiones equivalentes a (2.21) y (2.22). Como la cuadrícula escogida contiene regiones continuas en el espacio de estados, las probabilidades a calcular deben ser integradas sobre dichos intervalos. Esta sutileza da origen a las expresiones (2.57) y (2.58), donde $\bar{x}_{k-1}^j$ representa el centro del intervalo *j-ésimo* en el instante de tiempo $k-1$ [2].

$$w^i_{k|k-1} \triangleq \sum_{j=1}^{N_s} w^j_{k-1|k-1} \int_{x \in x^i_k} p(x|\bar{x}^j_{k-1}) dx \tag{2.57}$$

$$w^i_{k|k} \triangleq \frac{w^i_{k|k-1} \int_{x \in x^i_k} p(y_k|x) dx}{\sum_{j=1}^{N_s} w^j_{k|k-1} \int_{x \in x^j_k} p(y_k|x) dx} \tag{2.58}$$

Para simplificar el cálculo, en [2] se sugiere evaluar las expresiones anteriores en el centro de la celda correspondiente a x^i_k, dando origen a las expresiones (2.59) y (2.60), que son las que finalmente se utilizan.

$$w^i_{k|k-1} \triangleq \sum_{j=1}^{N_s} w^j_{k-1|k-1} p(\bar{x}^i_k|\bar{x}^j_{k-1}) \tag{2.59}$$

$$w^i_{k|k} \approx \frac{w^i_{k|k-1} p(y_k|\bar{x}^i_k)}{\sum_{j=1}^{N_s} w^j_{k|k-1} p(y_k|\bar{x}^j_k)} \tag{2.60}$$

2.3. Filtro de partículas sensible al riesgo

Un filtro de partículas como los descritos anteriormente generará partículas basado sólo en la verosimilitud posterior de los estados [14]. Esto implica que si la probabilidad de ocurrencia de unos estados es baja, ellos no necesariamente se seguirán, a pesar de que puedan ser críticos para el sistema.

Así, la idea que está detrás de un filtro de partículas sensible al riesgo (RSPF[6] por sus siglas en inglés) es la de favorecer el seguimiento de los estados que son más críticos para el sistema, aún cuando su probabilidad posterior sea baja. Para ello se genera una mayor cantidad de partículas en las regiones de riesgo del modelo mediante la incorporación de una función de costo en la descripción de la distribución de importancia, como se observa en la expresión (2.61) [10] [14]. Este cambio logra generar más partículas en ciertas regiones del espacio de estados que son de un interés en particular [16].

$$q(\tilde{d}_k, \tilde{x}_k|\tilde{d}^{(i)}_{0:k-1}, x^{(i)}_{0:k-1}, y_{1:k}) = \gamma_k r(d_k) p(d_k, \tilde{x}_k|y_{1:k}) \tag{2.61}$$

Como se describe en [10], en (2.61), d_k es un conjunto de estados de valor discreto que representan modos de falla[7], x_k es un conjunto de estados de valor continuo que describen la evolución del sistema dadas las condiciones de operación, $r(d_k)$ es una función de riesgo positiva que depende del modo de falla y γ_k es una constante de normalización que asegura que (2.61) se trate de una distribución de probabilidad.

[6] *Risk Sensitive Particle Filter.*

[7] Los modos de falla son, en otras palabras, los modos de operación identificados para el sistema. Dependiendo de cómo esté operando la planta, $r(d_k)$ favorecerá la asignación de partículas para los estados críticos del sistema, en desmedro de aquellos que no lo son para dicho modo de funcionamiento.

En cuanto a las desventajas que acusa esta variación del filtro de partículas se encuentra la necesidad de incluir funciones auxiliares [8] (externas) en el modelo que no son necesariamente fáciles de implementar [10]. En [14] se encuentran algunas directrices a seguir para seleccionar correctamente la función de riesgo a incluir en el algoritmo, en base a un proceso de decisión de Markov (MDP[9] por su sigla en inglés) que calcule el riesgo futuro aproximado de las decisiones realizadas para un estado en particular.

2.4. Predicción de eventos catastróficos

2.4.1. Detección de fallas

La detección de fallas es de utilidad en diversos sistemas (mecánicos, eléctricos, etc.) pues juega un rol importante en el logro de una operación que sea fiable y costo-efectiva [10].

Una estrategia que intente prever un evento catastrófico no puede esperar a tener certeza de la existencia de una falla pues el tiempo que se perdería en ello es valioso. Es por esto que en la práctica se emplea un marco de trabajo que, en base a mediciones, sea capaz de declarar una falla con un cierto intervalo de confianza predeterminado [8], prediciendo con ello la vida útil remanente (RUL[10] por sus siglas en inglés) del subsistema en estudio. Lo anterior permite ejecutar las acciones de contingencia pertinentes a un problema de dicha naturaleza de manera anticipada.

2.4.2. Determinación de la vida útil remanente de un sistema

A continuación, en base a lo expuesto en [8], se presentan algunos de los elementos que se pueden emplear a la hora de intentar detectar una falla, así como para determinar su severidad en caso de que ella exista.

Preprocesamiento de datos

La estimación de la RUL de un subsistema en falla se basa fuertemente en la calidad de la información que proveen los sensores presentes en la planta en estudio. De esta forma, si ella es deficiente, la capacidad de detección de fallas se verá mermada, lo que influirá negativamente en la calidad de la estimación de la RUL.

Para intentar dar robustez al sistema de detección de fallas, evitando falsas alarmas, se hace necesario analizar previamente los datos de las mediciones. La validación de ellos es seguido por el preprocesamiento de las señales, paso empleado por ejemplo para reducir la dimensión de los datos y mejorar su relación señal a ruido (SNR[11] por sus siglas en inglés) [8]. El preprocesamiento de las

[8] Funciones que dan cuenta del riesgo asociado con cada modo de falla.
[9] *Markov Decision Process*
[10] *Remaining Useful Life.*
[11] *Signal-to-noise ratio.* Cuantifica la relación entre la potencia de la señal útil y el ruido presente en el sistema.

señales puede considerar, entre otros, la compresión de los datos y su filtrado, así como el cálculo de promedios de la señal y la aplicación de la transformadas de Fourier [8].

Extracción de la "característica" (indicador de condición)

Se denomina característica a la señal originada a partir del preprocesamiento de los datos de medición para efectos de realizar una correcta detección. Esta característica puede obtenerse desde multitud de fuentes, dentro de las cuales se incluyen tanto datos de mediciones como conocimiento del sistema y su historial de funcionamiento.

La correcta selección y extracción de la característica constituye la etapa más importante para lograr detectar y diagnosticar una falla de manera efectiva [8].

Fusión de los datos provenientes de los sensores y fusión de las características

Por una parte, la fusión de los datos de medición que provienen de distintas fuentes permiten minimizar el efecto del ruido sobre ellos e incrementar la calidad de la señal. Lo anterior se traduce en una reducción en la aparición de falsas alarmas y un aumento en la confiabilidad de la detección [8].

Por otro lado, la fusión de las características permite reducir su dimensión para obtener un indicador que incremente la confiabilidad de la detección y simplifique la utilización de indicadores de tipo umbral (para una señal) [8].

Toma de decisiones

El proceso de toma de decisiones se basa en la información que provee el diagnóstico de la falla y en la predicción de su evolución en el tiempo, lo cual permite realizar un estimado de la RUL.

La decisión se debe tomar a partir de una cadena de suposiciones, cálculos, modelos, todos falibles en alguna medida. Es por ello que una decisión certera debe apoyarse en algoritmos que sean capaces de interpretar correctamente las características relevantes extraídas del sistema.

2.5. Filtros de partículas aplicados al pronóstico de sistemas

2.5.1. Consideraciones previas

Un pronóstico (*prognosis* en inglés) puede ser entendido como la generación de predicciones a largo plazo, en ausencia de nuevas mediciones, que describan la evolución temporal de una señal de interés en particular o un indicador de falla [8]. Lo anterior se realiza con el propósito de estimar la vida útil remanente de un componente o subsistema en falla [10].

Como se expuso en el Capítulo 2, los filtros de partículas exigen para su funcionamiento la disponibilidad de mediciones, siendo ellas muy importantes en el correcto desempeño del algoritmo.

Es por esto que, en una primera instancia, los filtros de partículas no constituirían una elección adecuada cuando se desean realizar pronósticos para un sistema.

Para eludir el problema antes mencionado es necesario introducir algunas consideraciones adicionales en la implementación del algoritmo. Estas consideraciones dicen relación con el desarrollo de un procedimiento capaz de proyectar las partículas en el tiempo, aun cuando no existan nuevas mediciones, con un nivel de incertidumbre los más bajo posible [10].

2.5.2. Generación de predicciones a largo plazo

Tal como se describe en [10], la generación de predicciones a p pasos de la *pdf* de los estados puede obtenerse a partir de la expresión (2.1), modelo de la planta, y la estimación actual del estado. El resultado de hacer lo anterior se presenta en (2.62).

$$\begin{aligned}\tilde{p}(x_{k+p}|y_{1:k}) &= \int \tilde{p}(x_k|y_{1:k}) \prod_{j=k+1}^{k+p} p(x_j|x_{j-1}) dx_{k:k+p-1} \\ &\approx \sum_{i=1}^{N} w_k^{(i)} \int \cdots \int p(x_{k+1}|x_k^{(i)}) \prod_{j=k+2}^{k+p} p(x_j|x_{j-1}) dx_{k+1:k+p-1}\end{aligned} \tag{2.62}$$

Para poder dar solución a (2.62), a continuación se presentan dos alternativas: la actualización de pesos y la proyección en el tiempo del valor esperado de los estados.

Actualización de pesos

En este caso, lo que se realiza en primera instancia es calcular el valor esperado de la ecuación (2.1) (la de los estados) para cada instante de tiempo considerando el valor asociado a cada partícula como condición inicial, tal y como se observa en (2.63) [10].

$$\hat{x}_{k+p}^{(i)} = \mathbb{E}[f_{k+p}(\hat{x}_{k+p-1}^{(i)}, \omega_{k+p})]; \; \hat{x}_k^{(i)} = \tilde{x}_k^{(i)} \tag{2.63}$$

Junto a lo anterior también se debe actualizar el peso de las partículas en cada instante de tiempo. Esto se realiza para reflejar que la *pdf* del estado puede cambiar a causa del ruido y la no linealidad del problema [10].

Sin embargo, como se vio en la deducción de las ecuaciones del filtro de partículas, la actualización de los pesos depende de la obtención de nuevas mediciones. Como en la etapa de pronóstico no se cuenta con esta información, es necesario introducir una estrategia que se encargue de sortear este problema.

Como se describe en [10], si se asume que los pesos iniciales $\{w_k^{(i)}\}_{i=1}^N$ representan adecuadamente la *pdf* actual del estado, entonces (2.64) representa, a su vez, la *pdf* predicha del estado en el instante

de tiempo $k+r$.

$$\tilde{p}(x_{k+r}|\tilde{x}_{1:k+r-1}) \approx \sum_{i=1}^{N} w_{k+r-1}^{(i)} \cdot \tilde{p}(x_{k+r}|\tilde{x}_{k+r-1});\ r = 1, \ldots, p \tag{2.64}$$

La expresión (2.64) considera que $\hat{p}(x_{k+r}^{(i)}|\hat{x}_{k+r-1}^{(i)})$ describe la distribución del estado en el instante de tiempo futuro $k+r$, $r = 1, \ldots, p$ cuando se toma $\hat{x}_{k+r-1}^{(i)}$ como condición inicial.

Proyección en el tiempo del valor esperado de los estados

Esta alternativa sugiere que (2.63) modela suficientemente bien la evolución de $\hat{x}_{0:k+r}^{(i)}$ y que por tanto no es necesario actualizar los pesos de las partículas, de modo que ellos se mantienen constantes [10].

El argumento que sostiene esta alternativa se basa en la suposición de que el error asociado a mantener constantes los pesos de las partículas es despreciable con respecto a otras fuentes de error, tales como inexactitudes del modelo e hipótesis erróneas en los parámetros de ruido [10].

2.5.3. Caracterización estadística de la vida útil remanente de piezas de un equipo

Una vez generadas las predicciones a largo plazo, estas pueden ser utilizadas para calcular la probabilidad de falla del sistema en el futuro. Esto se realiza en base al conocimiento empírico que se tenga de las condiciones críticas del sistema, información que se traduce en umbrales para los indicadores de falla, demarcando zonas de riesgo.

Idealmente, las zonas de riesgo debiesen reflejar estadísticamente un historial de información de fallas, definiendo una *pdf* crítica con un límite superior y otro inferior para el indicador de falla (H_{ub} y H_{lb}, respectivamente) [10].

El cálculo de la probabilidad de falla para cualquier instante de tiempo futuro (*pdf* de la RUL) se puede conseguir mediante (2.65). Esta expresión se basa en el hecho que la probabilidad de falla para un valor fijo de un indicador de falla queda determinada por la zona de riesgo y en que $\{w_{k+r}^{(i)}\}_{i=1}^{N}$ representa la probabilidad predicha para el conjunto de trayectorias calculadas [10].

$$\hat{p}_{TTF}(ttf) = \sum_{i=1}^{N} \mathbb{P}(\text{Falla}|X = \hat{x}_{ttf}^{(i)}, H_{lb}, H_{ub}) \cdot w_{ttf}^{(i)} \tag{2.65}$$

Como referencia, en la Figura 2.1 se presenta la estimación y pronóstico de la longitud de una grieta en un sistema mecánico. Junto a ella se puede observar la representación gráfica de algunos de los conceptos tratados recientemente. Por ejemplo, se presenta el umbral en torno al cual la información empírica revela que se puede producir una falla catastrófica. Para modelar esta posibilidad

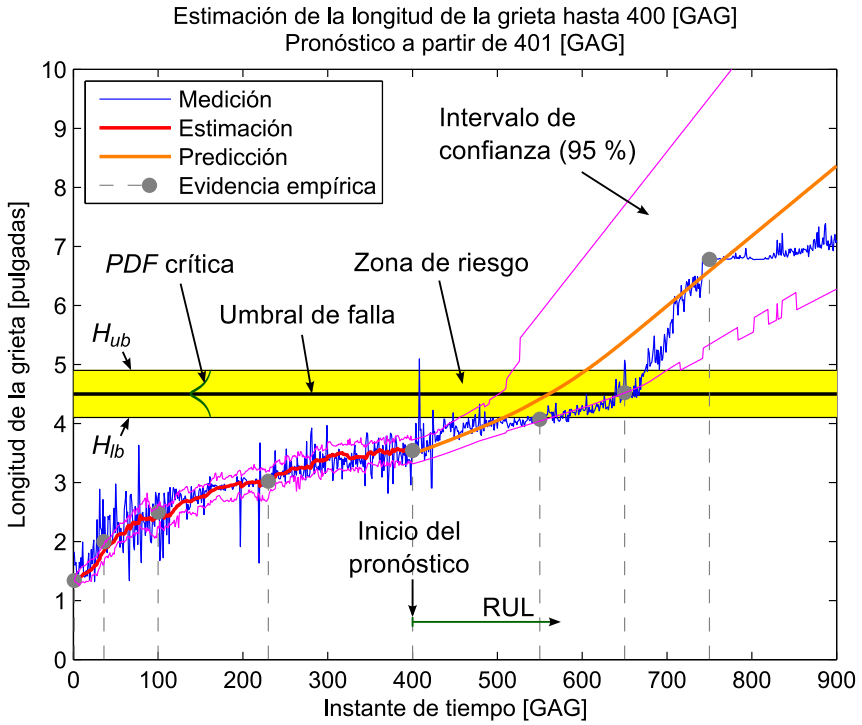


Figura 2.1: *Imagen ilustrativa de algunos conceptos de utilidad para entender el presente trabajo*

se traza una zona de riesgo delimitada superiormente por H_{ub} e inferiormente por H_{lb}. Dentro de dicha zona se define una *pdf* crítica la cual pretende representar la probabilidad de falla en torno al umbral. Y es en base a esta *pdf* que se calcula la RUL, el tiempo de vida útil que le resta al sistema en estudio.

Capítulo 3

Problema a abordar

3.1. Antecedentes

El problema específico que se abordará en el presente trabajo de memoria corresponde a la estimación de la vida útil de una placa de engranaje planetario ("*planetary gear carrier plate*" en inglés) presente en la caja de transmisión del rotor principal de un helicóptero [10]. Una imagen de esta pieza, en su contexto, se puede observar en la Figura 3.1. La estimación se realizará a partir del pronóstico de crecimiento de una fractura en la placa ya mencionada.

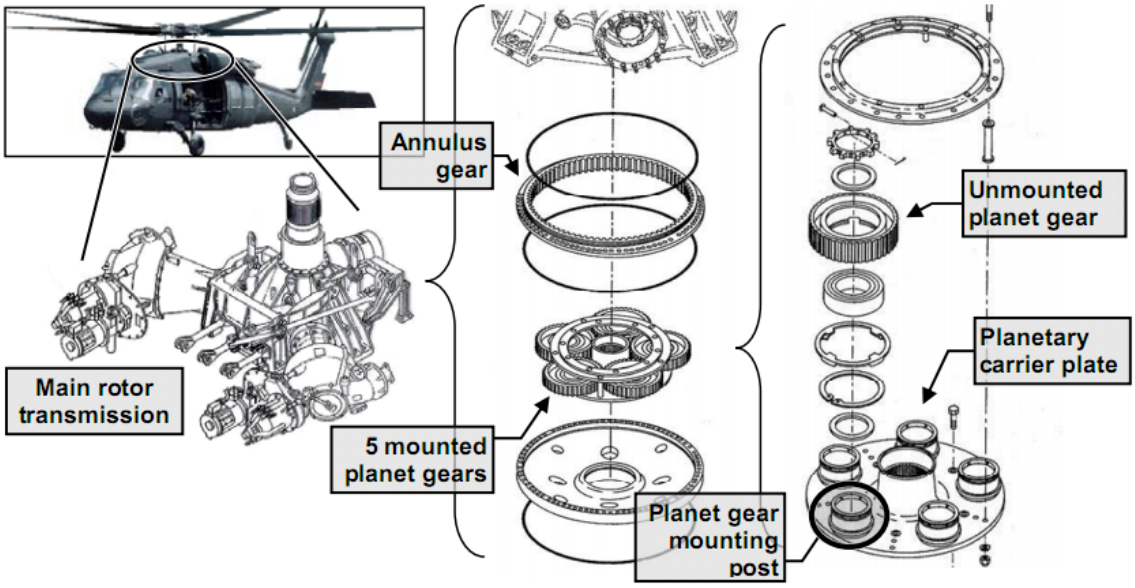


Figura 3.1: *Componentes mecánicos de la transmisión del helicóptero [12]*

El engranaje planetario en estudio es de vital importancia para el funcionamiento del helicóptero ya que se encarga de transmitir la energía mecánica del motor a su hélice principal [12]. Una falla en esta componente durante la operación puede llevar a la pérdida de la aeronave y, lo más importante,

de su tripulación y eventualmente más vidas humanas.

La gravedad del problema fue así fue entendida por el ejército estadounidense luego de detectar anomalías en uno de sus helicópteros modelo UH-60A. Esto los obligó a dejar alrededor de mil de sus aeronaves (incluyendo algunas modelo UH-60Q, todas las UH-60A y varios helicópteros derivados de este último modelo) en tierra para someterlos a revisión [13].

La observación y registro del helicóptero que presentaba anomalías reveló que estas se debían a una fractura en su placa de engranaje planetario. Debido a que posteriormente se encontró el mismo tipo de falla en otras aeronaves, la milicia estadounidense y las compañías involucradas en el diseño y mantención de las máquinas comenzaron a financiar proyectos que se dedicaran a estudiar este inconveniente [12].

3.2. Importancia del estudio de este problema

Si bien se ha mencionado que fue la milicia estadounidense la que se interesó en el problema asociado a la placa de engranaje planetario, este no es de su exclusividad. Helicópteros basados en el modelo afectado son utilizados alrededor de todo el mundo tanto en aplicaciones militares como civiles [12], de modo que el problema es generalizado.

Por un tema de costos y complejidad, rediseñar la placa, reemplazarla en todos los helicópteros en uso y darle mantenimiento no es una opción viable [12]. Es por ello que estrategias que permitan diagnosticar la falla sin necesidad de sacar de servicio las aeronaves cobran vital importancia.

Más aún, ser capaz de estimar la vida útil remanente del helicóptero, en caso de falla, puede resultar de vital importancia a la hora de resguardar las vidas de los pasajeros de la aeronave. La tripulación, al contar con esa información, tiene la posibilidad de escoger como respuesta la mejor estrategia de acuerdo a las circunstancias.

3.3. Obtención de los datos a analizar

3.3.1. El fenómeno de las vibraciones y los datos a utilizar en el estudio

Una falla como la que se ha descrito provoca vibraciones indeseables en el helicóptero, fenómeno que se puede utilizar para detectar la aparición de una grieta en la placa y estimar su crecimiento.

La obtención de los datos de vibraciones se llevó a cabo en una placa previamente intervenida [10] [12]. Esta placa, con una fractura axial de longitud inicial conocida, fue sometida a una carga variable de manera cíclica con el fin de estudiar el crecimiento de la rotura [10].

A partir de los datos de vibraciones se puede obtener una estimación ruidosa de la evolución de la fractura en la placa, estimación que se presenta en la Figura 3.2. Esos datos, ya preprocesados, son los que se utilizarán en este trabajo de memoria para el pronóstico de la vida útil restante del helicóptero cuando la placa se encuentra en falla.

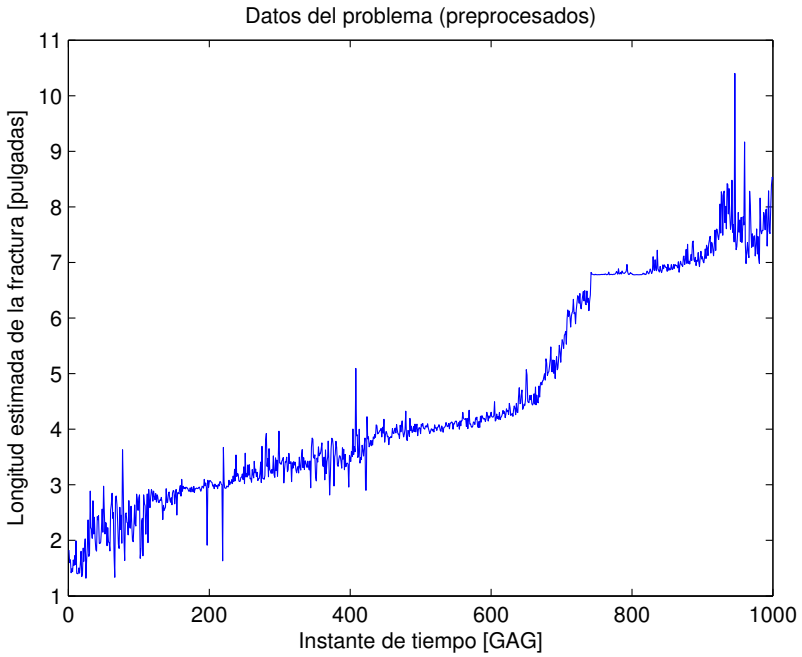


Figura 3.2: *Estimación ruidosa de la longitud de la grieta, a partir de datos de vibraciones, en función de los instantes de tiempo GAG*

3.3.2. Descripción del experimento (toma de datos)

Los datos corresponden a características de falla basadas en datos de vibración de la caja de transmisión, en una prueba experimental llevada a cabo por un agente externo. Lo que se busca es reflejar el comportamiento real de los elementos en estudio.

Para realizar la prueba, primero se hizo crecer artificialmente una grieta en la placa hasta que alcanzara un largo de 1.34 pulgadas. Posteriormente se sometió la transmisión del helicóptero a un torque variable (ciclo menor a 3 minutos) entre el 20 y 120 % de una carga de referencia [10]. A cada ciclo de esta prueba se le denominó GAG[1] y representa el proceso de despegue y aterrizaje del helicóptero. El perfil de carga, en función de los ciclos, se puede observar en la Figura 3.3.

Es importante mencionar que durante la realización de la prueba se decidió disminuir la carga máxima a la que se sometía el equipo llevándola de 120 a 93 %. Se optó por lo anterior debido a que la grieta crecía con una rapidez mayor a la esperada, de modo que disminuyendo el torque máximo aplicado se buscaba poder obtener una mayor cantidad de datos antes de que la placa se fracturara completamente. Esta variación se introdujo a partir del GAG 321.

El cambio en las condiciones de la prueba toma importancia a la hora de analizar el funcionamiento del filtro de partículas, ya que este deberá ser capaz de adaptarse a la variación en el torque máximo aplicado y generar un pronóstico de vida útil remanente acertado. Esto se verificará tanto

[1] *Ground-Air-Ground* o Tierra-Aire-Tierra.

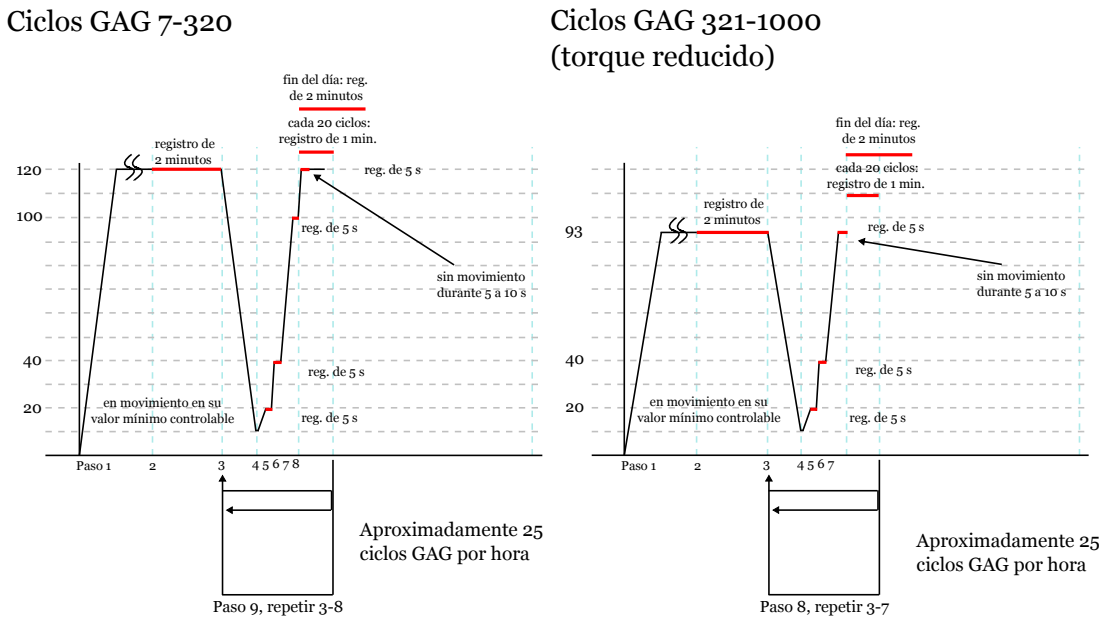


Figura 3.3: *Perfil de carga en función de los ciclos (GAG) [10]*

con el filtro de partículas clásico como con aquel sensible al riesgo, comparando sus desempeños.

Como el experimento se desarrolló en base a condiciones controladas, se tuvo la posibilidad de medir el largo real de la grieta en instantes específicos de tiempo. Estos datos se presentan en la Tabla 3.1.

GAG	Longitud de la fractura (pulgadas)
0	1.34
36	2.00
100	2.50
230	3.02
400	3.54
550	4.07
650	4.52
750	6.78

Tabla 3.1: *Longitud real de la fractura en instantes específicos de tiempo*

A estos datos se les denomina "evidencia empírica" ("*ground truth*" en inglés) y pueden utilizarse para verificar el correcto funcionamiento del filtro de partículas.

3.4. Estrategia a seguir

Para enfrentar el problema descrito anteriormente se utilizará un filtro de partículas. Se escoge este tipo de filtro para estimar el crecimiento de la fractura dado que es una herramienta idónea para analizar problemas no lineales y con ruido no *Gaussiano* y porque, además, este tipo de instrumento es capaz de manejar correctamente la incertidumbre inherente al problema de la predicción a largo plazo [10].

El filtro de partículas se utilizará en primera instancia para estimar el crecimiento de la fractura en la placa a partir de, tal como ya se describió, los datos de la Figura 3.2. Esto replica el trabajo realizado en [10]. Posteriormente se pronosticará el tiempo de vida útil remanente del equipo (con un intervalo de confianza de un 95 %) utilizando como condición inicial los resultados arrojados por el filtro en el último instante de tiempo de la estimación. Esto se comparará, en términos generales, con los resultados presentados en [10] de modo de verificar el correcto funcionamiento del algoritmo.

Una vez realizado todo lo anterior se procederá a implementar la variante sensible al riesgo del filtro, siguiendo los mismos pasos ya descritos, para finalmente comparar su desempeño con los resultados existentes y verificar si se justifica su utilización o no.

Capítulo 4

Implementación del sistema de estimación de Vida Útil Remanente

4.1. Aspectos generales de la modelación

En base a los estudios publicados en [11] se puede utilizar una formulación empírica, que popularmente se conoce como "Ley de Paris", para describir el crecimiento de una grieta. Esta expresión modela la dinámica de crecimiento de la rotura en función del esfuerzo ("*stress*") que sufre. La representación del modelo es simple y se puede expresar como (4.1) [10].

$$\frac{dL}{dn} = C[U(n)\Delta K(n)]^m \tag{4.1}$$

En (4.1), L es la distancia total de la grieta, C y m son coeficientes específicos de cada material, n es el índice del ciclo, $U(n)$ es un parámetro que modela el efecto del "cierre de la grieta" durante el ciclo n y $\Delta K(n)$ corresponde a la variación del *stress* en los extremos de la grieta, en el instante n, medido en $[MN/m^{1.5}]$.

En tiempo discreto, (4.1) puede expresarse, incluyendo nuevos parámetros, como (4.2). Así, nótese la incorporación del término $\alpha(k)$, el cual evoluciona de acuerdo a (4.3). Este parámetro se encarga de corregir la incertidumbre que existe en el valor de la constante C del material. Al estar multiplicando a C, y asumiendo que dicho valor se encuentra cercano al real, se espera que $\alpha(k)$

varíe en torno a 1 para hacerse cargo de las discrepancias que podrían existir.

$$L(k+1) = L(k) + C\alpha(k)[\Delta K_1(k)^m + \Delta K_2(k)^m] + \omega_1(k) \tag{4.2}$$

$$\alpha(k+1) = \alpha(k) + \omega_2(k) \tag{4.3}$$

$$\Delta K_1(k) = f_1(Carga(k), L(k)) = K_1(k)U(k) \tag{4.4}$$

$$\Delta K_2(k) = f_2(Carga(k), L(k)) = K_2(k)U(k) \tag{4.5}$$

$$y(k) = L(k) + \nu(k) \tag{4.6}$$

Por su parte, $\Delta K(n)$ se transforma en $\Delta K_1(k)$ y $\Delta K_2(k)$. El primer término considera el *stress* producido en el extremo externo de la grieta (tomando como referencia la placa) mientras que el segundo al *stress* en el extremo interno. Matemáticamente ellos se describen, respectivamente, mediante (4.4) y (4.5). Las expresiones para $K_1(k)$, $K_2(k)$ y $U(k)$ se calculan en base a datos experimentales para cada instante de tiempo.

Para los datos de fractura estimados que se utilizarán en el presente trabajo (los de la Figura 3.2), los valores de C y m son los siguientes: $C = 7.7331 \cdot 10^{-10}$ y $m = 3.5074$.

Finalmente, téngase en cuenta la inclusión de los ruidos ω_1, ω_2 y ν en el modelo. Por simplicidad, para el caso del filtro de partículas clásico se impone que todos los ruidos mencionados se rigen por una distribución normal de parámetros constantes. En cambio, para el caso del filtro de partículas sensible al riesgo, lo anterior es cierto sólo para los ruidos ω_1 y ν. Esto, ya que ω_2 se asocia con la suma de dos *Gaussianas* de parámetros constantes, ponderadas por números reales a y b respectivamente.

Para la resolución del problema, en la estructura de ambos filtros, los ruidos de proceso y observación se vinculan con dimensiones de falla específicas: ω_1 se relaciona con el ruido de la estimación del largo de la fractura, ω_2 con la variación del parámetro α y ν con la medición de la señal.

4.2. Herramienta utilizada para el desarrollo del trabajo

Para desarrollar la totalidad del trabajo que se presenta en este documento se utilizó exclusivamente el *software* MATLAB®, propiedad de la empresa The MathWorks®. Específicamente se utilizó la versión 7.3.0 (R2006b) para Linux, aunque también se hicieron algunas pruebas bajo la plataforma Microsoft Windows®.

La compatibilidad del código está garantizada puesto que en su desarrollo no se utilizó ninguna función que sea exclusiva de una plataforma en particular o cuyo funcionamiento varíe de una versión a otra de MATLAB®. Además, en general se intentó evitar la utilización de funciones que no sean nativas de MATLAB® por lo que bastaría la versión básica del programa para poder ejecutar las rutinas satisfactoriamente.

El computador donde se llevaron a cabo las simulaciones posee 3 [GB] de memoria RAM y un procesador Intel® Core™2 Duo de 2.2 [GHz].

4.3. Consideraciones respecto a la programación

4.3.1. Estructuras de datos

Buscando la simpleza del código, se decidió hacer uso de estructuras de datos que agruparan todas las variables relevantes en base a un "concepto": por ejemplo, la estructura `particles` (partículas) se entiende como una variable *madre* cuyos *hijos* son `particles.weights` (pesos) y `particles.value` (valores de las partículas), entre otros. Lo anterior favorece el orden y hace más legible e intuitivo el código. En la Tabla 4.1 se presentan todas las variables "madre" de tipo estructura utilizadas en el código fuente. La descripción de sus "hijos" se puede consultar directamente en el código, el cual se encuentra convenientemente comentado para su comprensión.

Estructura	Descripción
`noise_p`	Ruido de proceso
`noise_o`	Ruido de observación (distribución normal)
`model`	Parámetros asociados al modelo
`failure`	Parámetros de la distribución (normal) en torno a umbral de falla
`particles`	Parámetros y variables asociadas con las partículas
`output`	Variables de salida

Tabla 4.1: *Estructuras utilizadas en el código fuente*

4.3.2. Archivos

Para favorecer la claridad se optó por escribir una rutina para cada tipo de filtro. Así, el programa `MAIN.m` corresponde al filtro de partículas clásico mientras que `MAINrs.m` a su variante sensible al riesgo. Las demás funciones implementadas son comunes a ambos filtros salvo por aquella que actualiza el valor y peso de las partículas. En este caso se siguió la misma convención de nombre ya utilizada de modo que `update_particles.m` corresponde a la actualización de las partículas para el filtro clásico y `update_particlesrs.m` al filtro sensible al riesgo.

Si bien se diferenció entre cada filtro al momento de programar, para realizar las pruebas se utilizó un tercer programa al que se denominó `experimentos.m`. Todos los archivos desarrollados y una breve descripción de la finalidad de cada uno de ellos puede consultarse en la Tabla 4.2.

4.3.3. Flujo de programa para las rutinas principales

Los pasos que siguen los programas `MAIN.m` y `MAINrs.m` están bien definidos y, en términos generales, son bastante sencillos de entender. Desde el punto de vista del flujo de los programas, ambos programas realizan los mismos pasos, de modo que basta con centrarse en uno de ellos para entender su funcionamiento. Este se describe en la Figura 4.1.

Básicamente, luego de cargar los datos a analizar e inicializar las variables, cada uno de los

Nombre	Descripción
`MAIN.m`	Rutina principal para el filtro de partículas clásico
`MAINrs.m`	Rutina principal para el filtro de partículas sensible al riesgo
`confidence_interval.m`	Función que calcula el intervalo de confianza de la estimación
`experimentos.m`	Rutina utilizada para realizar las pruebas a los filtros
`failure_prognosis.m`	Función que calcula la probabilidad de falla en base a una distribución normal centrada en un umbral de falla
`filter_data.m`	Función utilizada para suavizar la función de densidad de probabilidad acumulada
`gauss_likelihood.m`	Función que calcula la verosimilitud de una predicción en base a una distribución de probabilidad normal
`gauss_sample.m`	Función que obtiene una muestra de una función de probabilidad normal
`output_mean.m`	Función que calcula la esperanza matemática de un conjunto de datos
`resampling.m`	Función que se ocupa del remuestreo de partículas
`residualresample.m`	Método que utiliza `resampling.m` para realizar el remuestreo
`sog_sample.m`	Función que obtiene una muestra a partir de una suma de *Gaussianas*
`update_particles.m`	Función que se ocupa de actualizar los valores y pesos de las partículas para el filtro clásico
`update_particlesrs.m`	Función que se ocupa de actualizar los valores y pesos de las partículas para el filtro sensible al riesgo

Tabla 4.2: *Estructura de archivos utilizada*

programas entra en un ciclo[1] que se encarga de estimar la longitud de la fractura actualizando los valores y pesos de las partículas, remuestreando si es necesario. La estimación se realiza hasta un tiempo L a partir del cual comienza el pronóstico de crecimiento de la longitud de la fractura. En este ciclo, que se extenderá hasta un tiempo P definido, no se actualizan los pesos de las partículas ni se realiza remuestreo. En este instante de la ejecución del programa también se estima el tiempo en el cual se espera que la pieza fracturada falle.

Para el cálculo del tiempo de falla se utilizan dos criterios: el del valor esperado de las partículas y el del cálculo del valor JIT ("*Just in Time*"). Esto se discutirá mayormente en la Sección 4.8.

4.4. Consideraciones relacionadas con el modelo de la planta

La implementación del modelo en tiempo discreto presentado en la Sección 4.1 se realiza mediante el bloque de instrucciones presente en el Código 4.1. Parte de las líneas allí presentes corresponden a `update_particles.m` mientras que otras provienen de `MAIN.m` (es igual para la versión sensible al

[1]En programación, se denomina ciclo (también conocido como "bucle") a un conjunto de líneas de código que se ejecutan repetidamente hasta que se cumple una condición de término.

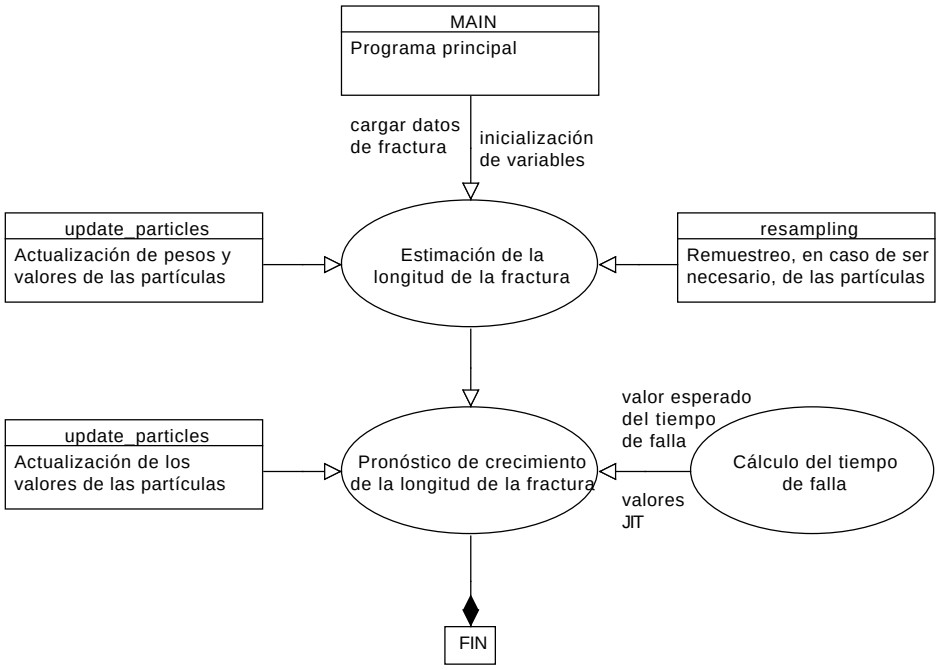


Figura 4.1: *Funcionamiento básico del programa*

riesgo), pero para efectos didácticos y de coherencia se presentan en conjunto.

El código que modela la planta fue extraído del trabajo realizado en [10]. Esto es, las constantes `model.K_1`, `model.K_2`, `model.Lengths`, `model.U_93`, `model.U_100` y `model.U_120` no fueron deducidas específicamente para el trabajo que se presenta en este documento, sino que se tomaron de un modelo de planta ya existente, cuya fiabilidad ha sido probada.

```
model.param.m   = 3.5074;      % constante del material
model.param.C   = 7.7331e-10;  % parámetro cuyo valor se corrige con alpha
model.K_1       = [63.14 63.85 57.95 56.15 61.91 76.91 78.47]./1.0988;
model.K_2       = [80.95 74.78 62.08 59.55 77.63 71.08 79.49]./1.0988;
model.Lengths   = [1.5 2 2.5 3 3.5 4 4.5]./2;
model.U_93      = (0.3195^model.param.m + 0.3195^model.param.m +...
  0.7618^model.param.m + 0.7618^model.param.m + 0.7618^model.param.m +...
  0.7618^model.param.m)^(1/model.param.m);
model.U_100     = (0.3195^model.param.m + 0.3195^model.param.m +...
  0.7618^model.param.m + 0.7618^model.param.m + 0.7618^model.param.m +...
  0.7618^model.param.m + 0.5733^model.param.m)^(1/model.param.m);
model.U_120     = (0.3195^model.param.m + 0.3195^model.param.m +...
  0.7618^model.param.m + 0.7618^model.param.m + 0.7618^model.param.m +...
  0.7618^model.param.m + 0.5733^model.param.m +...
  0.5733^model.param.m)^(1/model.param.m);
```

```
K1 = interp1(model.Lengths,model.K_1,particles.value(1,:)./2,'spline');
K2 = interp1(model.Lengths,model.K_2,particles.value(1,:)./2,'spline');
K1(K1<10) = 10; K2(K2>100) = 100;

U = interp1([93 100 120],[model.U_93 model.U_100 model.U_120],...
  load_u*ones(1,particles.N),'linear','extrap');

DELTA_K = model.param.C.*((U.*K1).^model.param.m+(U.*K2).^model.param.m);
```

Código 4.1: *Implementación del modelo de la planta*

Por su parte, `model.param.m` y `model.param.C` no dependen directamente del modelo, sino que son constantes del material en estudio, aquel donde se produce la fractura.

Finalmente, como se desprende de las expresiones para `K1` y `K2`, el modelo de la planta varía numéricamente con cada actualización de las partículas.

4.5. Filtro de partículas

Para la implementación del filtro de partículas clásico se hicieron algunas consideraciones que simplifican el trabajo, disminuyendo la complejidad computacional en beneficio de la inmediatez del cálculo.

Como se vio en el Capítulo 2, una de las simplificaciones que se pueden utilizar a la hora de implementar un filtro de partículas es considerar que se cumple la igualdad (4.7).

$$q_k(x_k|x_{k-1}) = p(x_k|x_{k-1}) \tag{4.7}$$

De esta forma, la actualización de los pesos de las partículas se rige por (4.8). La igualdad se alcanza normalizando los valores obtenidos.

$$w(x_k) \propto w(x_{k-1})p(y_k|x_k) \tag{4.8}$$

Como se describió en la Sección 4.1, el ruido de observación ν se asume *Gaussiano*. Específicamente, se considera que $\nu \sim \mathcal{N}(0, \sigma_\nu^2)$, de modo que la verosimilitud $p(y_k|x_k)$ se puede calcular directamente del error existente entre la observación y la estimación de la longitud de la fractura. Esto es, basta con restar (4.2) de (4.6) para obtener (4.9), la verosimilud buscada.

$$p(y_k|x_k) = \frac{1}{\sqrt{2\pi\sigma_\nu^2}} e^{\frac{error^2}{2\sigma_\nu^2}} \tag{4.9}$$

La variable *error* se calcula, como ya se mencionó, según (4.10).

$$error = y(k) - L(k) \tag{4.10}$$

Por tanto, la actualización de los pesos de las partículas es directa y se calcula por medio de (4.11) y (4.12).

$$\tilde{w}(x_k) = w(x_{k-1})p(y_k|x_k) \tag{4.11}$$

$$w(x_k) = \frac{\tilde{w}(x_k)}{\sum_{i=1}^{N} \tilde{w}(x_k)} \tag{4.12}$$

En este punto se considera importante mencionar que en el código, al resultado del cálculo de la verosimilitud se le suma $1 \cdot 10^{-99}$, sólo como un artilugio computacional, para evitar errores numéricos.

En cuanto al proceso de actualización de los valores de las partículas, las muestras de los ruidos *Gaussianos* se obtienen simplemente utilizando la función `randn` de MATLAB® de manera adecuada.

4.6. Filtro de partículas sensible al riesgo

La gran diferencia del filtro de partículas sensible al riesgo con su par clásico se da en la actualización de los pesos de las partículas. Recuérdese que, en este caso, el ruido ω_2 (igualdad (4.3)) se impone que distribuya como una suma de dos *Gaussianas* de parámetros constantes.

La elección de la suma de dos *Gaussianas* se realiza para inducir a que el algoritmo del filtro no privilegie tan fuertemente la generación de partículas en torno al valor de α estimado, sino que también dé la oportunidad de aumentar la importancia de partículas que se encuentran alejadas de la estimación. Qué tanto se desea favorecer la generación de partículas alejadas del valor de α estimado se controla mediante el valor de los parámetros a y b, pesos asociados con cada *Gaussiana* considerada en la suma.

Lo que se busca con lo descrito en el párrafo anterior es que filtro de partículas sea más sensible a cambios que se puedan dar en las condiciones del experimento y reaccione con mayor prontitud a ellas.

Como para modelar la primera componente del ruido de proceso se hace uso de una distribución de probabilidad correspondiente a la suma de dos *Gaussianas*, la actualización de los pesos deja de ser directa. Esto se debe a que la expresión (4.7) ya no es válida.

Dado que el ruido de observación utilizado distribuye como una normal, $p(y_k|x_k)$ puede ser calculado directamente, al igual que en el filtro clásico, por medio de la igualdad (4.9). Pero esto no es suficiente para poder actualizar los pesos de las partículas. Para ello se debe calcular, además, $p(x_k|x_{k-1})$ y $q(x_k|x_{k-1})$. Estadísticamente, para todos los efectos de este trabajo se consideran sucesos independientes, de modo que en el primer caso, una normal bivariada, $p(x_k|x_{k-1})$ se puede

calcular de acuerdo a (4.13) [17], donde ω_1 corresponde al ruido asociado con L, la longitud estimada de la fractura (igualdad (4.2)), y ω_2 al ruido relacionado con el parámetro α (igualdad (4.3)).

$$p(x_k|x_{k-1}) = \frac{1}{2\pi\sigma_{\omega_1}\sigma_{\omega_2}} \cdot exp\left(-\frac{\omega_1^2(k)}{2\sigma_{\omega_1}^2}\right) exp\left(-\frac{\omega_2^2(k)}{2\sigma_{\omega_2}^2}\right) \tag{4.13}$$

La deducción de (4.13) queda de manifiesto en (4.14), donde asumir que se tienen sucesos independientes no está tan lejos de la realidad.

$$\begin{pmatrix} \omega_1(k) \\ \omega_2(k) \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{bmatrix} \sigma_{\omega_1}^2 & 0 \\ 0 & \sigma_{\omega_2}^2 \end{bmatrix}\right) \tag{4.14}$$

$$\begin{aligned}
&= \frac{1^2}{\sqrt{2\pi \left|\begin{bmatrix} \sigma_{\omega_1} & 0 \\ 0 & \sigma_{\omega_2} \end{bmatrix}\right|}^2} exp\left(-\frac{1}{2}\begin{bmatrix} \omega_1(k) & \omega_2(k) \end{bmatrix}\begin{bmatrix} \sigma_{\omega_1}^2 & 0 \\ 0 & \sigma_{\omega_2}^2 \end{bmatrix}^{-1}\begin{bmatrix} \omega_1(k) \\ \omega_2(k) \end{bmatrix}\right) \\
&= \frac{1}{2\pi\sigma_{\omega_1}\sigma_{\omega_2}} \cdot exp\left(-\frac{1}{2}\begin{bmatrix} \omega_1(k) & \omega_2(k) \end{bmatrix}\begin{bmatrix} \frac{1}{\sigma_{\omega_1}^2} & 0 \\ 0 & \frac{1}{\sigma_{\omega_2}^2} \end{bmatrix}\begin{bmatrix} \omega_1(k) \\ \omega_2(k) \end{bmatrix}\right) \\
&= \frac{1}{2\pi\sigma_{\omega_1}\sigma_{\omega_2}} \cdot exp\left(-\frac{1}{2}\left(\frac{\omega_1^2(k)}{{\sigma_{\omega_1}}^2} + \frac{\omega_2^2(k)}{{\sigma_{\omega_2}}^2}\right)\right) \\
&= \frac{1}{2\pi\sigma_{\omega_1}\sigma_{\omega_2}} \cdot exp\left(-\frac{\omega_1^2(k)}{2{\sigma_{\omega_1}}^2}\right) exp\left(-\frac{\omega_2^2(k)}{2{\sigma_{\omega_2}}^2}\right)
\end{aligned} \tag{4.15}$$

A su vez, la función de densidad de importancia q (suma de dos *Gaussianas*) lleva a que $q(x_k|x_{k-1})$ se pueda calcular según la expresión (4.16).

$$q(x_k|x_{k-1}) = \frac{1}{2\pi\sigma_{\omega_1}\sigma_{\omega_2}} exp\left(-\frac{\omega_1^2(k)}{2\sigma_{\omega_1}^2}\right)\left[a \cdot exp\left(-\frac{\omega_2^2(k)}{2\sigma_{\omega_2}^2}\right) + b \cdot exp\left(-\frac{(\omega_2(k)-d)^2}{2\sigma_{\omega_2}^2}\right)\right] \tag{4.16}$$

La obtención de la expresión (4.16) se realiza considerando que ella proviene de una distribución de probabilidad conjunta $g_{\omega_1,\omega_2}(\omega_1(k), \omega_2(k))$ para las variables aleatorias $\omega_1(k)$ y $\omega_2(k)$. Dado que se consideran sucesos independientes, entonces se cumple $g_{\omega_1,\omega_2}(\omega_1(k), \omega_2(k)) = g_{\omega_1}(\omega_1(k))g_{\omega_2}(\omega_2(k))$, donde $g_{\omega_1}(\omega_1(k))$ corresponde a la distribución de probabilidad de $\omega_1(k)$ y $g_{\omega_2}(\omega_2(k))$ a la de $\omega_2(k)$.

Considerando que $g_{\omega_1}(\omega_1(k))$ describe a una *Gaussiana* $\mathcal{N}(\mu_1, \sigma_1^2)$ y $g_{\omega_2}(\omega_2(k))$ a la suma de *Gaussianas* $G(a, b, \mu_1, \mu_2, \sigma_1^2, \sigma_2^2)$, la deducción de (4.16) ya se puede entender sin mayores problemas. Ella se presenta en (4.17).

$$\begin{pmatrix} \omega_1(k) \\ \omega_2(k) \end{pmatrix} \sim \left[a \cdot \frac{1}{\sqrt{2\pi\sigma_{\omega_2}^2}} exp\left(-\frac{\omega_2^2(k)}{2\sigma_{\omega_2}^2} \right) + b \cdot \frac{1}{\sqrt{2\pi\sigma_{\omega_2}^2}} exp\left(-\frac{(\omega_2(k)-d)^2}{2\sigma_{\omega_2}^2} \right) \right] \cdot ...$$

$$\frac{1}{\sqrt{2\pi\sigma_{\omega_1}^2}} exp\left(-\frac{\omega_1^2(k)}{2\sigma_{\omega_1}^2} \right) \tag{4.17}$$

$$= \frac{1}{\sqrt{2\pi\sigma_{\omega_2}^2}} \left[a \cdot exp\left(-\frac{\omega_2^2(k)}{2\sigma_{\omega_2}^2} \right) + b \cdot exp\left(-\frac{(\omega_2(k)-d)^2}{2\sigma_{\omega_2}^2} \right) \right] \cdot ...$$

$$\frac{1}{\sqrt{2\pi\sigma_{\omega_1}^2}} exp\left(-\frac{\omega_1^2(k)}{2\sigma_{\omega_1}^2} \right) \tag{4.18}$$

$$= \left[a \cdot exp\left(-\frac{\omega_2^2(k)}{2\sigma_{\omega_2}^2} \right) + b \cdot exp\left(-\frac{(\omega_2(k)-d)^2}{2\sigma_{\omega_2}^2} \right) \right] \cdot ...$$

$$\frac{1}{2\pi\sigma_{\omega_1}\sigma_{\omega_2}} exp\left(-\frac{\omega_1^2(k)}{2\sigma_{\omega_1}^2} \right) \tag{4.19}$$

Finalmente, el resultado de $p(x_k|x_{k-1})/q(x_k|x_{k-1})$ se presenta en (4.20).

$$\frac{p(x_k|x_{k-1})}{q(x_k|x_{k-1})} = \frac{exp\left(-\frac{{\omega_2}^2(k)}{2\sigma_{\omega_2}^2} \right)}{a \cdot exp\left(-\frac{{\omega_2}^2(k)}{2\sigma_{\omega_2}^2} \right) + b \cdot exp\left(-\frac{(\omega_2(k)-d)^2}{2\sigma_{\omega_2}^2} \right)} \tag{4.20}$$

Con todos los resultados presentados ya es posible calcular la actualización de los pesos de las partículas mediante (4.21) y su normalización (que es la que efectivamente se utiliza) por medio de (4.22).

$$\tilde{w}(x_k) = w(x_{k-1}) p(y_k|x_k) \frac{p(x_k|x_{k-1})}{q(x_k|x_{k-1})} \tag{4.21}$$

$$w(x_k) = \frac{\tilde{w}(x_k)}{\sum_{i=1}^{N} \tilde{w}(x_k)} \tag{4.22}$$

En cuanto a la forma de obtener muestras de la distribución q, suma de *Gaussianas*, se optó por trabajar con su función de distribución, más conocida como CDF[2] por sus siglas en inglés. Para ello se tomó en cuenta la expresión (4.23), válida para una $\mathcal{N}(\mu, \sigma^2)$ [19] [20], a partir de la cual se construyó la distribución deseada.

$$CDF(x) = \frac{1}{2} \left[1 + erf\left(\frac{x-\mu}{\sqrt{2\sigma^2}} \right) \right] \tag{4.23}$$

$$erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt \tag{4.24}$$

La forma de la CDF de q, la que se denotará por CDF_q se obtiene al sumar la CDF de una

[2] *Cumulative Distribution Function*. También se conoce simplemente por "*Distribution Function*"

normal $\mathcal{N}(0, \sigma^2_{\omega_1})$ ponderada por un número real a y la CDF de una normal $\mathcal{N}(d, \sigma^2_{\omega_1})$ ponderada por un número real b, con $a > 0$, $b > 0$, $a > b$, $a + b = 1$ y $d < 0$. De esta forma, considerando (4.23), CDF_q queda determinada por (4.25).

$$CDF_q(x) = \frac{a}{2}\left[1 + erf\left(\frac{x}{\sqrt{2\sigma^2_{\omega_2}}}\right)\right] + \frac{b}{2}\left[1 + erf\left(\frac{x-d}{\sqrt{2\sigma^2_{\omega_2}}}\right)\right] \tag{4.25}$$

Así, para obtener muestras de q, lo que se realiza es resolver (4.26) para x, asignándole valores a y de manera aleatoria según una distribución uniforme en el intervalo $]0, 1[$.

$$y = CDF_q(x) \tag{4.26}$$

Como resulta imposible obtener la función inversa de $f(x) = CDF_q(x)$, se hace uso de la función `fsolve` de MATLAB® para dar solución al problema propuesto de manera numérica. Dado que la convergencia a la solución de la rutina `fsolve` implementada en MATLAB® depende fuertemente de los x_0, valores iniciales de x que se le entregan al algoritmo, es necesario realizar varias pruebas preliminares enfocadas en la selección de aquellos x_0 para asegurar el correcto funcionamiento del programa.

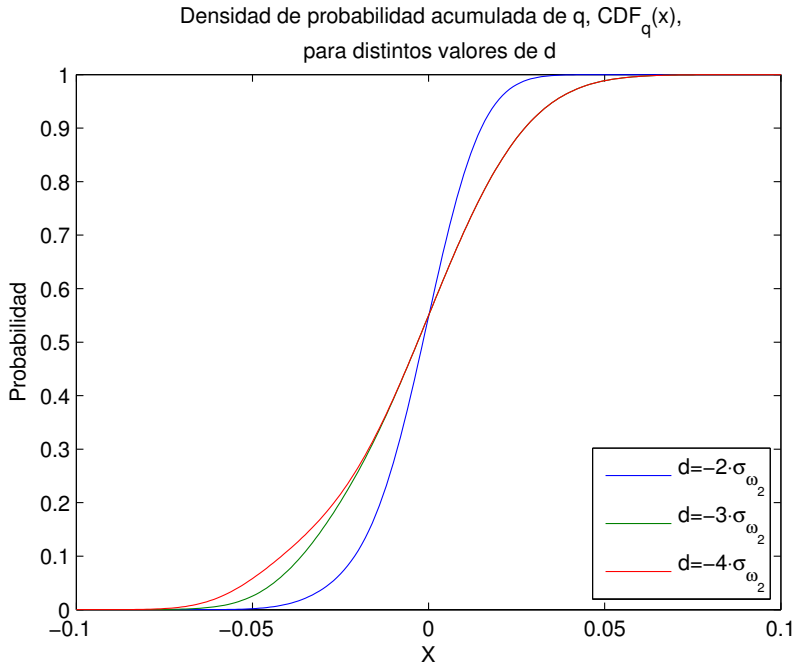


Figura 4.2: *Gráfica de $CDF_q(x)$, con los datos del problema, para distintos valores de d (distancia entre las Gaussianas)*

En la Figura 4.2 se presenta la gráfica de (4.26) utilizando los valores $a = 0.9$, $b = 0.1$, $\sigma^2_{\omega_2} = 1.5 \cdot 10^{-4}$ y d variable, los mismos que se consideraron a la hora de resolver el problema y cuyos

resultados se presentan en la Sección 5.

A partir de la gráfica de (4.26), y por un método de ensayo y error, se determinaron los x_0 necesarios para asegurar la convergencia del algoritmo utilizado en `fsolve`[3]. De esta forma, para todos los valores de d considerados, se dividió el intervalo]0,1[(aquel sobre el que la distribución uniforme considerada genera los números aleatorios) en dos:]0,0.5] y]0.5,1[. En el primer intervalo se utilizó $x_0 = -0.025$ y en el segundo, $x_0 = +0.025$. Con los valores mencionados, `fsolve` convergió a la solución en todas las simulaciones realizadas para este trabajo.

4.6.1. Consideraciones de la implementación

Al implementar el filtro de partículas sensible al riesgo se tuvo en cuenta lo siguiente:

Etapa de filtraje

El ruido de proceso cumple con lo siguiente:

- ω_1 se comportará como una $\mathcal{N}(0, 5 \cdot 10^{-4})$.
- ω_2 se comportará como una suma de *Gaussianas* con $d < 0$. Se dan tres casos excluyentes: $d = -2\sigma_{\omega_2}$, $d = -3\sigma_{\omega_2}$ y $d = -4\sigma_{\omega_2}$, con $\sigma_{\omega_2} = \sqrt{1.5 \cdot 10^{-4}}$.

Etapa de predicción

Se dan dos casos (excluyentes):

1. ω_1 y ω_2 distribuyen como *Gaussianas* de media cero. Se calcula valor esperado.
2. ω_1 distribuye como $\mathcal{N}(0, 5 \cdot 10^{-4})$ y $\omega_2 = 0$. No se calcula valor esperado.

4.7. La predicción y estimación de vida útil remanente

El proceso de filtraje de la señal de entrada para obtener una estimación de la longitud de la fractura es claro y ya ha sido discutido. Ahora es importante explicar qué consideraciones se deben tener en cuenta a la hora de implementar el proceso de predicción.

Cuando se desea estimar una señal con los filtros, esta se observa para comparar y saber qué error se está cometiendo y así corregir en base a aquella información los valores y pesos de las partículas. Por su parte, cuando se realiza la predicción, no existen mediciones con las cuales comparar, por lo que no se puede calcular directamente el error en que se está incurriendo.

Para efectos de la implementación de la etapa de predicción, en el presente trabajo no se consideró ninguna estrategia compleja que intente hacer frente al problema antes mencionado, sino que se

[3]En este caso, dicho algoritmo corresponde a "*Trust Region Dogleg*", el empleado por omisión

adoptó la opción de congelar los pesos de las partículas, haciéndolos constantes e iguales al último valor asignado durante la etapa de filtraje.

El congelamiento de los pesos es un supuesto relativamente fuerte al momento de enfrentar la predicción. Esto se traduce en que las partículas serán tan importantes ("pesarán tanto") en la predicción como lo hicieron en el último instante de tiempo de la estimación.

La simplificación expuesta facilita el proceso de predicción y reduce los cálculos que serían necesarios en caso de implementar una estrategia que se encargara de actualizar los pesos de las partículas.

4.8. Marco de pruebas

Para analizar el funcionamiento de los filtros se decidió tomar en cuenta dos mediciones:

1. El cálculo del tiempo de falla por medio del valor esperado de las partículas.
2. El cálculo del valor JIT ("*Just in Time*").

En el primer caso se toman en cuenta los valores y pesos de las partículas para calcular la estimación de la longitud de la fractura. Se considera que es necesario elevar una alerta de falla cuando este valor es igual o superior a 4.5 pulgadas (umbral predefinido como crítico).

Por su parte, el cálculo del valor JIT se obtiene en base a la CDF de falla y se puede construir a partir de una función de densidad de probabilidad en torno al umbral crítico.

Así, la distribución que se decidió utilizar en torno al umbral de falla corresponde a una normal de parámetros $\mu = 4.5$ y σ_f^2 tal que $\sigma_f \in \{0.1, 0.15, 0.2\}$. Esta distribución se trunca artificialmente en $4.5 \pm 2\sigma$ de modo que para las partículas cuyos valores son inferiores a $4.5 - 2\sigma$ la probabilidad de falla es 0 mientras que para las superiores a $4.5 + 2\sigma$ es 1. Por su parte, el cálculo de la probabilidad de falla para las partículas cuyo valor se encuentra dentro del rango ya especificado se realiza tomando en cuenta la expresión (4.23).

Finalmente, la distribución anterior y el pronóstico del valor de cada partícula (etapa de predicción) permiten construir una CDF de falla. El valor JIT no es más que el tiempo necesario para alcanzar cierto porcentaje de la curva construida (al ser una CDF, el máximo es 1). Para efectos de este trabajo, se buscarán los valores JIT = 5 % y JIT = 10 %.

Capítulo 5

Discusión de resultados

En esta sección se expondrán los resultados obtenidos al aplicar un filtro de partículas SIR y otro RSPF en la estimación y pronóstico del crecimiento de una fractura en un sistema mecánico. Lo que se busca con lo anterior es comparar las estimaciones del tiempo de vida útil remanente del sistema, el cual se modela con las ecuaciones que se presentan a continuación:

$$L(k+1) = L(k) + C\alpha(k)[\Delta K_1(k)^m + \Delta K_2(k)^m] + \omega_1(k) \tag{5.1}$$

$$\alpha(k+1) = \alpha(k) + \omega_2(k) \tag{5.2}$$

$$\Delta K_1(k) = f_1(Carga(k), L(k)) = K_1(k)U(k) \tag{5.3}$$

$$\Delta K_2(k) = f_2(Carga(k), L(k)) = K_2(k)U(k) \tag{5.4}$$

$$y(k) = L(k) + \nu(k) \tag{5.5}$$

En las ecuaciones anteriores, L representa el largo estimado de la fractura y α es un parámetro que intenta corregir la incertidumbre en el valor de la constate C.

Si se quieren obtener más detalles del problema en sí, se recomienda consultar el Capítulo 3. Si en cambio lo que se desea es conocer más a fondo la implementación del sistema, se puede recurrir al Capítulo 4.

A continuación se presentan los resultados obtenidos con el filtro de partículas clásico (SIR) para luego realizar el mismo ejercicio con los resultados del filtro de partículas sensible al riesgo (RSPF). El capítulo finaliza con una comparación entre los resultados obtenidos con cada implementación del filtro de partículas.

5.1. Filtro de partículas clásico (SIR)

5.1.1. Filtraje de la señal

El primer uso que se le da al filtro de partículas se centra en la estimación de la señal de entrada. Dicha señal, al ser ruidosa, es filtrada para poder extraer un valor útil a partir del cual se pueda obtener información relevante. Así, la primera utilidad del filtro de partículas es estimar el valor real de la fractura, lo que se presenta gráficamente en la Figura 5.1.

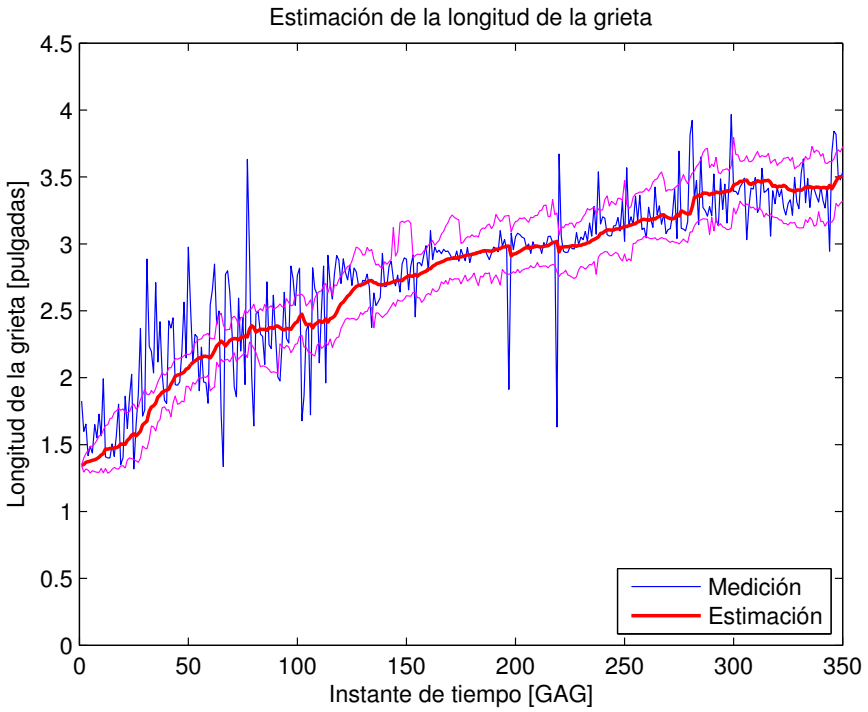


Figura 5.1: *Medición, estimación y su intervalo de confianza (95 %)*

Lo que se aprecia en la Figura 5.1 es la señal ruidosa, la estimación que realiza el filtro de su valor real y el intervalo dentro del cual, con un 95 % de confianza, se encuentra el valor estimado.

En la curva que describe la estimación de la fractura se puede observar claramente la influencia de la *Gaussiana* escogida para modelar el ruido de proceso. Esto, debido a que el valor de la estimación tiene la libertad de disminuir o aumentar para ajustarse a los datos de entrada.

En el presente trabajo se realizan las estimaciones hasta 350 o 400 [GAG], dependiendo de la prueba. Se escogen dichos valores de manera arbitraria con la única condición de ser mayores a 320 [GAG], que es el instante de tiempo a partir del cual, como se comentó en la Subsección 3.3.2, se introduce un cambio en las condiciones del experimento en el que se tomaron los datos. El propósito es estudiar cómo se comporta el filtro haciendo frente a aquella variación implícita en los datos de entrada. Luego, no tiene sentido comenzar a hacer predicción antes que se produzca dicho cambio.

5.1.2. Parámetro α

En la Figura 5.2 se presenta la evolución del parámetro α para tres realizaciones de una prueba, todas ellos con parámetros iguales, de modo que la única diferencia proviene de la aleatoriedad del ruido. Nótese cómo una de las señales expuestas toma un valor constante a partir del GAG 350: se expone dicho fenómeno para reflejar el caso en que se calcula el valor esperado de los estados.

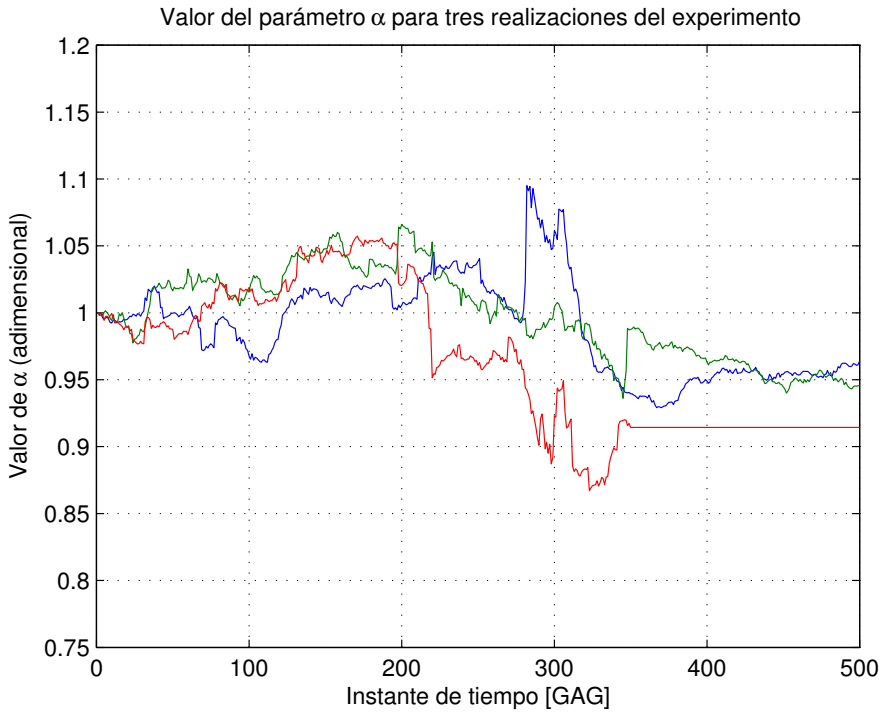


Figura 5.2: *Evolución del parámetro α en un horizonte de tiempo de 700 GAG*

Como se mencionó en el Capítulo 4, el término α del modelo se encarga de corregir C, parámetro cuyo valor es incierto. En el caso ideal el valor de α no debiera variar demasiado de su condición inicial, pero como la planta se encuentra en falla y dado que existen cambios en el perfil de carga y en los parámetros del experimento durante su ejecución, este supuesto no puede asumirse como válido durante toda la ventana de tiempo estimada.

En efecto, lo anterior se refleja en una variación evidente del valor de α, aún antes de que se produzca el cambio en las condiciones del experimento en el que se tomaron los datos.

En base a las tres realizaciones del experimento se desprende que el desempeño del filtro no es suficientemente consistente.

5.1.3. Pronóstico de falla catastrófica

Sin considerar el cálculo del valor esperado de la trayectoria para predicción a largo plazo

La influencia de la distribución normal escogida para modelar el ruido de proceso también influye en el ámbito de la predicción. Como el valor predicho de las partículas en el instante $k+1$ puede ser mayor o menor al del instante k, al trabajar con los datos se pueden producir fenómenos no deseados. En la Figura 5.3 se puede apreciar uno de ellos.

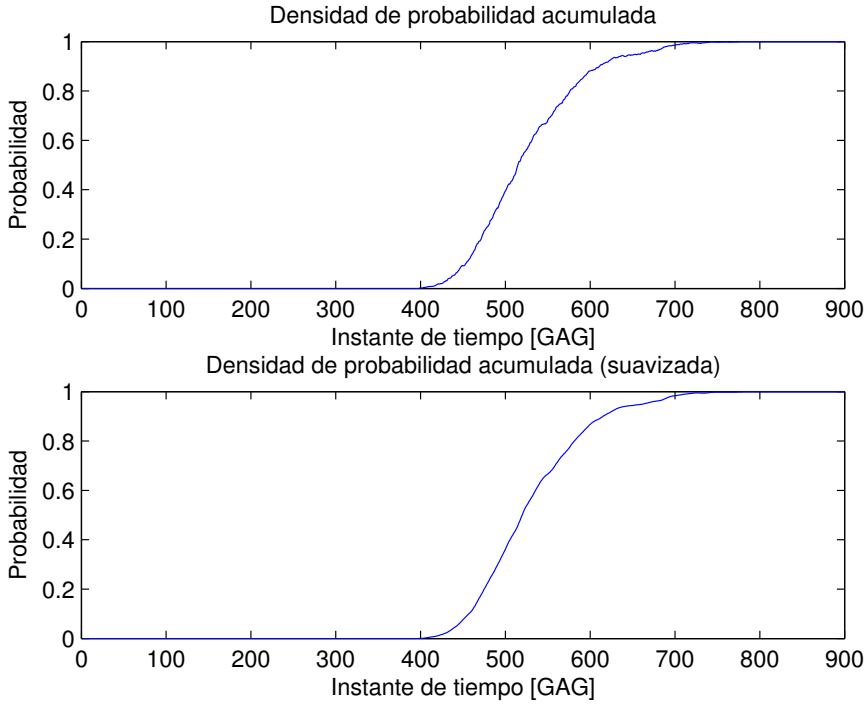


Figura 5.3: *CDF de la distribución de predicción de falla ("real" y suavizada)*

En la imagen de la Figura 5.3 se presenta la CDF de la distribución de falla. Una CDF necesariamente es no decreciente, pero debido al ya mencionado fenómeno que se da con el valor de las partículas, la probabilidad de falla acumulada a veces puede disminuir.

Esta disminución en el valor de la CDF se puede apreciar claramente en la Figura 5.4. Allí se presenta una ampliación, entre los GAG 540 y 550, de la curva sin suavizar expuesta anteriormente.

Para hacer frente a este contratiempo, el cual bajo las simplificaciones realizadas en este trabajo es inevitable, se opta por suavizar la curva calculando un promedio móvil con una ventana de tamaño 10 [GAG].

La estrategia anterior tiene influencia también en el cálculo de la PDF de falla, cuestión evidente en la Figura 5.5. En este caso, más que la respuesta a un contratiempo, la construcción de la PDF a partir de la CDF suavizada facilita la lectura e interpretación del funcionamiento del filtro de

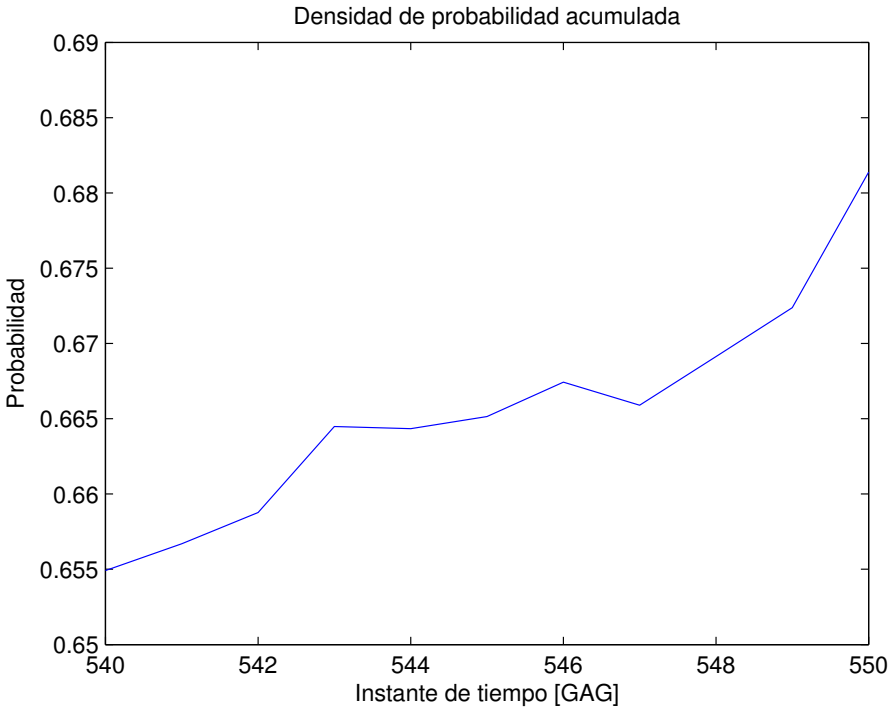


Figura 5.4: *CDF de la distribución de predicción de falla ("real" y suavizada)*

partículas en la predicción.

En la Figura 5.5, la banda de color amarillo representa la zona de riesgo, que corresponde al intervalo dentro del cual se calcula la probabilidad de falla. De esta forma, cuando una partícula ingresa a dicho intervalo, la PDF lo registra con un aumento del área bajo ella. Para visualizar esta idea, tómese como ejemplo el GAG 400: se puede observar cómo las primeras partículas que ingresan a la zona de riesgo provocan un aumento de la densidad de probabilidad y, por ende, del área bajo la curva.

Por su parte, tómese ahora en cuenta el GAG cercano a 800: todas las partículas han superado la banda y se encuentran por sobre ella. Esto se traduce en que ya no pueden aportar con una aumento en la probabilidad de falla (el área ya suma 1), de modo que la curva que representa la PDF disminuye hasta 0 (no se aporta más área bajo la curva).

Nótese que esta visualización de la probabilidad de que se produzca la falla es útil, pero a veces puede encontrarse con problemas. Si bien no se puede apreciar directamente en la Figura 5.5, sí se puede intuir que en ocasiones las partículas que han superado el límite superior de la banda pueden empezar a disminuir al punto de volver a ingresar a ella. Es lo que podría haber ocurrido inmediatamente después del GAG 800 si es que alguna de las partículas allí presentes hubiese seguido disminuyendo de valor en lugar de aumentar.

Otro fenómeno no deseado que puede ocurrir es que algunas partículas, aun transcurrido todo el

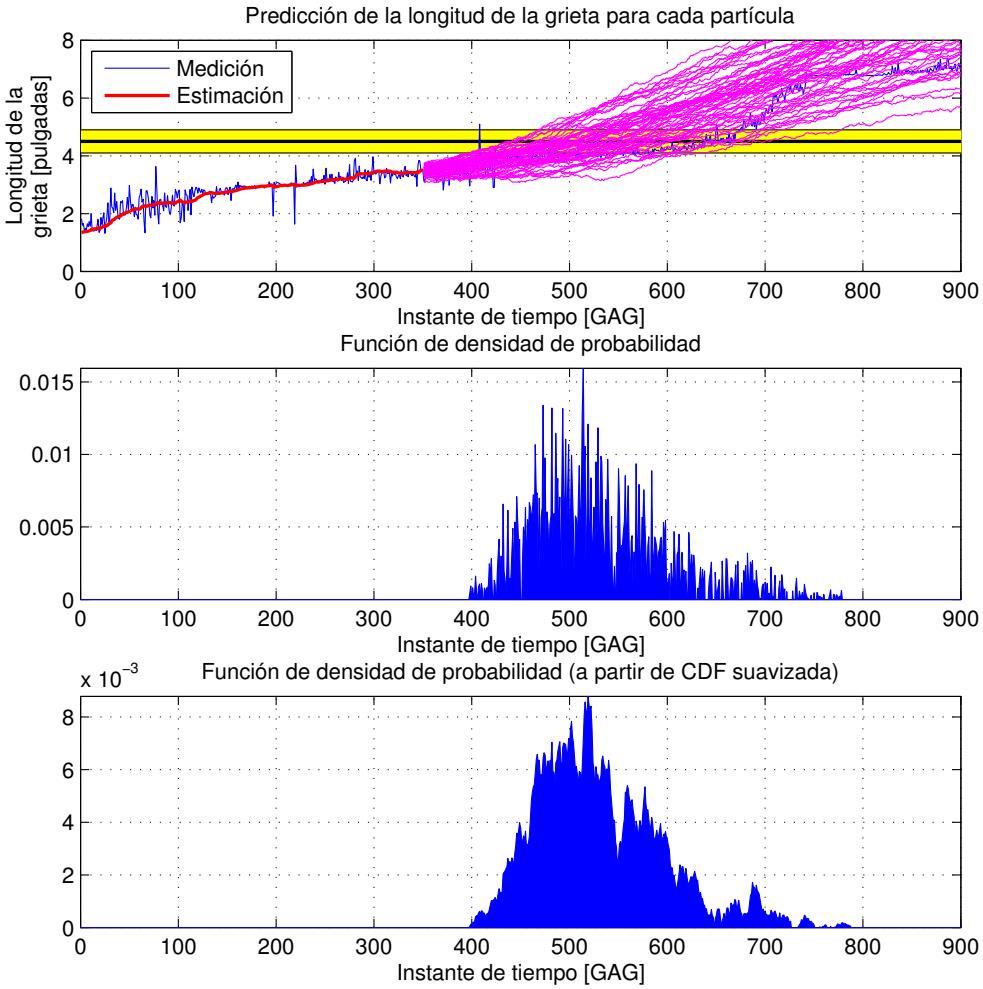


Figura 5.5: *En la imagen superior: la evolución de cada una de las partículas (y en amarillo el intervalo que cubre la distribución normal en torno al umbral de falla). En la imagen central: PDF de la distribución de predicción de falla ("real"). En la imagen inferior, la PDF antes mencionada, suavizada*

tiempo de predicción, no hayan sido capaces de superar H_{ub}, el límite superior de la zona de riesgo. Esto lleva a que la CDF nunca alcance el valor 1, pero asumiendo que si se otorgara mayor tiempo de predicción sí se alcanzaría la unidad, esto deja de ser un contratiempo de cuidado.

Lo descrito en los dos párrafos anteriores introduce "ruido" en la visualización del fenómeno de predicción, mas no en su interpretación. Tampoco influye en el pronóstico de tiempo de falla puesto que para declararla no es necesario esperar a tener 100 % de certeza de que ella ocurrirá. El aviso se produce antes, ya que el hecho de esperar a tener total certeza claramente juega en contra de las medidas que se puedan tomar, restando tiempo de reacción ante la eventualidad de un problema grave.

Considerando el cálculo del valor esperado de la trayectoria para predicción a largo plazo

El enfoque presentado anteriormente deja espacio a la posibilidad de que la longitud predicha de la fractura pueda aumentar o disminuir con respecto al instante de tiempo anterior. Para limitar dicha posibilidad se puede trabajar con los valores esperados de los estados en el tiempo.

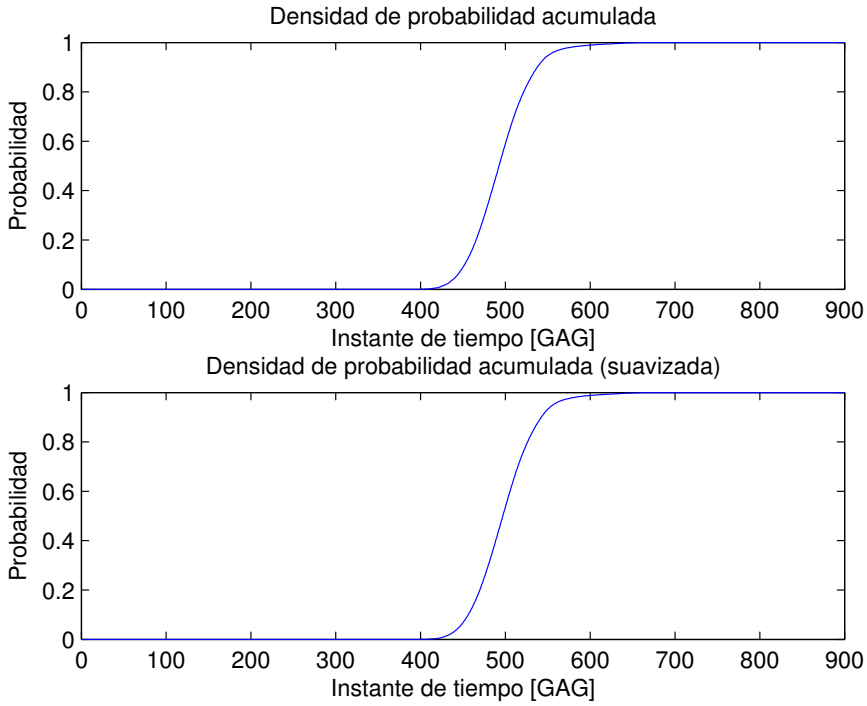


Figura 5.6: *CDF de la distribución de predicción de falla ("real" y suavizada)*

Dado que el ruido de proceso es *Gaussiano* y aditivo, el valor esperado de la trayectoria puede calcularse considerando una suma entre la parte determinística del modelo y la esperanza de dicho ruido de proceso, operación cuyo resultado se puede apreciar gráficamente en la Figura 5.7.

Nótese que, dado que la parte aleatoria ha sido mitigada con el cálculo del valor esperado, la PDF "real" que describe la distribución de predicción de falla es mucho más lisa que aquella que se puede apreciar en la figura 5.5. Luego, al suavizarla, su forma no cambia de manera tan evidente como en el caso que describe la figura ya referida.

Lo mismo se puede observar en la Figura 5.6: la CDF calculada a partir de los datos que no han sido procesados es suave y no disminuye en ningún instante de tiempo.

Finalmente, las trayectorias de las partículas expuestas en la Figura 5.7 también dejan entrever el cálculo del valor esperado: ellas ahora son suaves y no presentan las oscilaciones que se dejaban ver en la Figura 5.5.

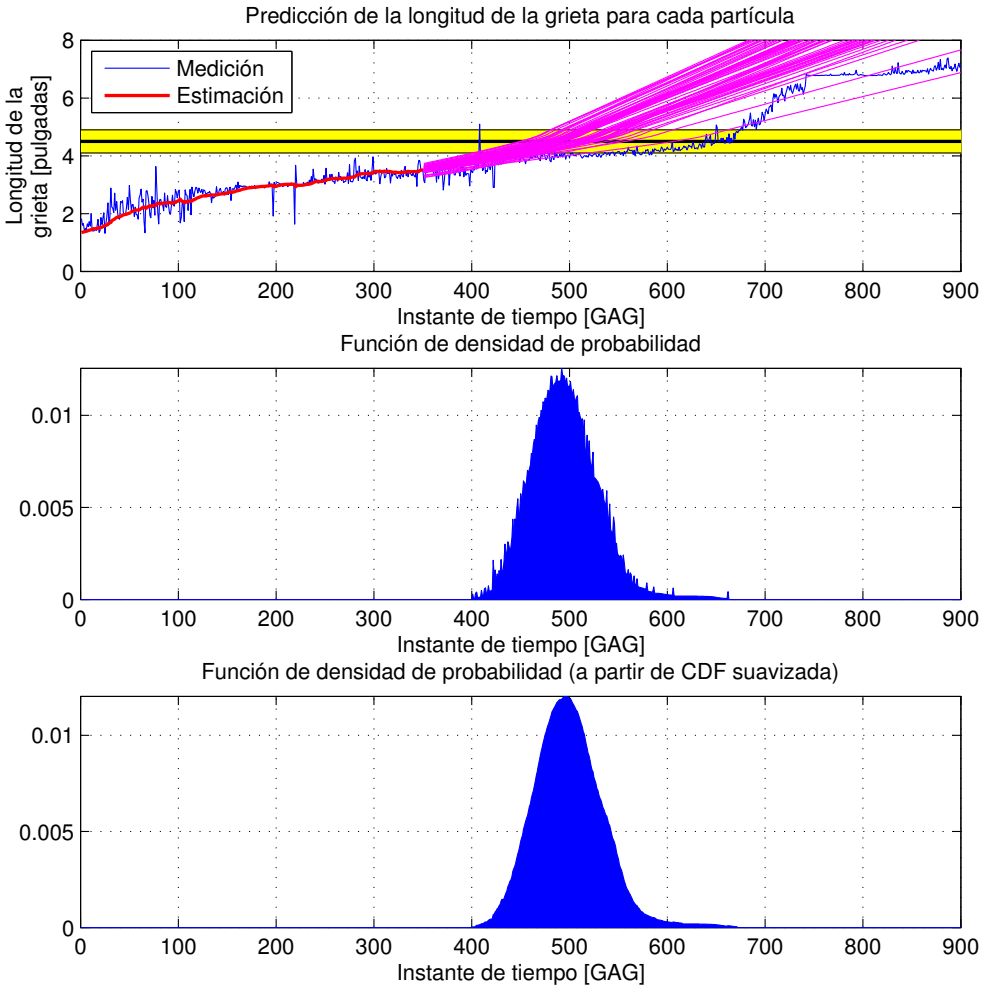


Figura 5.7: *En la imagen superior: la evolución de cada una de las partículas (y en amarillo el intervalo que cubre la distribución normal en torno al umbral de falla). En la imagen central: PDF de la distribución de predicción de falla ("real"). En la imagen inferior, la PDF antes mencionada, suavizada*

5.1.4. Predicción de la trayectoria futura del estado

Para completar la presentación de información, en la Figura 5.8 se puede apreciar la realización de una prueba que considera la utilización de 60 partículas, un horizonte de estimación de 350 [GAG] y que tiene en cuenta el cálculo del valor esperado de los estados durante la etapa de pronóstico.

En el gráfico también se traza un intervalo de confianza de 95 % y se despliega la evidencia empírica, es decir, la longitud real de la fractura en algunos instantes de tiempo según la información procedente de la Tabla 3.1.

Nótese que en el GAG 400 la longitud real de la fractura todavía está dentro del intervalo de

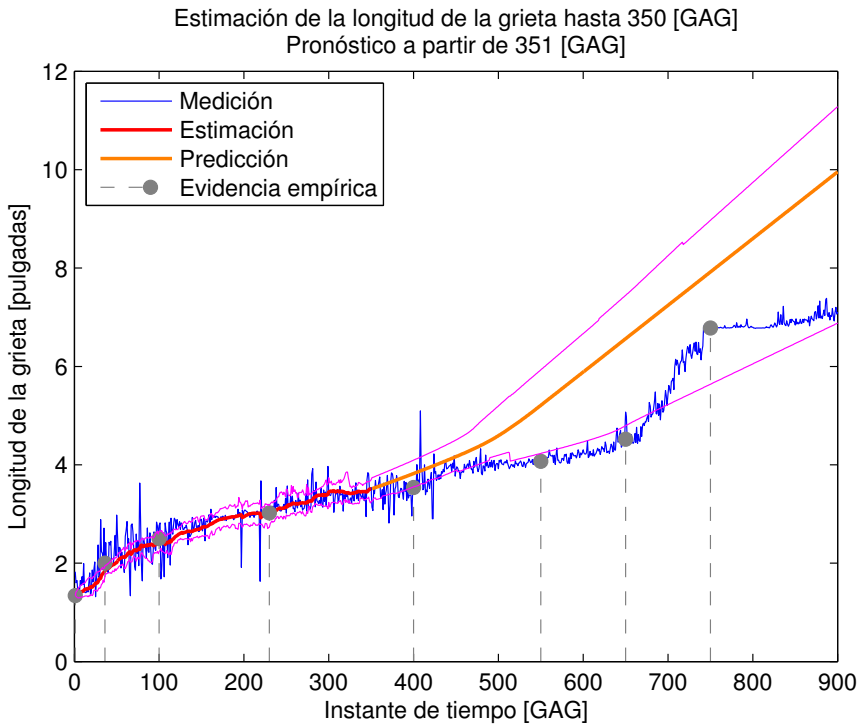


Figura 5.8: *Evidencia empírica, medición, estimación, pronóstico y su intervalo de confianza (95 %). Filtro de partículas clásico con* 60 *partículas y cálculo de valor esperado en la etapa de pronóstico*

confianza, pero a partir de dicho instante de tiempo esto deja de ser así. No obstante, dado que a partir del GAG 700 la longitud estimada de la fractura (la señal de entrada, la cual no se utiliza en esta etapa) crece con gran rapidez, la evidencia empírica vuelve a quedar dentro del intervalo de confianza.

5.2. Filtro de partículas sensible al riesgo (RSPF)

5.2.1. Filtraje de la señal

A pesar que en el caso del filtro de partículas sensible al riesgo se hace uso de una función de densidad de importancia distinta a $p(x_k|x_{0:k-1})$, la etapa de estimación da origen a resultados visualmente muy similares a los obtenidos para el caso del filtro clásico. Esto es lo que se observa en la Figura 5.9, estimación que no difiere mayormente de lo expuesto en la Figura 5.1.

Así, el desempeño del filtro RSPF implementado, en cuanto a resultados, no presenta desventajas con respecto al filtro SIR. Sin embargo, en términos de rendimiento, su utilización requiere una cantidad de cálculo mucho mayor a la que demanda el filtro clásico. Es importante tener dicho fenómeno en mente.

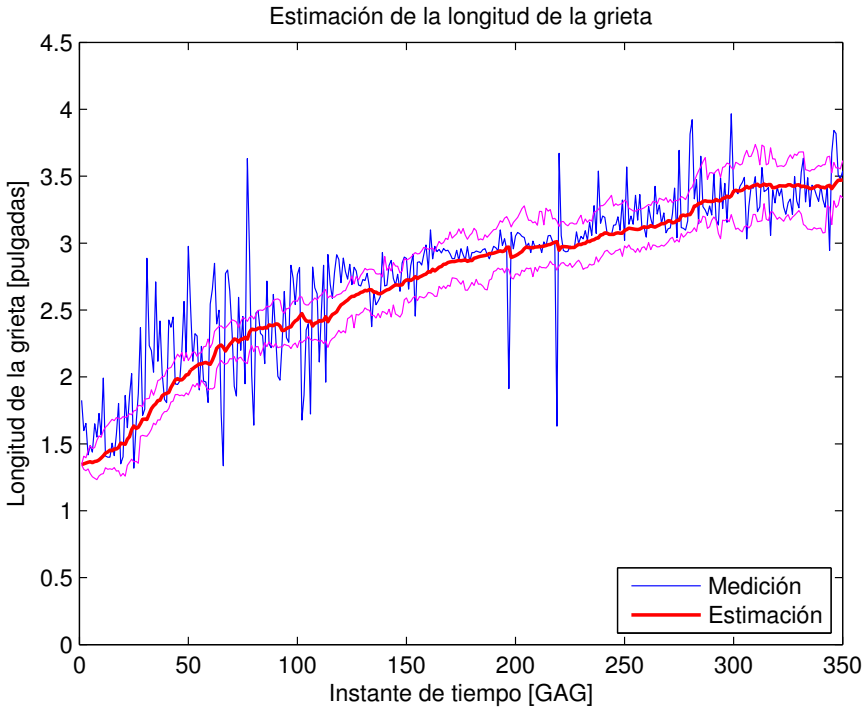


Figura 5.9: *Medición, predicción y su intervalo de confianza (95 %)*

5.2.2. Parámetro α

Como ya se expuso, la estimación de la longitud de la fractura no difiere mucho de la obtenida con el filtro SIR. Sin embargo, en este caso la evolución del parámetro α sí se diferencia fuertemente.

En la Figura 5.10 se presenta cómo evoluciona el parámetro α para tres pruebas diferentes. En la implementación que se realizó del filtro de partículas sensible al riesgo siempre se mantiene constante el valor de α durante la etapa de predicción, razón por la cual se observa el cese de variaciones en las señales a partir de 350 [GAG].

En cuanto a la dinámica exhibida por el parámetro α durante la etapa de estimación, la diferencia con lo obtenido para el filtro SIR es evidente. En este caso, α se aleja prácticamente desde un comienzo de su condición inicial, pudiendo llegar a variar casi hasta en un 30 %. Este valor supera ampliamente al 13 % de variación que se observó en el caso del filtro clásico.

Otro aspecto que se desprende de la Figura 5.10 es la inconsistencia presente en la evolución temporal de α para distintas realizaciones de una prueba: si en un caso aumenta su valor desde la condición inicial, en el siguiente puede tender a hacer exactamente lo contrario.

Que α no oscile en torno a 1, lo que se espera en el caso ideal, es perfectamente comprensible para el filtro RSPF implementado. Esto, dado que la suma de *Gaussianas* influye directamente sobre dicho parámetro para, justamente, evitar que se mantenga en torno a aquel valor. Esto se realiza buscando que el filtro reaccione de mejor forma al cambio en las condiciones del experimento.

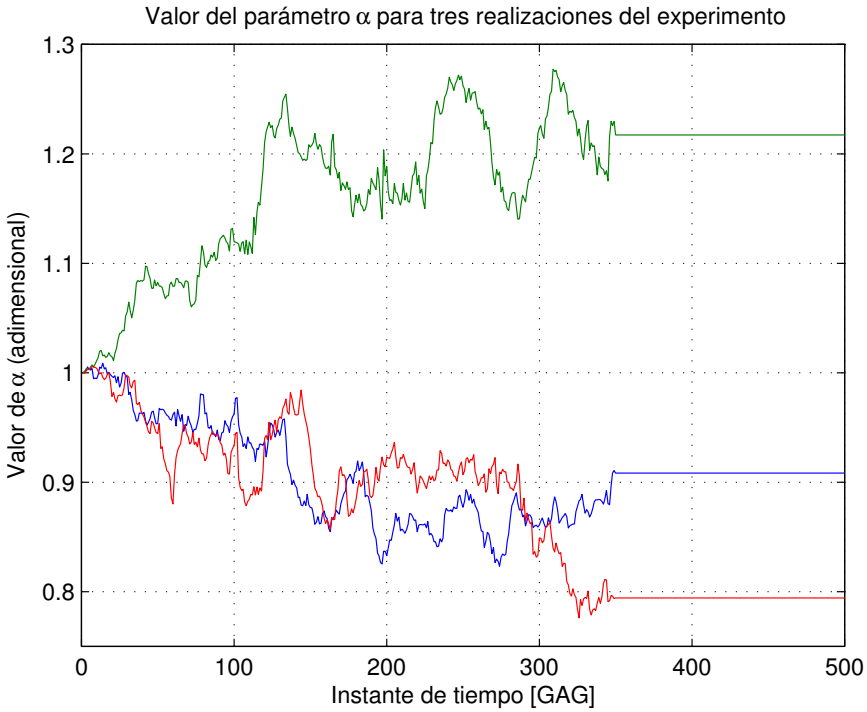


Figura 5.10: *Evolución del parámetro α en un horizonte de tiempo de 700 GAG*

5.2.3. Pronóstico de falla catastrófica

Sin considerar el cálculo del valor esperado de la trayectoria para predicción a largo plazo

En este caso, el valor de L se ve afectado por ruido *Gaussiano*, mientras que α se mantiene constante.

Dado que con la suma de *Gaussianas* se busca controlar el crecimiento de α, eventualmente su valor al finalizar la etapa de estimación puede ser bajo. Como dicho valor se mantiene constante durante toda la etapa de pronóstico y, además, como un bajo valor de α provoca que el término $C\alpha(k)[\Delta K_1(k)^m + \Delta K_2(k)^m]$ pierda influencia al momento de actualizar el valor de L, se puede dar el caso presente en la Figura 5.11.

En ella, el crecimiento de la CDF es bastante lento, llegando a 1 después de los 800 [GAG]. Esto se traduce en que la alarma se declara mucho después que en los casos expuestos anteriormente, acercándose considerablemente al momento en que realmente dicha alarma realmente debiera dispararse.

Como una forma de corroborar lo que la CDF sugiere se puede apreciar la información disponible en la Figura 5.12. En ella, la *pdf* de predicción de falla se encuentra desplazada hacia los valores de GAG altos, reafirmando lo recién expuesto.

En la Figura 5.12 nótese también cómo la trayectoria de las partículas responde al ruido *Gaus-*

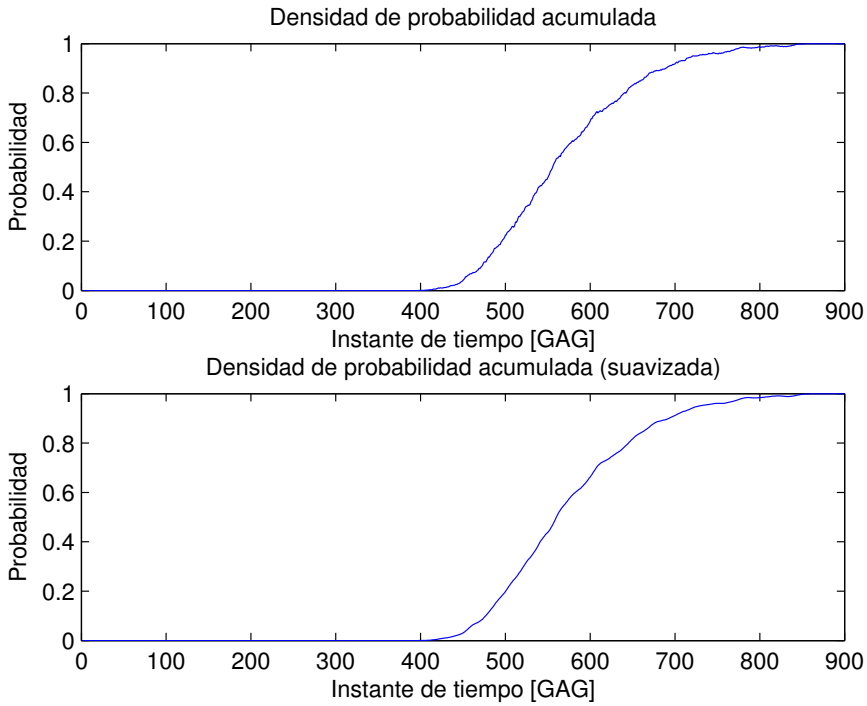


Figura 5.11: *CDF de la distribución de predicción de falla ("real" y suavizada)*

siano, pudiendo aumentar o disminuir su valor.

Considerando el cálculo del valor esperado de la trayectoria para predicción a largo plazo

En esta ocasión, al calcular el valor esperado, el ruido de proceso se hace cero. Esto provoca que α se mantenga constante y que L no pueda disminuir su valor.

En el caso que α haya terminado la etapa de estimación con un valor relativamente alto puede provocar que se sobrestime la velocidad de crecimiento de la fractura durante la etapa de pronóstico. Si además se suma el hecho que L ya no disminuirá de valor a causa del ruido Gaussiano (es cero en este caso), la estimación de la RUL puede verse afectada seriamente, declarando la alarma mucho tiempo antes de que sea realmente necesario hacerlo.

Para comprobar lo que ocurre en el caso recién descrito, se presentan las Figuras 5.13 y 5.14. Nótese cómo la CDF alcanza el valor 1 en el GAG 600 cuando en realidad, según la Tabla 3.1, la falla debería declararse recién a los 650 [GAG].

Por su parte, en la Figura 5.14 se puede ver claramente cómo los valores de las partículas sólo pueden aumentar, provocando que todas ellas superen la zona de riesgo demasiado pronto. Como la alarma se declara mucho antes de que se dé el fenómeno descrito anteriormente, el desempeño de este filtro, para este caso en particular, es bastante pobre.

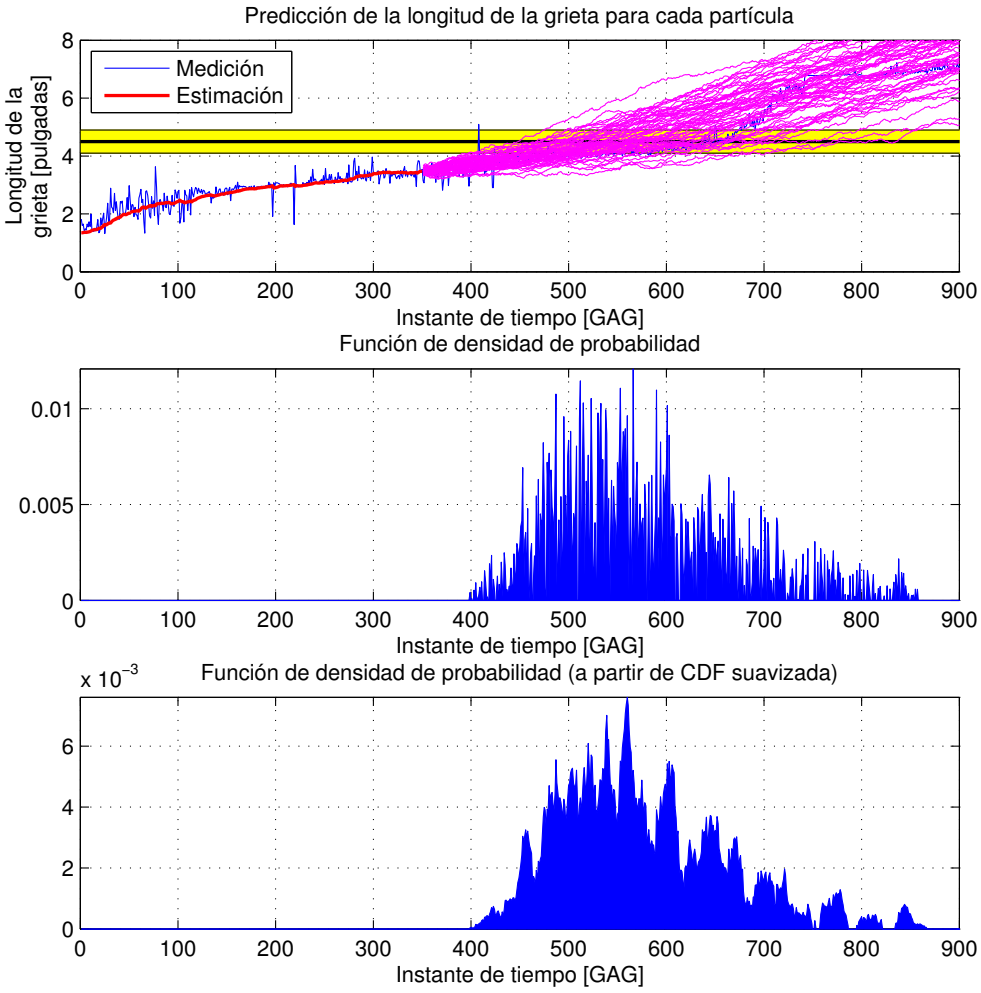


Figura 5.12: *En la imagen superior: la evolución de cada una de las partículas (y en amarillo el intervalo que cubre la distribución normal en torno al umbral de falla). En la imagen central: PDF de la distribución de predicción de falla ("real"). En la imagen inferior, la PDF antes mencionada, suavizada*

5.2.4. Predicción de la trayectoria futura del estado

Al igual que en el caso del filtro clásico, en la Figura 5.15 se presenta la realización de una prueba que considera la utilización de 60 partículas, un horizonte de estimación de 350 [GAG] y que también tiene en cuenta el cálculo del valor esperado de los estados durante la etapa de pronóstico.

En relación a la Figura 5.15, comparándola con la Figura 5.8, los resultados son bastante similares. Si bien en este punto no se puede asegurar que este comportamiento sea generalizable al desempeño de ambos filtros, en caso que así lo fuera, se recomendaría la utilización del filtro SIR dado que es poco intensivo en cuanto a cálculos computacionales.

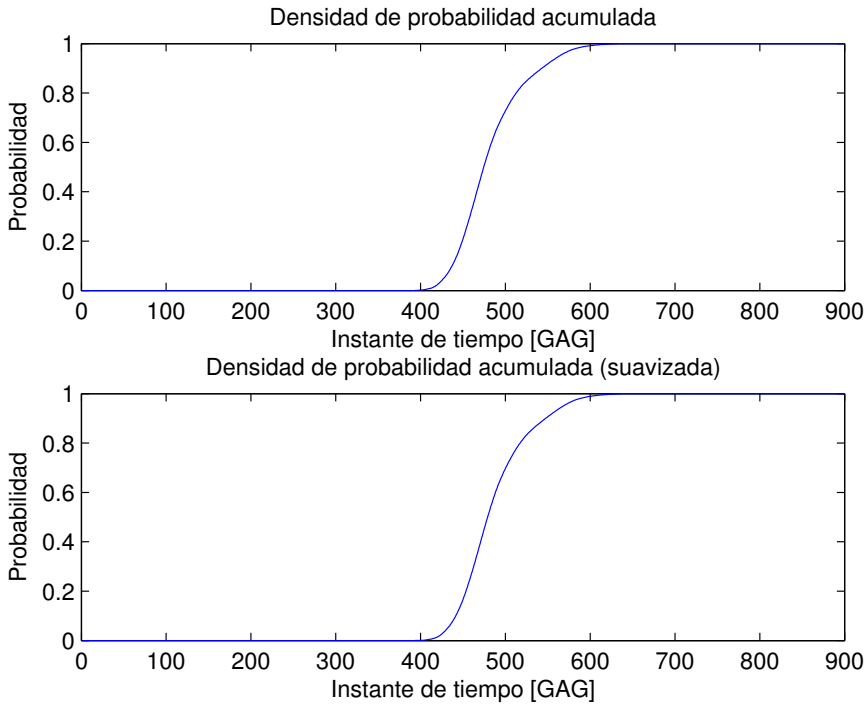


Figura 5.13: *CDF de la distribución de predicción de falla ("real" y suavizada)*

5.3. Resultados numéricos de los filtros de partículas

Los resultados que aquí se exponen provienen de experimentos que constan de cuatro pruebas básicas, una de ellas relacionada con el filtro de partículas clásico y las otras tres con el filtro de partículas sensible al riesgo.

En cada realización de un experimento se especifican cuatro parámetros comunes a todos los filtros:

- M: la cantidad de veces que se ejecutará cada una de las pruebas.
- σ_f: desviación estándar de la distribución de probabilidad *Gaussiana* de falla en torno a un umbral crítico predeterminado.
- N: número de partículas a utilizar.
- L: horizonte de estimación (último instante de tiempo antes de comenzar el pronóstico).

Con los parámetros mencionados ya es posible poner en funcionamiento el filtro de partículas clásico, pero no así para el protocolo de pruebas relacionado con el filtro de partículas sensible al riesgo.

Para poder hacer uso del filtro RSPF falta especificar la distancia d entre las *Gaussianas* a emplear. Este grado de libertad en la función de densidad de importancia q es el que se utiliza como

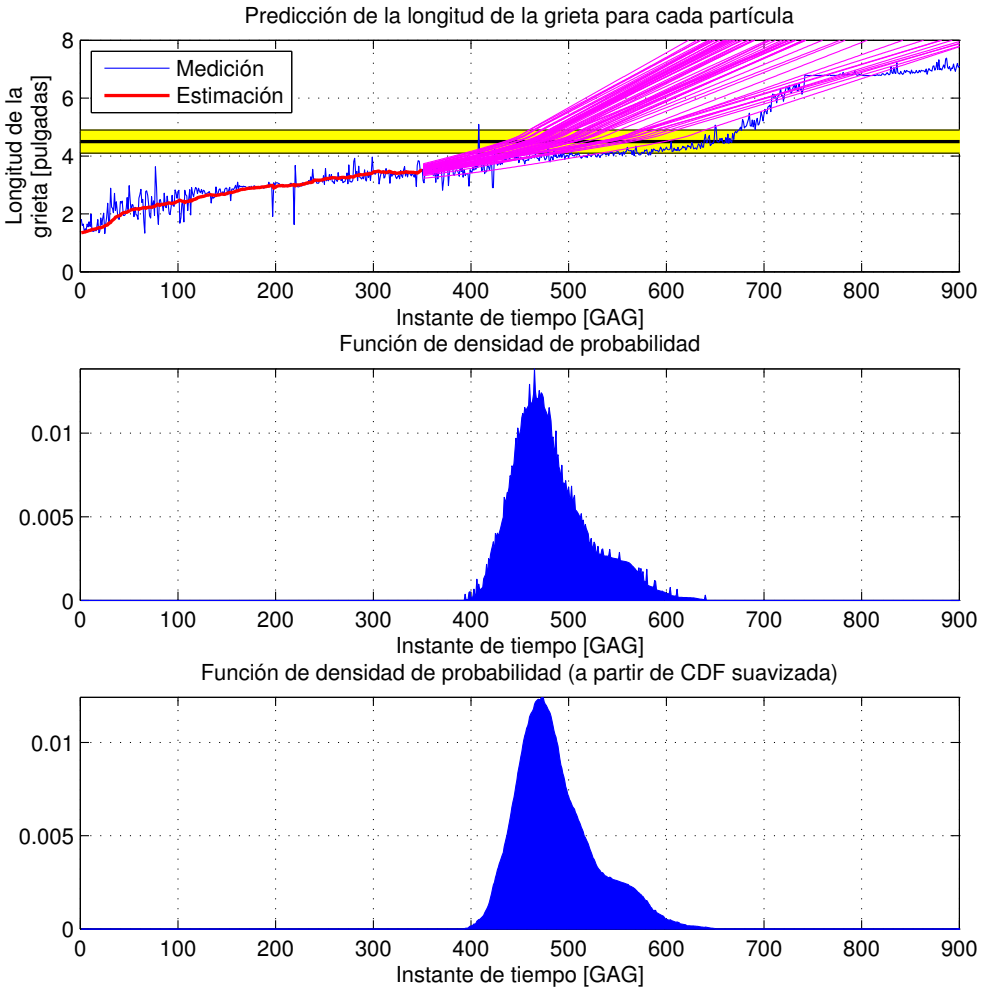


Figura 5.14: *En la imagen superior: la evolución de cada una de las partículas (y en amarillo el intervalo que cubre la distribución normal en torno al umbral de falla). En la imagen central: PDF de la distribución de predicción de falla ("real"). En la imagen inferior, la PDF antes mencionada, suavizada*

quinto parámetro en los experimentos y es el que da origen a las tres pruebas a las que se sometió el filtro sensible al riesgo.

La variación en los parámetros ya descritos se traduce en la realización de 16 pruebas diferentes cuyos resultados numéricos pueden ser consultados al final del documento, en las tablas de datos presentes en el Anexo A.

En cada una de las pruebas se realizaron $M = 6$ repeticiones a partir de las cuales se calculó un promedio simple como estimador de los valores obtenidos. Los valores originales, antes de promediarse, se obtienen en base a los dos indicadores presentados en la Sección 4.8: el cálculo del tiempo de falla por medio del valor esperado de las partículas y el cálculo del valor JIT.

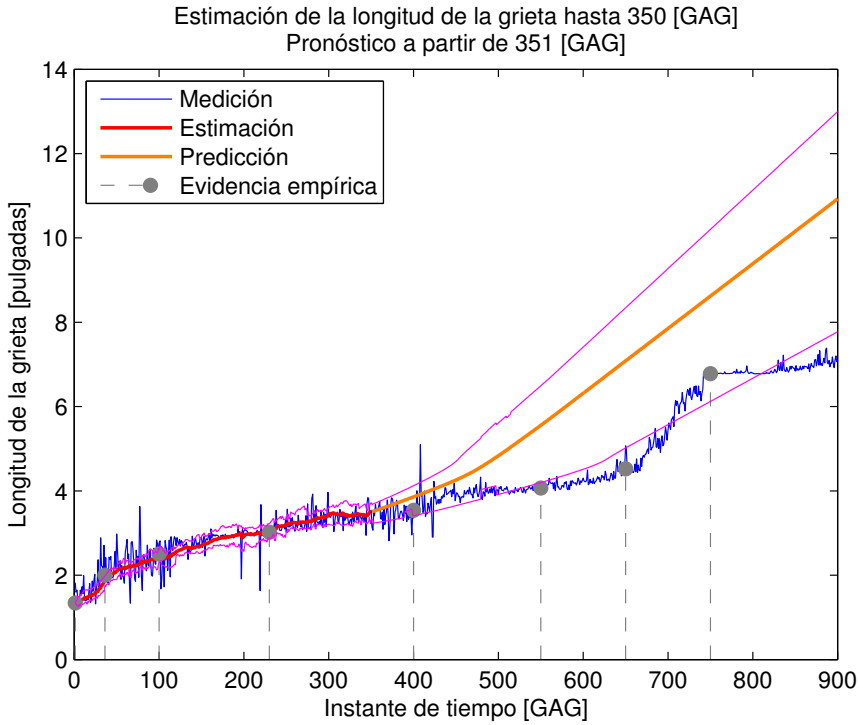


Figura 5.15: *Evidencia empírica, medición, estimación, pronóstico y su intervalo de confianza (95 %). Filtro de partículas sensible al riesgo con* 60 *partículas y cálculo de valor esperado en la etapa de pronóstico*

5.3.1. Parámetros comunes a todos los experimentos

Existen algunos parámetros importantes cuyos valores son idénticos para todos los experimentos realizados. Ellos se presentan en la Tabla 5.1.

Valor	Descripción
900	GAG hasta el que se realizará la predicción
0.85	Umbral para realizar remuestreo
0.95	Intervalo de confianza
4.5	Umbral de falla (pulgadas)
1.344	Longitud inicial de la fractura (pulgadas)
1	Condición inicial de α (adimensional)
0.9	Peso de la CDF *Gaussiana* con $\mu = 0$ en q
0.1	Peso de la CDF *Gaussiana* con $\mu = d$ en q

Tabla 5.1: *Parámetros comunes a todos los experimentos*

5.3.2. Parámetros de ruido

- Ruido de observación $\nu \sim \mathcal{N}(0, 0.15)$

- Ruido de proceso $p \sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{bmatrix} 0.0005 & 0 \\ 0 & 0.00015 \end{bmatrix}\right)$
- Ruido de proceso $q = \left[0.9 \cdot \mathcal{N}(0, 5 \cdot 10^{-4}) + 0.1 \cdot \mathcal{N}(0, 1.5 \cdot 10^{-4})\right] \cdot \mathcal{N}(0, 1.5 \cdot 10^{-4})$

La variable d podrá tomar los valores $d = -2\sigma_{\omega_2}$, $d = -3\sigma_{\omega_2}$ y $d = -4\sigma_{\omega_2}$, con $\sigma_{\omega_2} = \sqrt{1.5 \cdot 10^{-4}}$.

5.3.3. Otros parámetros

El horizonte de tiempo L hasta el que se hace la estimación y a partir del cual comienza la predicción puede tomar los valores 350 o 400 dependiendo de la prueba. A su vez, el número de partículas N utilizadas puede ser de 60 o 120, también dependiendo de la prueba que se esté realizando.

5.3.4. Datos numéricos obtenidos a partir de los experimentos

Casi la totalidad de los datos experimentales obtenidos se encuentran en el Anexo A, pero se prefirió destacar algunos situándolos en este lugar del documento.

Así, la información que se deseó privilegiar dice relación con las Figuras 5.8 y 5.15 por su relevancia a la hora de entender los análisis posteriores. La información en cuestión se presenta en las Tablas 5.2 y 5.3.

Longitud medida de la fractura		Intervalo de confianza		
GAG	Longitud de la fractura (pulgadas)	−95 %	Media	+95 %
1	1.3400	1.3431	1.3440	1.3450
36	2.0000	1.7347	1.8431	1.9452
100	2.5000	2.2369	2.4379	2.6491
230	3.0200	2.7264	3.0012	3.1737
400	3.5400	3.5488	3.8215	4.0938
550	4.0700	4.2313	5.2092	5.9299
650	4.5200	4.7946	6.5624	7.4446
750	6.7800	5.6374	7.9196	8.9697

Tabla 5.2: *Datos numéricos correspondientes a la Figura 5.8: filtro de partículas clásico con* 60 *partículas y cálculo de valor esperado en la etapa de pronóstico. Horizonte de estimación de 350 [GAG]*

En relación a la información presente en el Anexo A, el despliegue de los datos en las tablas denota por columnas el tiempo (GAG) en el que se cumple con los indicadores descritos en la Sección 4.8. Por su parte, las filas de las tablas representan el filtro con el que se experimentó. Así, el valor de d especifica la distancia existente entre las *Gaussianas* para el filtro de partículas sensible al riesgo. Nótese que cuando no se especifica un valor para el parámetro d es porque los datos de aquella fila corresponden a un filtro de partículas clásico.

Longitud medida de la fractura		Intervalo de confianza		
GAG	Longitud de la fractura (pulgadas)	−95 %	Media	+95 %
1	1.3400	1.3431	1.3440	1.3449
36	2.0000	1.7483	1.8849	1.9827
100	2.5000	2.3010	2.4384	2.6022
230	3.0200	2.8618	3.0339	3.2277
400	3.5400	3.4113	3.8578	4.1230
550	4.0700	4.1977	5.5549	6.4935
650	4.5200	5.0186	7.0835	8.3410
750	6.7800	6.1211	8.6198	10.2023

Tabla 5.3: *Datos numéricos correspondientes a la Figura 5.15: filtro de partículas sensible al riesgo con* 60 *partículas y cálculo de valor esperado en la etapa de pronóstico. Horizonte de estimación de 350 [GAG]*

Cada valor al interior de las tablas corresponde al promedio de 6 datos y está en unidades de GAG. Se optó por no aproximarlos. En caso de necesitar valores enteros se sugiere simplemente truncar los números y utilizar sólo la parte entera.

5.3.5. Análisis de los datos

Desempeño de los filtros de partículas implementados para horizontes de predicción extensos

La información presente en las Tablas 5.2 y 5.3 es crítica para entender los datos del Anexo A, y su tratamiento en lo que resta de documento, de modo que es necesario hacer un análisis previo de ella.

Cuando se presentaron las Figuras 5.8 y 5.15 se indicó que a partir del GAG 400, por una cantidad considerable de tiempo, el intervalo de confianza asociado con la predicción de la longitud de la fractura no contenía la información provista por la evidencia empírica. Ello se puede verificar numéricamente gracias a la información presente en las Tablas 5.2 y 5.3.

Nótese en ambas tablas cómo para los GAG 550 y 650, la longitud real de la fractura queda fuera del intervalo de confianza de 95 % para la predicción calculada por los filtros. Dado que que es torno a esos GAG que se debiese declarar la alarma (al menos para el caso en que se ella se dispara por medio del cálculo del valor esperado de la longitud de la fractura), se debe mirar con cierto recelo la información que entregan los filtros para horizontes de predicción demasiado extensos.

Que la longitud real de la fractura en el instante de tiempo 750 [GAG] sí se encuentre dentro del intervalo de confianza del pronóstico otorgado por los filtros no se debe a que ellos mejoren su desempeño, sino a que a partir del GAG 650 el crecimiento de la fractura se acelera. Eventualmente ese aumento logra hacer que los datos ingresen al intervalo de confianza de la predicción, pero ello se debe más que nada a una coincidencia.

Dado que no se implementó ninguna estrategia encaminada a intentar reducir la incertidumbre

del pronóstico calculado por los filtros para un horizonte de estimación extenso, lo recién discutido era perfectamente esperable.

No obstante lo descrito, es posible que en algunas ocasiones las predicciones realizadas por los filtros para horizontes de tiempo extensos sean correctas. Esto, debido a que como se verificó en la Figura 5.2 y sobre todo en la Figura 5.10, ninguno de los filtros exhibe un funcionamiento realmente consistente: la aleatoriedad de los ruidos asociados con el modelo es suficiente para dar origen a resultados bastante distintos.

Comparación del desempeño de los filtros de partículas

Estudiando los datos presentes en el Anexo A se observa que, dependiendo del valor que se le asigne al parámetro d, el desempeño del filtro sensible al riesgo puede ser en un caso superior, pero en otro caso inferior al del filtro clásico. Lo anterior quiere decir que no se puede asegurar que uno de los filtros sea superior a los demás en todos los casos, sino que los resultados dependerán del problema en estudio y de los parámetros que se le asignen a cada filtro.

Para poder visualizar de mejor manera la información presente en las tablas de las Secciones A.1, A.2, A.3 y A.4 del Anexo A, sus datos han sido graficados y expuestos en las Figuras 5.16, 5.17, 5.18 y 5.19, respectivamente.

A partir de todas las figuras antes mencionadas se desprende que, para un filtro en particular, siempre la alerta de falla se declarará con mayor prontitud para el caso JIT = 5 %, dejando el caso JIT = 10 % en segundo lugar. Y a la luz de las pruebas realizadas, el cálculo del valor esperado de la longitud de la fractura siempre declarará la alarma en último lugar.

Por ejemplo, nótese en la Figura 5.16 el caso para $d = -2\sigma_{\omega_2}$ y $\sigma_f = 0.1$: cuando se calcula la RUL para JIT = 5 %, este filtro declarará una alarma en torno al GAG 440. Si se realiza el mismo cálculo para JIT = 10 %, la alarma aparecerá alrededor del GAG 450. Finalmente, si la estimación de vida útil remanente se realiza con el valor esperado de la longitud de la fractura, la alarma no se declarará antes del GAG 540. Lo mismo ocurre para todos los filtros analizados.

En relación a lo anterior, si se recuerdan las Figuras 5.3, 5.6, 5.11 y 5.13 (gráficos de distintas funciones de distribución), que el indicador de falla para $\mathbb{E}(\text{fractura}) = 4.5$ sea el último en dispararse tiene una justificación evidente: debiera utilizarse un valor JIT muy superior al 10 % para que los tiempos de alarma fuesen similares a los que surgen del cálculo de la longitud pronosticada. Cualquier valor JIT pequeño hará que la alarma se dispare antes de que se pronostique que la longitud de la fractura es de 4.5".

En cuanto a las Figuras 5.16 y 5.17 resulta interesante observar cómo el filtro clásico supera en desempeño a todos los otros filtros considerados en casi todas las oportunidades. Sin embargo, lo anterior ya no es verdad para los datos de las Figuras 5.18 y 5.19.

Otra conclusión que se puede extraer de las Figuras 5.16 y 5.17 es que cuando el valor de L aumenta de 350 a 400, el GAG para el cual se disparan las alarmas de falla catastrófica aumenta de manera considerable, acercándose a 650 [GAG] (el valor deseado según la Tabla 3.1).

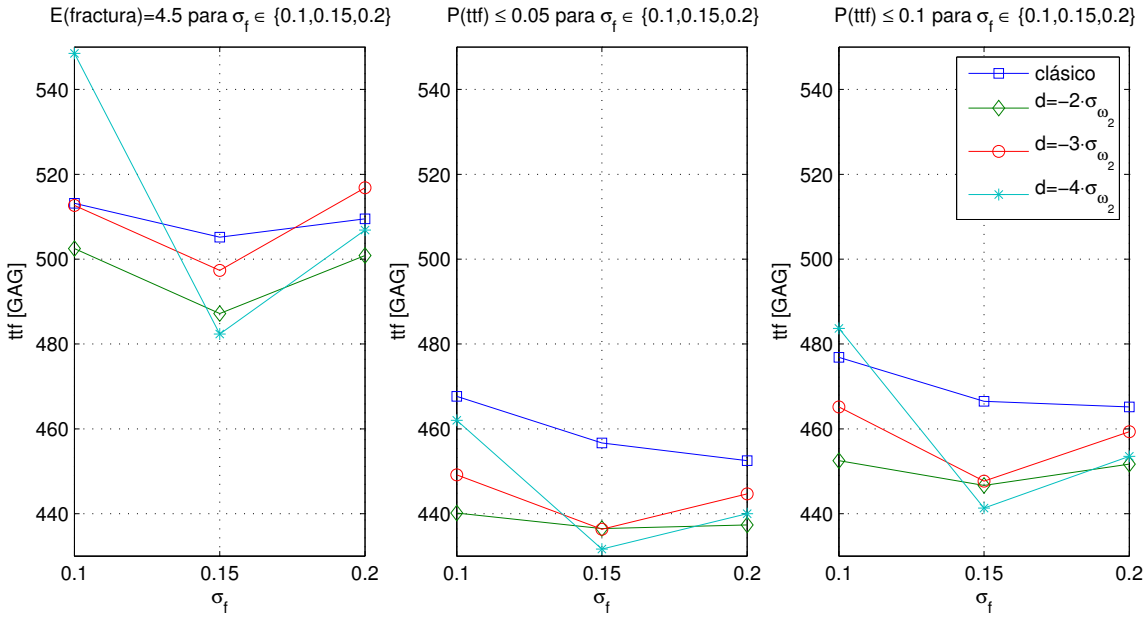


Figura 5.16: *Comparación del tiempo que demoran los filtros de partículas clásico y sensible al riesgo implementados en lanzar la alerta de falla en función de σ_f para distintos d. Parámetros: $N = 60$, $L = 350$, sin cálculo de valor esperado en el pronóstico. Longitud real de la fractura igual a 4.52" en el GAG 650*

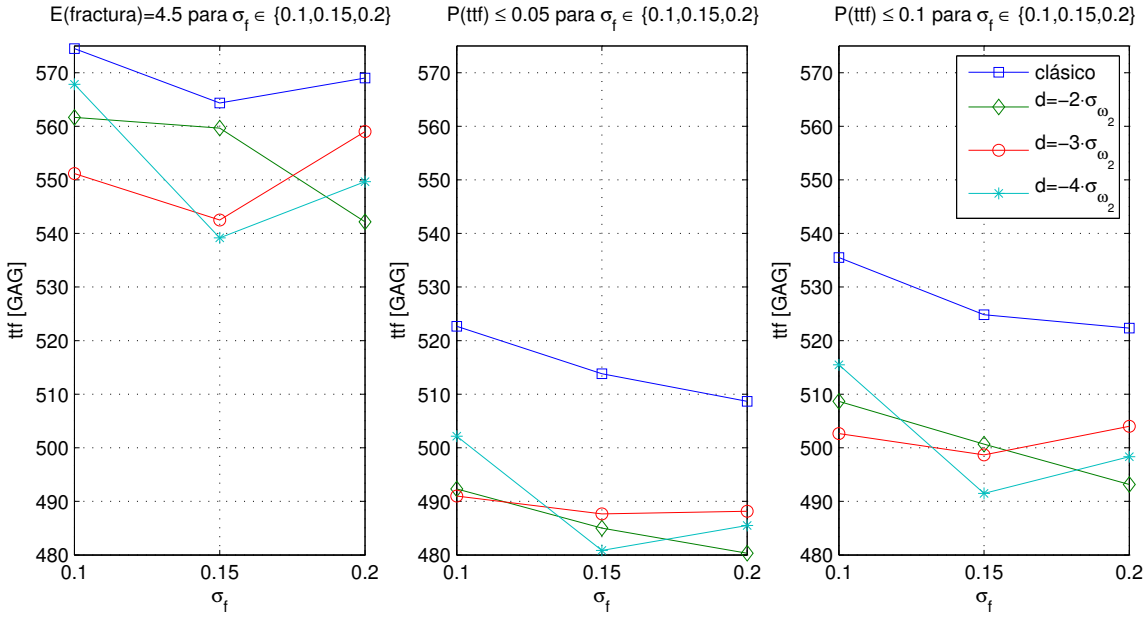


Figura 5.17: *Comparación del tiempo que demoran los filtros de partículas clásico y sensible al riesgo implementados en lanzar la alerta de falla en función de σ_f para distintos d. Parámetros: $N = 60$, $L = 400$, sin cálculo de valor esperado en el pronóstico. Longitud real de la fractura igual a 4.52" en el GAG 650*

Lo anterior se explica por el hecho que entre el GAG 320, que es a partir del cual las condiciones en las que se tomaron las mediciones variaron, y el GAG 350 hay apenas 30 unidades de tiempo

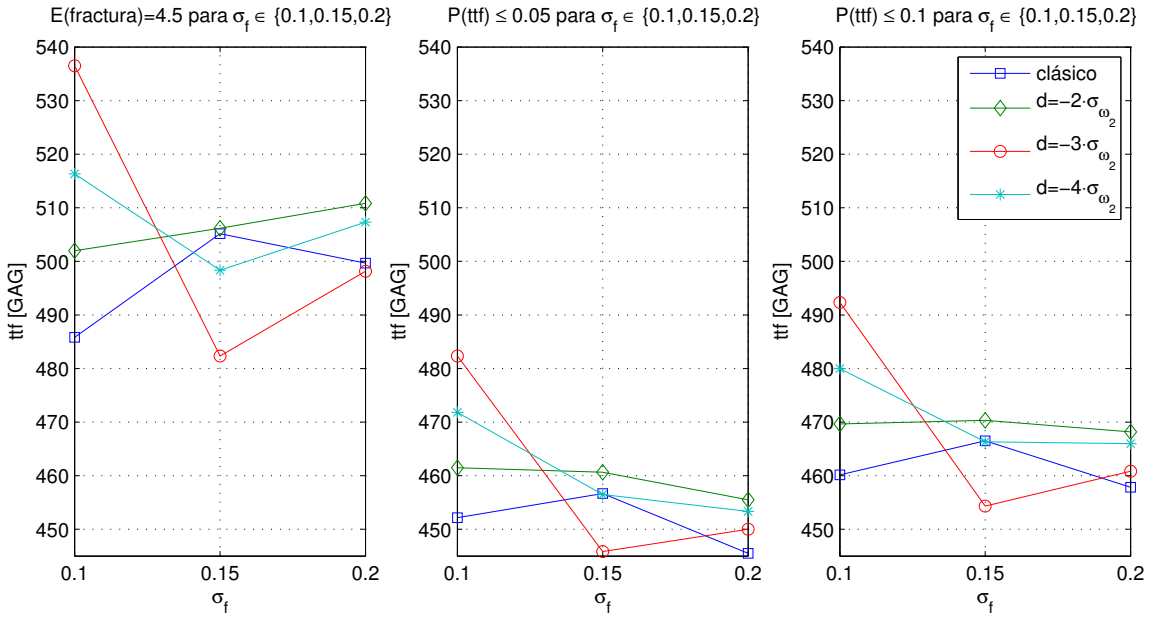


Figura 5.18: *Comparación del tiempo que demoran los filtros de partículas clásico y sensible al riesgo implementados en lanzar la alerta de falla en función de σ_f para distintos d. Parámetros: $N = 60$, $L = 350$, calculando el valor esperado en el pronóstico. Longitud real de la fractura igual a 4.52" en el GAG 650*

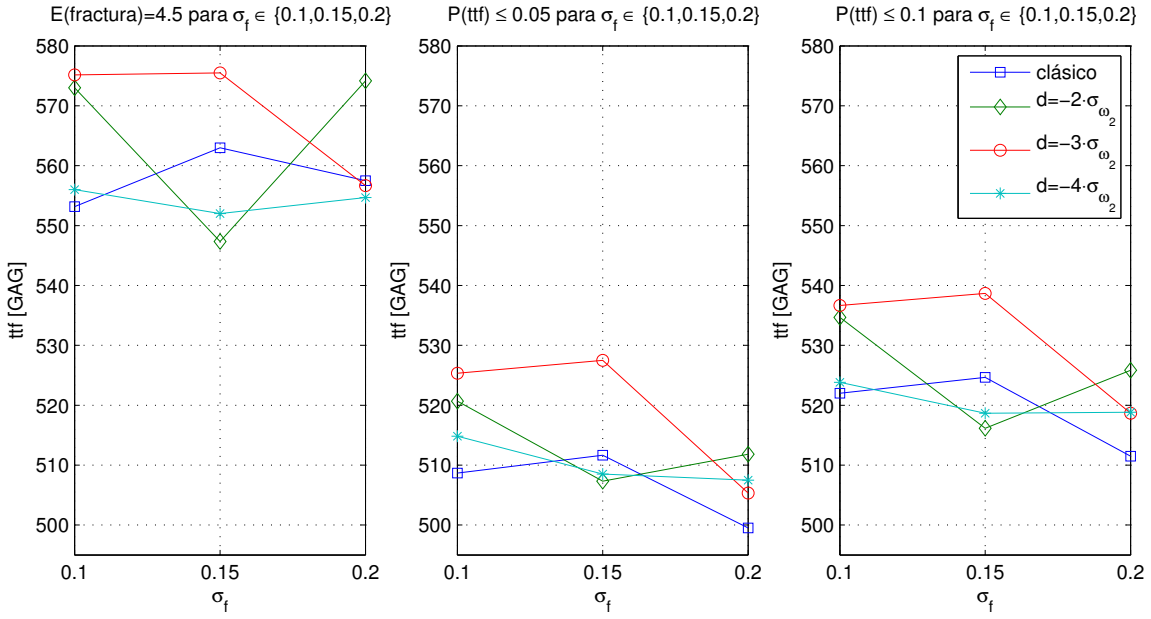


Figura 5.19: *Comparación del tiempo que demoran los filtros de partículas clásico y sensible al riesgo implementados en lanzar la alerta de falla en función de σ_f para distintos d. Parámetros: $N = 60$, $L = 400$, calculando el valor esperado en el pronóstico. Longitud real de la fractura igual a 4.52" en el GAG 650*

de diferencia. Esto provoca que los filtros no logren recabar toda la información deseable sobre el cambio para luego enfrentar la etapa de pronóstico.

Pero cuando la predicción comienza en el GAG 400 ya han transcurrido 70 instantes de tiempo, por lo que los filtros ya han sido capaces de asimilar de mejor manera la nueva información, dando origen a una predicción más acertada.

El análisis recién realizado es idéntico al considerar las Figuras 5.18 y 5.19. La única diferencia en este caso con respecto al anterior es que los filtros consideran el cálculo del valor esperado de los estados en la etapa de predicción.

En cuanto a la Figura 5.17, nótese cómo los filtros RSPF disparan la alarma de falla con extrema prontitud cuando ella es calculada mediante los valores JIT, exhibiendo un desempeño muy pobre en relación al filtro SIR. Sin embargo, para el caso en que el instante de falla se pronostica mediante el cálculo del valor estimado de la predicción, esta diferencia se acorta considerablemente.

Una forma de explicar lo anterior puede radicar en el hecho que los filtros de partículas sensibles al riesgo comienzan a aumentar su *pdf* pronosticada de falla para instantes de tiempo GAG bajos, provocando que el indicador de falla relacionado con el cálculo de los valores JIT se cumpla demasiado pronto. Este problema no se presenta cuando la alarma se dispara mediante el cálculo de $\mathbb{E}(\text{fractura}) = 4.5$.

En relación al parámetro σ_f, las pruebas realizadas no son concluyentes a la hora de explicar el desempeño de los filtros según su valor. De esta forma, por ejemplo, no se puede asegurar que a mayor valor de σ_f, menor será el valor de la RUL estimada.

Para la siguiente comparación considérense las Figuras 5.16 y 5.18. La única diferencia entre los experimentos que son representados por cada gráfico radica en si se utiliza el cálculo del valor esperado en la predicción o no.

Como una forma de facilitar la visualización de la influencia del parámetro de simulación antes mencionado, en la Figura 5.20 se presenta la resta entre los datos asociados con la Figura 5.16 (minuendo) y los que están asociados con la Figura 5.18 (sustraendo).

De la Figura 5.20 (datos asociados con $L = 350$) se desprende que, para el filtro clásico, en ningún caso se recomienda calcular el valor esperado de los estados cuando se hace la predicción. En cambio, para el filtro RSPF con $d = -2\sigma_{\omega_2}$, siempre es preferible hacerlo.

Por su parte, si sólo se consideran las alarmas de falla surgidas a partir del cálculo del valor JIT, se recomienda, en caso de utilizar un filtro RSPF, calcular el valor esperado de los estados en la etapa de predicción.

Si ahora se vuelve a realizar una resta, pero con los datos de la Figura 5.17 (minuendo) y de la Figura 5.19 (sustraendo), se obtienen los gráficos de la Figura 5.21 (datos asociados con $L = 400$). En este caso, se mantienen las recomendaciones realizadas anteriormente para los filtros SIR y RSPF.

Por su parte, en cuanto a si se recomienda comenzar la predicción en $L = 350$ o $L = 400$, los resultados expuestos en las Figuras 5.22 y 5.23 (obtenidos de la misma forma anterior, considerando los datos asociados con $L = 350$ como minuendo y los relacionados con $L = 400$ como sustraendo) son concluyentes para todos los casos estudiados: siempre se recomienda iniciar la predicción en $L = 400$.

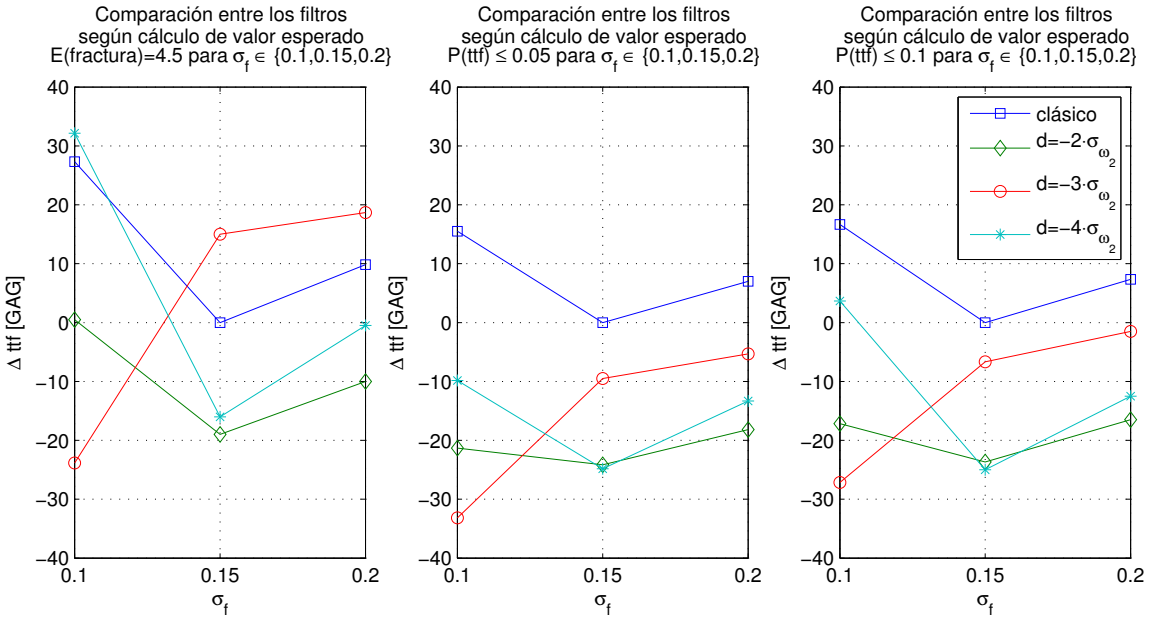


Figura 5.20: *Diferencia en el tiempo que demoran los filtros implementados en lanzar una alerta de falla de acuerdo a si se calcula el valor esperado de los estados en la etapa de predicción o no.* $\Delta ttf < 0$ *indica que el segundo tipo de filtros toman más tiempo en hacerlo. Parámetros:* $N = 60$, $L = 350$

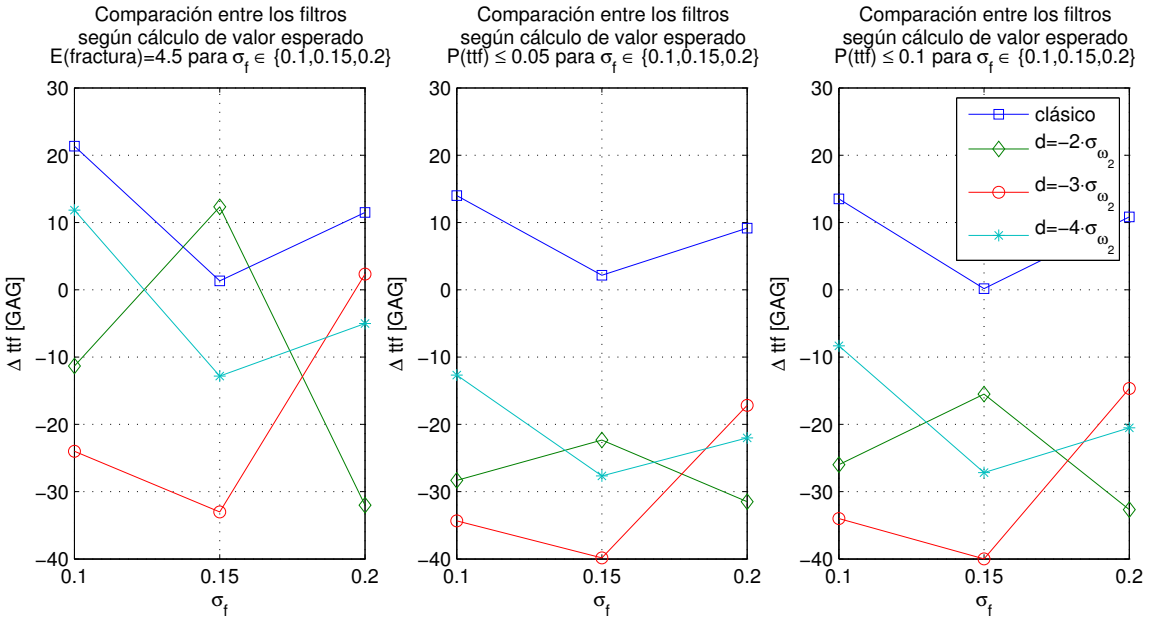


Figura 5.21: *Diferencia en el tiempo que demoran los filtros implementados en lanzar una alerta de falla de acuerdo a si se calcula el valor esperado de los estados en la etapa de predicción o no.* $\Delta ttf < 0$ *indica que el segundo tipo de filtros toman más tiempo en hacerlo. Parámetros:* $N = 60$, $L = 400$

Para finalizar, en la Figura 5.24 se presenta una comparación del desempeño de los filtros de partículas con $\sigma_f = 0.1$ y $L \in \{350, 400\}$ para los casos en que se utilizan 60 partículas (minuendo)

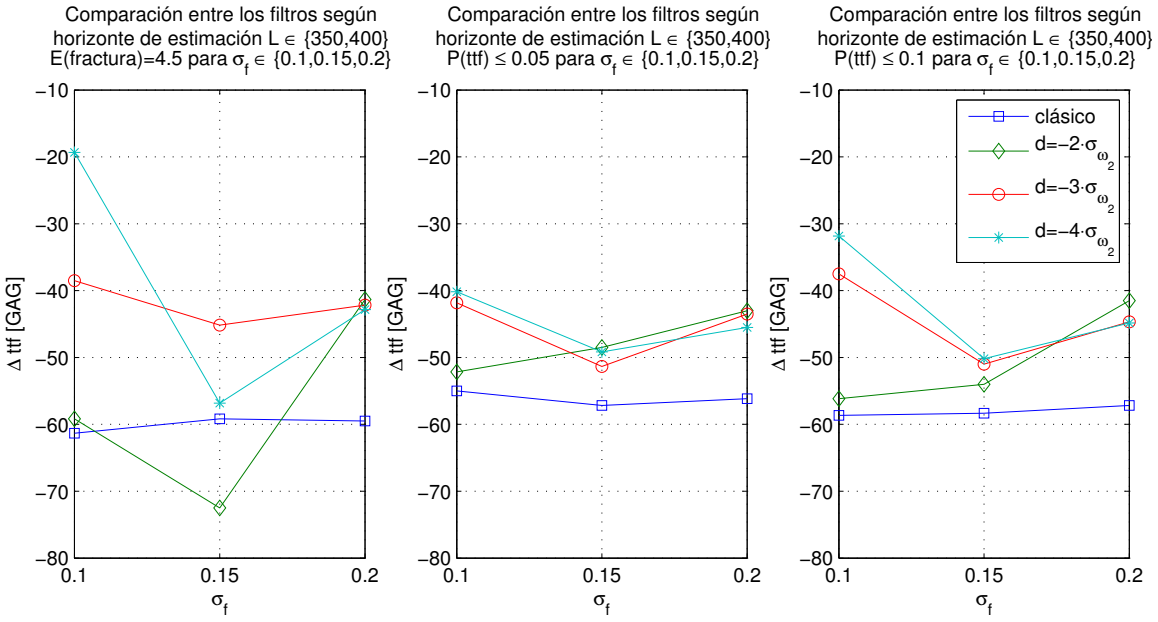


Figura 5.22: *Diferencia en el tiempo que demoran los filtros implementados en lanzar una alerta de falla de acuerdo a la longitud de la etapa de estimación.* $\Delta ttf < 0$ *indica que los filtros con* L *mayor toman más tiempo en hacerlo. Parámetros:* $N = 60$, $L \in \{350, 400\}$, *sin cálculo de valor esperado en el pronóstico*

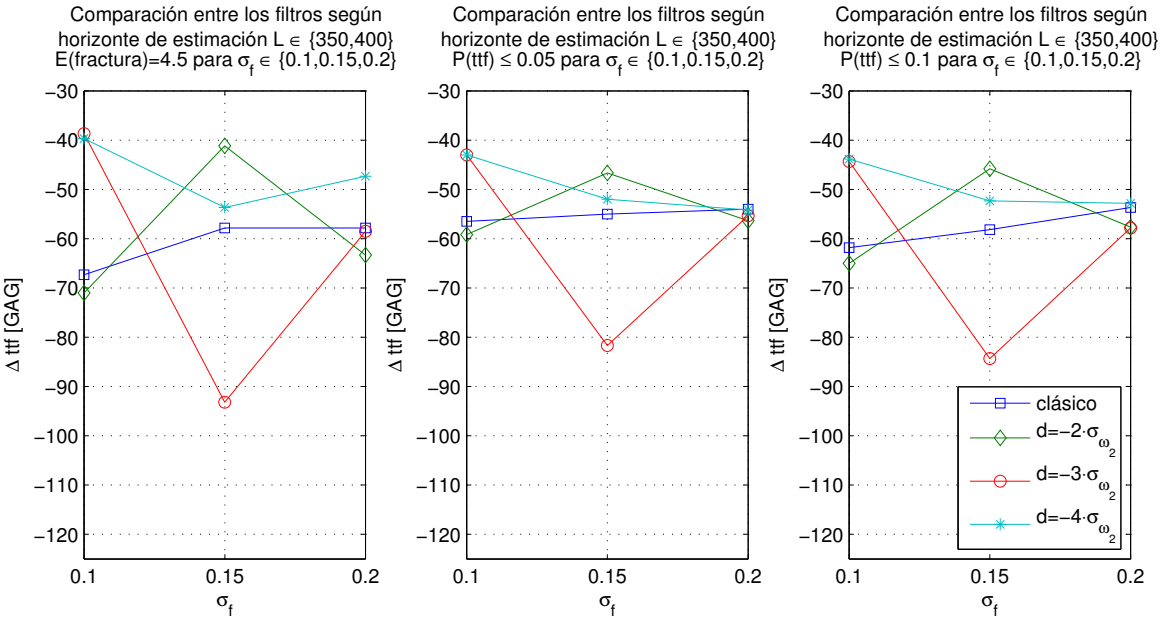


Figura 5.23: *Diferencia en el tiempo que demoran los filtros implementados en lanzar una alerta de falla de acuerdo a la longitud de la etapa de estimación.* $\Delta ttf < 0$ *indica que los filtros con* L *mayor toman más tiempo en hacerlo. Parámetros:* $N = 60$, $L \in \{350, 400\}$, *calculando el valor esperado en el pronóstico*

y 120 partículas (sustraendo). Los datos numéricos provienen de las tablas de las Secciones A.5 y A.6 del Anexo A.

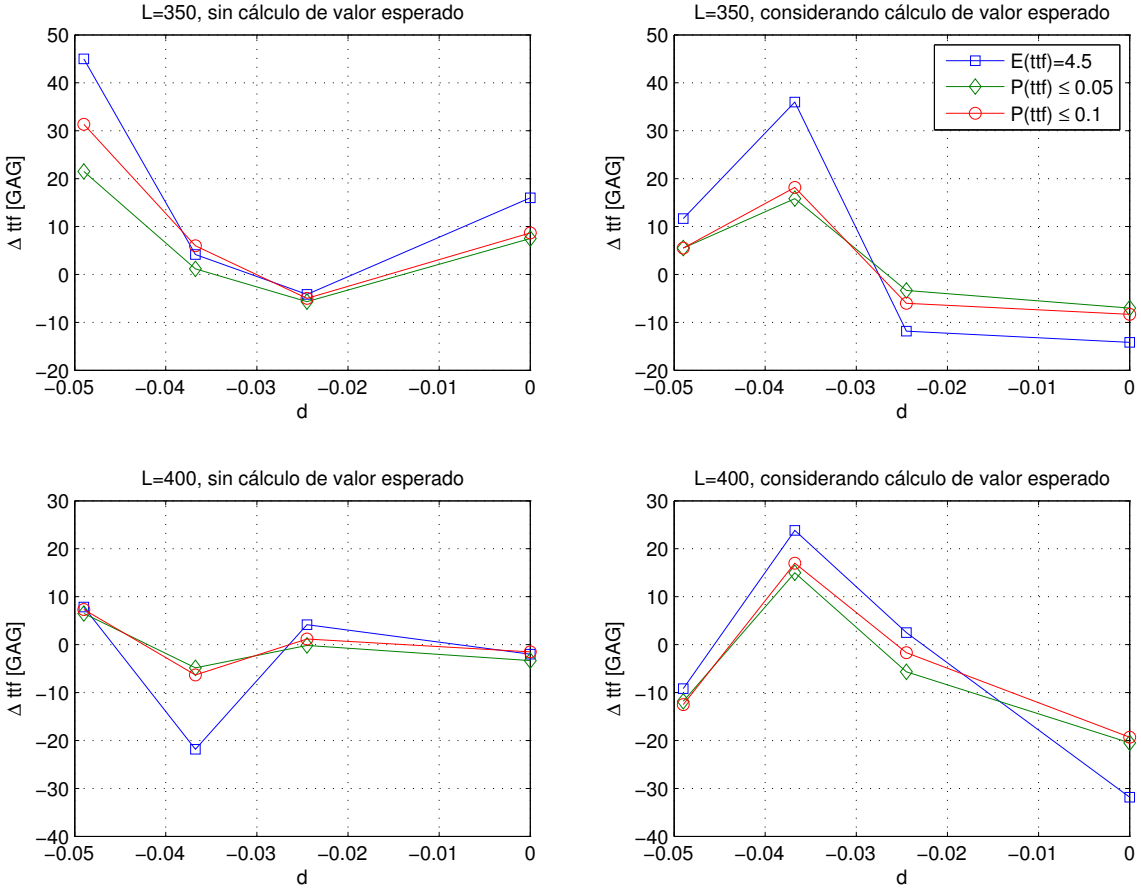


Figura 5.24: *Comparación de filtros según número de partículas. Los gráficos permiten visualizar la diferencia entre pares de datos cuando el minuendo está asociado al filtro de 60 partículas y el sustraendo al de 120*

Notar que, como se considera σ_f constante, en la Figura 5.24 el eje de las abscisas representa la distancia entre las *Gaussianas* de los filtros RSPF. Para efectos de la figura se consideró que el filtro clásico utiliza $d = 0$.

A partir de la Figura 5.24 se desprende que, cuando no se calcula el valor esperado de los estados en la etapa de predicción, por lo general será recomendable utilizar filtros con 60 partículas solamente.

Por su parte, cuando sí se calcula el valor esperado, podría ser recomendable utilizar filtros con 120 partículas, pero dado que la mejora no es muy significativa, es preferible descartar dicha opción para reducir la cantidad de cálculos.

Comentarios finales

En base a los datos obtenidos, el filtro de partículas sensible al riesgo implementado logró ser una alternativa viable al filtro clásico, en cuanto a resultados, sólo en determinadas ocasiones.

No obstante lo anterior, en caso de ser necesario escoger uno de los filtros examinados, se recomienda utilizar el filtro clásico en todos los casos. El criterio en el que se basa dicha recomendación radica en la cantidad de cálculos que requiere hacer un filtro RSPF para realizar la estimación de la RUL, cantidad que es muy superior a la del filtro SIR implementado.

Como una forma de hacer explícito lo comentado en el párrafo anterior, considérense los siguiente: el algoritmo del filtro SIR demora alrededor de 5 segundos en ejecutarse, mientras que una simulación bajo las misma condiciones, pero para un filtro RSPF, puede tomar 5 minutos.

Capítulo 6

Trabajo futuro y conclusiones

6.1. Conclusiones

A la luz de los resultados, no se pudo cumplir con el objetivo de estimar correctamente la vida útil remanente de la placa de engranaje planetario. Esto se traduce en que no se pudo pronosticar con exactitud cuánto tiempo más podía ser utilizado el helicóptero de manera segura.

La comparación de los resultados de las pruebas con la evidencia empírica (Tabla 3.1) no fue satisfactoria para ninguno de los filtros ni condiciones consideradas, de modo que se intuye que durante el trabajo se hicieron algunas simplificaciones que podrían haber resultado demasiado fuertes.

En particular, el hecho de mantener el peso de las partículas constante para todo el proceso de pronóstico pudo haber sido lo más relevante.

Otras causas de error podrían encontrarse en los parámetros utilizados en las distribuciones que modelan el ruido de proceso y observación. Esto, pues se verificó mediante las simulaciones que las variaciones en el ruido pueden dar origen a estimaciones de vida útil remanente muy distintas.

En cuanto a la demostración de las diferencias que se introducen al utilizar un filtro de partículas clásico y uno sensible al riesgo, los resultados fueron satisfactorios.

A partir del trabajo desarrollado se pudo analizar numéricamente la relevancia que tiene la manera en que se escoge la función de densidad de importancia q para el problema analizado. Y en base a los resultados, si se debe escoger entre los filtros implementados, se sugiere utilizar el filtro de partículas clásico puesto que, comparado con los filtros de partículas sensibles al riesgo, es mucho menos intensivo en cuanto a cálculo computacional y da origen a estimaciones de vida útil remanente de similar calidad.

En relación a los resultados obtenidos, un aspecto sobre el que se debe reflexionar es el del marco de pruebas utilizado. No se descarta que al considerar otras posibilidades se pueda obtener información de una calidad mayor a la expuesta en el presente documento.

La apreciación anterior dice relación, tanto con considerar otros indicadores de falla, como con explorar una cantidad más extensa de variaciones entre los grados de libertad que ofrece el problema

en estudio.

Finalmente, se deduce que los inconvenientes encontrados al momento de resolver el problema de detección de falla están en íntima relación con las simplificaciones introducidas en la implementación de los filtros.

No obstante lo anterior tampoco se descarta que los parámetros de ruido se puedan mejorar o que el modelo se pueda perfeccionar, pero dado que dichos elementos ya fueron utilizados satisfactoriamente en [10], su influencia en los resultados obtenidos en este problema particular se consideran de menor relevancia.

6.2. Trabajo futuro

Lo fundamental es lograr mejorar la predicción de falla obtenida en este trabajo y, con ello, la estimación de vida útil remanente del sistema. Para lograr lo anterior se sugiere reducir las simplificaciones realizadas comenzando por la forma en que se propagan las partículas al momento de la predicción. Mantener los pesos constantes no se presenta como una buena alternativa pues aparentemente resulta ser un supuesto demasiado fuerte.

Otro avance que se puede realizar es utilizar otro tipo de distribuciones para modelar el ruido inherente al problema, ya sea de proceso u observación. En particular, resulta interesante analizar si se puede reemplazar la distribución que modela el ruido de proceso, una normal, por otra distinta. Por ejemplo, se puede estudiar el efecto que tiene utilizar una distribución *Gamma* para modelar el ruido del primer estado, con el fin de imponer que el largo estimado de la fractura no puede disminuir a lo largo del tiempo (siguiendo la idea que una fractura no puede reducir su tamaño).

En la misma línea anterior, podría ser interesante buscar otras funciones q para el filtro de partículas sensible al riesgo, o bien seguir utilizando la suma de *Gaussianas*, pero con otros parámetros.

En cuanto al algoritmo en sí, la introducción de estrategias destinadas a la evaluación de la calidad de la predicción realizada resulta una alternativa necesaria a la luz de los resultados obtenidos. Esta alternativa necesariamente debe ser explorada si se sigue el camino de actualizar los pesos durante la etapa de predicción.

Otra alternativa que podría estudiarse corresponde a la implementación de los algoritmos considerando una predicción a más de un paso. A priori esta alternativa no resulta demasiado atractiva en el contexto de los resultados obtenidos en el presente trabajo, pero si se logra mejorar la predicción de la falla, puede resultar útil indagar en esta dirección.

Finalmente, un avance que se puede realizar, pero que es específico al problema estudiado en este trabajo, es el de generar un nuevo modelo (parámetros) para la planta, distinto al presentado en el Código 4.1. Si se lograra desarrollar un modelo más exacto, pero que no sacrifique tanto la simplicidad, podría obtenerse una mejor estimación y posterior pronóstico de falla. Esto, debido a que tanto la estimación como el pronóstico realizado en el presente estudio dependen fuertemente del modelo matemático utilizado de modo que un avance en ese sentido podría ser interesante para

el estudio del problema de la placa de engranaje planetario del helicóptero.

Bibliografía

[1] C. Andrieu, A. Doucet, and E. Punskaya. *Sequential Monte Carlo Methods in Practice*, chapter Sequential Monte Carlo Methods for Optimal Filtering. Springer-Verlag, 2001. 9, 10

[2] M. Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, February 2002. 5, 6, 7, 12, 13, 14, 15

[3] A. Doucet. On sequential monte carlo methods for bayesian filtering. Technical report, Department of Engineering, University of Cambridge, UK, 1998. 9, 12

[4] A. Doucet, S. Godsill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10(3):197–208, July 2000. 9

[5] A. J. Haug. A tutorial on bayesian estimation and tracking techniques applicable to nonlinear and non-gaussian processes. Technical report, The MITRE Corporation, 2005. 4

[6] Y. C. Ho and R. C. K. Lee. A bayesian approach to problems in stochastic estimation and control. *IEEE Transactions on Automatic Control*, AC-9:333–339, 1964. 5

[7] K. Kanazawa, D. Koller, and S. J. Russell. Stochastic simulation algorithms for dynamic probabilistic networks. *Proceedings of the 11th Annual Conference on Uncertainty in Artificial Intelligence*, pages 346–351, 1995. 8

[8] Canh Ly, Kwok Tom, Carl S. Byington, Romano Patrick, and George J. Vachtsevanos. Fault diagnosis and failure prognosis for engineering systems: A global perspective. *5th Annual IEEE Conference on Automation Science and Engineering*, pages 108–115, August 2009. 16, 17

[9] J. MacCormick and A. Blake. A probabilistic exclusion principle for tracking multiple objects. *Proceedings of the 7th International Conference on Computer Vision*, pages 572–578, 1999. 8

[10] Marcos E. Orchard. *A Particle Filtering-Based Framework for On-Line Fault Diagnosis and Failure Prognosis*. PhD thesis, Georgia Institute of Technology, 2007. VI, 4, 7, 8, 9, 10, 11, 12, 15, 16, 17, 18, 19, 21, 22, 23, 24, 25, 26, 30, 65

[11] Paul C. Paris and F. Erdogan. A critical analysis of crack propagation laws. *Journal of Basic Engineering*, 85(4):528–534, 1963. 26

[12] Romano Patrick-Aldaco. *A Model Based Framework for Fault Diagnosis and Prognosis of Dynamical Systems with an Application to Helicopter Transmissions*. PhD thesis, Georgia Institute of Technology, 2007. VI, 21, 22

[13] Marc Strass. Transmission crack grounds two-thirds of army black hawk fleet. Defense Daily, http://findarticles.com/p/articles/mi_6712/is_25_214/ai_n28912940/, May 2002. 22

[14] Sebastian Thrun, John Langford, and Vandi Verma. Risk sensitive particle filters. *Proceedings of Neural Information Processing Systems (NIPS)*, December 2001. 15, 16

[15] George Vachtsevanos, Frank L. Lewis, Michael Roemer, Andrew Hess, and Biqing Wu. *Intelligent Fault Diagnosis and Prognosis for Engineering Systems.* John Wiley & Sons, Inc., 2006. 1

[16] V. Verma, G. Gordon, R. Simmonds, and S. Thrun. Particle filters for rover fault diagnosis. *IEEE Robotics and Automation Magazine*, pages 56–64, June 2004. 15

[17] Eric W. Weisstein. Bivariate normal distribution. MathWorld–A Wolfram Web Resource, http://mathworld.wolfram.com/BivariateNormalDistribution.html. 33

[18] Eric W. Weisstein. Delta function. MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/DeltaFunction.html. 7

[19] Eric W. Weisstein. Erf. MathWorld–A Wolfram Web Resource, http://mathworld.wolfram.com/Erf.html. 34

[20] Eric W. Weisstein. Normal distribution. MathWorld–A Wolfram Web Resource, http://mathworld.wolfram.com/NormalDistribution.html. 34

[21] Eric W. Weisstein. Normal sum distribution. MathWorld–A Wolfram Web Resource, http://mathworld.wolfram.com/NormalSumDistribution.html.

Anexos

A. Tablas de datos

Cada número incluido en las tablas proviene de un promedio de 6 valores. La nomenclatura:

- *Sin cálculo de valor esperado*: pronóstico no considera cálculo de la esperanza de los estados.
- *Con cálculo de valor esperado*: pronóstico sí considera cálculo de la esperanza de los estados.
- d: distancia entre las *Gaussianas* para el filtro de partículas sensible al riesgo.
- N: número de partículas.
- L: horizonte de estimación (último GAG antes de comenzar el pronóstico).
- σ_f: desviación estándar de la distribución de probabilidad *Gaussiana* de falla en torno a 4.5.
- σ_{ω_2}: desviación estándar de la distribución de probabilidad del error asociado con el estado x_2. Para todos los casos, $\sigma_{\omega_2} = \sqrt{1.5 \cdot 10^{-4}}$.

A.1. Sin cálculo de valor esperado. $N = 60$ y $L = 350$

d	$\mathbb{E}(\text{fractura}) = 4.5$	$\mathbb{P}(\text{fractura}) \leq 0.05$	$\mathbb{P}(\text{fractura}) \leq 0.1$
-	513.1667	467.6667	476.8333
$-2\sigma_{\omega_2}$	502.5000	440.1667	452.5000
$-3\sigma_{\omega_2}$	512.6667	449.1667	465.1667
$-4\sigma_{\omega_2}$	548.5000	462.0000	483.6667

Tabla A.1: *Sin cálculo de valor esperado.* $N = 60$, $L = 350$ *y* $\sigma_f = 0.1$

d	$\mathbb{E}(\text{fractura}) = 4.5$	$\mathbb{P}(\text{fractura}) \leq 0.05$	$\mathbb{P}(\text{fractura}) \leq 0.1$
-	505.1667	456.6667	466.5000
$-2\sigma_{\omega_2}$	487.1667	436.5000	446.6667
$-3\sigma_{\omega_2}$	497.3333	436.3333	447.6667
$-4\sigma_{\omega_2}$	482.3333	431.6667	441.3333

Tabla A.2: *Sin cálculo de valor esperado.* $N = 60$, $L = 350$ *y* $\sigma_f = 0.15$

d	$\mathbb{E}(\text{fractura}) = 4.5$	$\mathbb{P}(\text{fractura}) \leq 0.05$	$\mathbb{P}(\text{fractura}) \leq 0.1$
-	509.5000	452.5000	465.1667
$-2\sigma_{\omega_2}$	500.8333	437.3333	451.6667
$-3\sigma_{\omega_2}$	516.8333	444.6667	459.3333
$-4\sigma_{\omega_2}$	506.8333	440.0000	453.5000

Tabla A.3: *Sin cálculo de valor esperado.* $N = 60$, $L = 350$ *y* $\sigma_f = 0.2$

A.2. Sin cálculo de valor esperado. $N = 60$ y $L = 400$

d	$\mathbb{E}(\text{fractura}) = 4.5$	$\mathbb{P}(\text{fractura}) \leq 0.05$	$\mathbb{P}(\text{fractura}) \leq 0.1$
-	574.5000	522.6667	535.5000
$-2\sigma_{\omega_2}$	561.6667	492.3333	508.6667
$-3\sigma_{\omega_2}$	551.1667	491.0000	502.6667
$-4\sigma_{\omega_2}$	567.8333	502.1667	515.5000

Tabla A.4: *Sin cálculo de valor esperado.* $N = 60$, $L = 400$ *y* $\sigma_f = 0.1$

d	$\mathbb{E}(\text{fractura}) = 4.5$	$\mathbb{P}(\text{fractura}) \leq 0.05$	$\mathbb{P}(\text{fractura}) \leq 0.1$
-	564.3333	513.8333	524.8333
$-2\sigma_{\omega_2}$	559.6667	485.0000	500.6667
$-3\sigma_{\omega_2}$	542.5000	487.6667	498.6667
$-4\sigma_{\omega_2}$	539.1667	480.8333	491.5000

Tabla A.5: *Sin cálculo de valor esperado.* $N = 60$, $L = 400$ *y* $\sigma_f = 0.15$

d	$\mathbb{E}(\text{fractura}) = 4.5$	$\mathbb{P}(\text{fractura}) \leq 0.05$	$\mathbb{P}(\text{fractura}) \leq 0.1$
-	569.0000	508.6667	522.3333
$-2\sigma_{\omega_2}$	542.1667	480.3333	493.1667
$-3\sigma_{\omega_2}$	559.0000	488.1667	504.0000
$-4\sigma_{\omega_2}$	549.6667	485.5000	498.3333

Tabla A.6: *Sin cálculo de valor esperado.* $N = 60$, $L = 400$ *y* $\sigma_f = 0.2$

A.3. Con cálculo de valor esperado. $N = 60$ y $L = 350$

d	$\mathbb{E}(\text{fractura}) = 4.5$	$\mathbb{P}(\text{fractura}) \leq 0.05$	$\mathbb{P}(\text{fractura}) \leq 0.1$
-	485.8333	452.1667	460.1667
$-2\sigma_{\omega_2}$	502.0000	461.5000	469.6667
$-3\sigma_{\omega_2}$	536.5000	482.3333	492.3333
$-4\sigma_{\omega_2}$	516.3333	471.8333	480.0000

Tabla A.7: *Con cálculo de valor esperado.* $N = 60$, $L = 350$ *y* $\sigma_f = 0.1$

d	$\mathbb{E}(\text{fractura}) = 4.5$	$\mathbb{P}(\text{fractura}) \leq 0.05$	$\mathbb{P}(\text{fractura}) \leq 0.1$
-	505.1667	456.6667	466.5000
$-2\sigma_{\omega_2}$	506.1667	460.6667	470.3333
$-3\sigma_{\omega_2}$	482.3333	445.8333	454.3333
$-4\sigma_{\omega_2}$	498.3333	456.5000	466.3333

Tabla A.8: *Con cálculo de valor esperado.* $N = 60$, $L = 350$ *y* $\sigma_f = 0.15$

d	$\mathbb{E}(\text{fractura}) = 4.5$	$\mathbb{P}(\text{fractura}) \leq 0.05$	$\mathbb{P}(\text{fractura}) \leq 0.1$
-	499.6667	445.5000	457.8333
$-2\sigma_{\omega_2}$	510.8333	455.5000	468.1667
$-3\sigma_{\omega_2}$	498.1667	450.0000	460.8333
$-4\sigma_{\omega_2}$	507.3333	453.3333	466.0000

Tabla A.9: *Con cálculo de valor esperado.* $N = 60$, $L = 350$ *y* $\sigma_f = 0.2$

A.4. Con cálculo de valor esperado. $N = 60$ y $L = 400$

d	$\mathbb{E}(\text{fractura}) = 4.5$	$\mathbb{P}(\text{fractura}) \leq 0.05$	$\mathbb{P}(\text{fractura}) \leq 0.1$
-	553.1667	508.6667	522.0000
$-2\sigma_{\omega_2}$	573.0000	520.6667	534.6667
$-3\sigma_{\omega_2}$	575.1667	525.3333	536.6667
$-4\sigma_{\omega_2}$	556.0000	514.8333	523.8333

Tabla A.10: *Con cálculo de valor esperado.* $N = 60$, $L = 400$ *y* $\sigma_f = 0.1$

d	$\mathbb{E}(\text{fractura}) = 4.5$	$\mathbb{P}(\text{fractura}) \leq 0.05$	$\mathbb{P}(\text{fractura}) \leq 0.1$
-	563.0000	511.6667	524.6667
$-2\sigma_{\omega_2}$	547.3333	507.3333	516.1667
$-3\sigma_{\omega_2}$	575.5000	527.5000	538.6667
$-4\sigma_{\omega_2}$	552.0000	508.5000	518.6667

Tabla A.11: *Con cálculo de valor esperado.* $N = 60$, $L = 400$ *y* $\sigma_f = 0.15$

d	$\mathbb{E}(\text{fractura}) = 4.5$	$\mathbb{P}(\text{fractura}) \leq 0.05$	$\mathbb{P}(\text{fractura}) \leq 0.1$
-	557.5000	499.5000	511.5000
$-2\sigma_{\omega_2}$	574.1667	511.8333	525.8333
$-3\sigma_{\omega_2}$	556.6667	505.3333	518.6667
$-4\sigma_{\omega_2}$	554.6667	507.5000	518.8333

Tabla A.12: *Con cálculo de valor esperado.* $N = 60$, $L = 400$ *y* $\sigma_f = 0.2$

A.5. Sin cálculo de valor esperado. $N = 200$ y $\sigma_f = 1$

d	$\mathbb{E}(\text{fractura}) = 4.5$	$\mathbb{P}(\text{fractura}) \leq 0.05$	$\mathbb{P}(\text{fractura}) \leq 0.1$
-	497.1667	460.1667	468.1667
$-2\sigma_{\omega_2}$	506.6667	445.8333	457.5000
$-3\sigma_{\omega_2}$	508.5000	448.0000	459.1667
$-4\sigma_{\omega_2}$	503.5000	440.5000	452.3333

Tabla A.13: *Sin cálculo de valor esperado.* $N = 200$, $L = 350$ *y* $\sigma_f = 0.1$

d	$\mathbb{E}(\text{fractura}) = 4.5$	$\mathbb{P}(\text{fractura}) \leq 0.05$	$\mathbb{P}(\text{fractura}) \leq 0.1$
-	576.5000	526.0000	537.0000
$-2\sigma_{\omega_2}$	557.5000	492.5000	507.5000
$-3\sigma_{\omega_2}$	573.0000	495.8333	509.0000
$-4\sigma_{\omega_2}$	560.0000	495.6667	508.1667

Tabla A.14: *Sin cálculo de valor esperado.* $N = 200$, $L = 400$ *y* $\sigma_f = 0.1$

A.6. Con cálculo de valor esperado. $N = 200$ y $\sigma_f = 1$

d	$\mathbb{E}(\text{fractura}) = 4.5$	$\mathbb{P}(\text{fractura}) \leq 0.05$	$\mathbb{P}(\text{fractura}) \leq 0.1$
-	500.0000	459.1667	468.5000
$-2\sigma_{\omega_2}$	513.8333	464.8333	475.6667
$-3\sigma_{\omega_2}$	500.5000	466.5000	474.1667
$-4\sigma_{\omega_2}$	504.6667	466.3333	474.5000

Tabla A.15: *Con cálculo de valor esperado.* $N = 200$, $L = 350$ *y* $\sigma_f = 0.1$

d	$\mathbb{E}(\text{fractura}) = 4.5$	$\mathbb{P}(\text{fractura}) \leq 0.05$	$\mathbb{P}(\text{fractura}) \leq 0.1$
-	585.0000	529.1667	541.3333
$-2\sigma_{\omega_2}$	570.5000	526.3333	536.3333
$-3\sigma_{\omega_2}$	551.3333	510.3333	519.6667
$-4\sigma_{\omega_2}$	565.1667	526.6667	536.3333

Tabla A.16: *Con cálculo de valor esperado.* $N = 200$, $L = 400$ *y* $\sigma_f = 0.1$

B. Código de los programas

B.1. MAIN.m

```
% Filtro de partículas: rutina principal
% Daniel Neira Bravo

%---Se cargan los datos medidos (observaciones)
if exist('Crack_Data.mat','file')
  load Crack_Data;
else
  disp('ERROR: No se encontró el archivo Crack_Data.mat');
  return;
end

%---Inicialización de los parámetros de ruido
%-Ruido de proceso
noise_p.tag = 'ruido de proceso (gaussiano)';
noise_p.mu  = [0; 0];
noise_p.cov = [0.0005, 0; 0, 0.00015];
%-Ruido de observación
noise_o.tag = 'ruido de observación (gaussiano)';
noise_o.mu  = 0;
noise_o.cov = 0.15;

%---Inicialización de los parámetros del modelo
model.tag       = 'parámetros asociados con el modelo';
model.param.m   = 3.5074;     % constante del material
model.param.C   = 7.7331e-10; % parámetro cuyo valor se corrige con alpha
model.L         = 350;        % tiempo hasta el que se utilizará el filtro
model.P         = 900;        % tiempo hasta el que se hará la predicción
model.noise_p   = noise_p;    % ruido de proceso
model.noise_o   = noise_o;    % ruido de observación
model.threshold = 0.85;       % parámetro para resampling: threshold*N
model.confInt   = 0.95;       % intervalo de confianza en la predicción
model.K_1       = [63.14 63.85 57.95 56.15 61.91 76.91 78.47]./1.0988;
model.K_2       = [80.95 74.78 62.08 59.55 77.63 71.08 79.49]./1.0988;
model.Lengths   = [1.5 2 2.5 3 3.5 4 4.5]./2;
model.U_93      = (0.3195^model.param.m + 0.3195^model.param.m +...
    0.7618^model.param.m + 0.7618^model.param.m + 0.7618^model.param.m +...
    0.7618^model.param.m)^(1/model.param.m);
model.U_100     = (0.3195^model.param.m + 0.3195^model.param.m +...
    0.7618^model.param.m + 0.7618^model.param.m + 0.7618^model.param.m +...
    0.7618^model.param.m + 0.5733^model.param.m)^(1/model.param.m);
model.U_120     = (0.3195^model.param.m + 0.3195^model.param.m +...
    0.7618^model.param.m + 0.7618^model.param.m + 0.7618^model.param.m +...
    0.7618^model.param.m + 0.5733^model.param.m +...
    0.5733^model.param.m)^(1/model.param.m);

%---Inicialización de los parámetros de distribución en torno a la falla
failure_i.tag       = 'distribución gaussiana en torno a un umbral';
```

```
failure_i.threshold = 4.5;
failure_i.mu        = failure_i.threshold;
failure_i.stdev     = 0.2;
failure_i.cov       = failure_i.stdev^2;
failure_i.max       = failure_i.mu+2*failure_i.stdev;
failure_i.min       = failure_i.mu-2*failure_i.stdev;
failure_i.est.prob  = zeros(model.P,1);

%---Inicialización de las partículas y sus parámetros
particles.init.crack = 1.344; % longitud inicial de la fractura
particles.init.alpha = 1;     % valor inicial de alpha
particles.N          = 60;    % número de partículas
particles.value      = zeros(2,particles.N);
particles.value(1,:) = 1e-3*(2*rand(1,particles.N)-1)+particles.init.crack;
particles.value(2,:) = 1e-5*(2*rand(1,particles.N)-1)+particles.init.alpha;
particles.weights    = ones(1,particles.N)/particles.N;

output.crack.part      = zeros(model.P,particles.N); % partículas
output.crack.part(1,:) = particles.value(1,:);       % condición inicial
output.crack.est       = zeros(model.P,1);           % estimación
output.crack.est(1,1)  = particles.init.crack;       % condición inicial
output.alpha.part      = zeros(model.L,particles.N); % partículas
output.alpha.part(1,:) = particles.value(2,:);       % condición inicial
output.alpha.est       = zeros(model.L,1);           % estimación
output.alpha.est(1,1)  = particles.init.alpha;       % condición inicial

alfa = zeros(model.P,1);

output.part.weights    = zeros(model.L,particles.N);
output.part.weights(1,:) = particles.weights;

output.interval        = zeros(model.P,2);
output.interval(1,:)   = confidence_interval(particles,...
    output.crack.est(1,1),model.confInt);

%---Estimación de la longitud de la fractura en base a las observaciones
for k = 2:model.L
  %-Actualización de los valores de las partículas y sus pesos
  if k < 321; load_u = 120; else load_u = 93; end;
  particles = update_particles(particles, model, load_u, Y(k-1),1);
  %-Resampling
  particles = resampling(particles, model.threshold);
  %-Estimación de la longitud de la fractura y de alpha
  [output.crack.est(k), output.alpha.est(k)] = output_mean(particles);
  %-Cálculo del intervalo de confianza
  boundaries = confidence_interval(particles,...
      output.crack.est(k),model.confInt);
  %-Se guardan las variables relevantes
  output.crack.part(k,:) = particles.value(1,:);
  output.alpha.part(k,:) = particles.value(2,:);
  output.part.weights(k,:) = particles.weights;
  output.interval(k,:) = [boundaries(1), boundaries(2)];
```

```
end

alfa(1:model.L,1) = output.alpha.est;

failure = zeros(1,3); % fractura = 4.5, p(fractura) <= 0.05; p(fractura) <= 0.1
%---Pronóstico de crecimiento de la longitud de fractura y tiempo de falla
for k = model.L+1:model.P
  %-Se actualiza el valor de las partículas y sus pesos se dejan constantes
  if k < 321; load_u = 120; else load_u = 93; end;
  particles = update_particles(particles, model, load_u, 0,0);
  output.crack.part(k,:) = particles.value(1,:);
  [crack, alpha] = output_mean(particles);
  if(crack <= 4.5)
      failure(1) = k+1;
  end
  %-Se estima la probabilidad de falla
  failure_i.est.prob(k) = failure_prognosis(particles,failure_i);
alfa(k,1)=alpha;
  output.crack.est(k) = crack;
  boundaries = confidence_interval(particles,crack,model.confInt);
  output.interval(k,:) = [boundaries(1), boundaries(2)];
end

%---Procesamiento de señales obtenidas
%-Suavizado de la CDF
signal_to_filter = failure_i.est.prob;
windowSize       = 10;
cdf_s            = filter_data(signal_to_filter,windowSize);
for k = model.L+1:model.P
  if(cdf_s(k) <= 0.05)
    failure(2) = k+1;
  end
  if(cdf_s(k) <= 0.1)
    failure(3) = k+1;
  else
    break;
  end
end
```

B.2. MAINrs.m

```
% Filtro de partículas sensible al riesgo: rutina principal
% Daniel Neira Bravo

%---Se cargan los datos medidos (observaciones)
if exist('Crack_Data.mat','file')
  load Crack_Data;
else
  disp('ERROR: No se encontró el archivo Crack_Data.mat');
  return;
```

```
end

%---Inicialización de los parámetros de ruido
%-Ruido de proceso (suma de gaussianas)
noise_p1.tag    = 'ruido de proceso (gaussiano)';
noise_p1.mu     = [0; 0];
noise_p1.cov    = [0.0005, 0; 0, 0.00015];
noise_p1.weight = 0.9;
noise_p2.tag    = 'ruido de proceso (gaussiano)';
noise_p2.cov    = [0.0005, 0; 0, 0.00015];
noise_p2.d      = -2*sqrt(noise_p2.cov(2,2));
noise_p2.mu     = [0; noise_p1.mu(2) + noise_p2.d];
noise_p2.weight = 0.1;
if (noise_p1.weight + noise_p2.weight) ~= 1
  disp('ERROR: La suma de los pesos de las gaussianas debe ser igual a 1');
  return;
end
%-Ruido de observación
noise_o.tag = 'ruido de observación (gaussiano)';
noise_o.mu  = 0;
noise_o.cov = 0.15;

%---Inicialización de los parámetros del modelo
model.tag       = 'parámetros asociados con el modelo';
model.param.m   = 3.5074;     % constante del material
model.param.C   = 7.7331e-10; % parámetro cuyo valor se corrige con alpha
model.pc        = 2;          % selecciona se se calcula valor esperado,"1" o no "2"
model.L         = 350;        % tiempo hasta el que se utilizará el filtro
model.P         = 900;        % tiempo hasta el que se hará la predicción
model.noise_p1  = noise_p1;   % ruido de proceso 1
model.noise_p2  = noise_p2;   % ruido de proceso 2
model.noise_o   = noise_o;    % ruido de observación
model.threshold = 0.85;       % parámetro para resampling: threshold*N
model.confInt   = 0.95;       % intervalo de confianza en la predicción
model.K_1       = [63.14 63.85 57.95 56.15 61.91 76.91 78.47]./1.0988;
model.K_2       = [80.95 74.78 62.08 59.55 77.63 71.08 79.49]./1.0988;
model.Lengths   = [1.5 2 2.5 3 3.5 4 4.5]./2;
model.U_93      = (0.3195^model.param.m + 0.3195^model.param.m +...
    0.7618^model.param.m + 0.7618^model.param.m + 0.7618^model.param.m +...
    0.7618^model.param.m)^(1/model.param.m);
model.U_100     = (0.3195^model.param.m + 0.3195^model.param.m +...
    0.7618^model.param.m + 0.7618^model.param.m + 0.7618^model.param.m +...
    0.7618^model.param.m + 0.5733^model.param.m)^(1/model.param.m);
model.U_120     = (0.3195^model.param.m + 0.3195^model.param.m +...
    0.7618^model.param.m + 0.7618^model.param.m + 0.7618^model.param.m +...
    0.7618^model.param.m + 0.5733^model.param.m +...
    0.5733^model.param.m)^(1/model.param.m);

%---Inicialización de los parámetros de distribución en torno a la falla
failure_i.tag       = 'distribución gaussiana en torno a un umbral';
failure_i.threshold = 4.5;
failure_i.mu        = failure_i.threshold;
```

```
failure_i.stdev     = 0.2;
failure_i.cov       = failure_i.stdev^2;
failure_i.max       = failure_i.mu+2*failure_i.stdev;
failure_i.min       = failure_i.mu-2*failure_i.stdev;
failure_i.est.prob  = zeros(model.P,1);%zeros(model.P-model.L,1);

%---Inicialización de las partículas y sus parámetros
particles.init.crack = 1.344; % longitud inicial de la fractura
particles.init.alpha = 1;     % valor inicial de alpha
particles.N          = 60;    % número de partículas
particles.value      = zeros(2,particles.N);
particles.value(1,:) = 1e-3*(2*rand(1,particles.N)-1)+particles.init.crack;
particles.value(2,:) = 1e-5*(2*rand(1,particles.N)-1)+...
    particles.init.alpha;
particles.weights    = ones(1,particles.N)/(particles.N);

output.crack.part      = zeros(model.P,particles.N); % partículas
output.crack.part(1,:) = particles.value(1,:);       % condición inicial
output.crack.est       = zeros(model.P,1);           % estimación
output.crack.est(1,1)  = particles.init.crack;       % condición inicial
output.alpha.part      = zeros(model.L,particles.N); % partículas
output.alpha.part(1,:) = particles.value(2,:);       % condición inicial
output.alpha.est       = zeros(model.L,1);           % estimación
output.alpha.est(1,1)  = particles.init.alpha;       % condición inicial

alfa = zeros(model.P,1);

output.part.weights    = zeros(model.L,particles.N);
output.part.weights(1,:) = particles.weights;

output.interval        = zeros(model.P,2);
output.interval(1,:)   = confidence_interval(particles,...
    output.crack.est(1,1),model.confInt);

%---Estimación de la longitud de la fractura en base a las observaciones
for k = 2:model.L
  %-Actualización de los valores de las partículas y sus pesos
  if k < 321; load_u = 120; else load_u = 93; end;
  particles = update_particlesrs(particles, model, load_u, Y(k-1),1,0);
  %-Resampling
  particles = resampling(particles, model.threshold);
  %-Estimación de la longitud de la fractura y de alpha
  [output.crack.est(k), output.alpha.est(k)] = output_mean(particles);
  %-Cálculo del intervalo de confianza
  boundaries = confidence_interval(particles,...
      output.crack.est(k),model.confInt);
  %-Se guardan las variables relevantes
  output.crack.part(k,:) = particles.value(1,:);
  output.alpha.part(k,:) = particles.value(2,:);
  output.part.weights(k,:) = particles.weights;
  output.interval(k,:) = [boundaries(1), boundaries(2)];
end
```

```
alfa(1:model.L,1) = output.alpha.est;

failure = zeros(1,3); % fractura = 4.5, p(fractura) <= 0.05; p(fractura) <= 0.1
%---Pronóstico de crecimiento de la longitud de fractura y tiempo de falla
for k = model.L+1:model.P
  %-Se actualiza el valor de las partículas y sus pesos se dejan constantes
  if k < 321; load_u = 120; else load_u = 93; end;
  particles = update_particlesrs(particles, model, load_u, 0,0,model.pc);
  output.crack.part(k,:) = particles.value(1,:);
  [crack, alpha] = output_mean(particles);
  if(crack <= 4.5)
      failure(1) = k+1;
  end
  %-Se estima la probabilidad de falla
  failure_i.est.prob(k) = failure_prognosis(particles,failure_i);
alfa(k,1)=alpha;
  output.crack.est(k) = crack;
  boundaries = confidence_interval(particles,crack,model.confInt);
  output.interval(k,:) = [boundaries(1), boundaries(2)];
end

%---Procesamiento de señales obtenidas
%-Suavizado de la CDF
signal_to_filter = failure_i.est.prob;
windowSize       = 10;
cdf_s            = filter_data(signal_to_filter,windowSize);
for k = model.L+1:model.P
  if(cdf_s(k) <= 0.05)
    failure(2) = k+1;
  end
  if(cdf_s(k) <= 0.1)
    failure(3) = k+1;
  else
    break;
  end
end
```

B.3. confidence_interval.m

```
%-Cálculo de los extremos inferior y superior del intervalo de confianza
function boundaries = confidence_interval(particles, crack_est, confInt)
  [temp.partValues temp.index] = sort(particles.value(1,:));
  temp.partWeights = particles.weights(temp.index);
  temp.probabilities = 0;
  temp.minIndex = 1;
  temp.maxIndex = particles.N;
  for index = 1:particles.N
    if temp.partValues(index) > crack_est
      break;
```

```
    end
  end
  for j = 1:particles.N
    if index-j > 0
      temp.probabilities = temp.probabilities+temp.partWeights(index-j);
      temp.minIndex = index-j;
    end
    if index+j <= particles.N
      temp.probabilities = temp.probabilities+temp.partWeights(index+j);
      temp.maxIndex = index+j;
    end
    if temp.probabilities >= confInt
      break;
    end
  end
  boundaries = [temp.partValues(temp.minIndex),...
      temp.partValues(temp.maxIndex)];
end
```

B.4. experimentos.m

```
%%%
% ANTES DE EJECUTAR ESTE SCRIPT, VERIFICAR:
%   EN TODOS LOS CASOS
%     ARCHIVO MAIN.m
%       COMENTAR LAS LÍNEAS 26, 50 Y 59
%     ARCHIVO MAINrs.m
%       COMENTAR LAS LÍNEAS 20, 36, 37, 62 Y 71
%   EN CASO DE PRONÓSTICO SIN CÁLCULO DE VALOR ESPERADO
%     ARCHIVO update_particles.m
%       COMENTAR LAS LÍNEAS 13, 15, 16, 17 y 18
%   EN CASO DE PRONÓSTICO CON CÁLCULO DE VALOR ESPERADO
%     ARCHIVO update_particles.m
%       VERIFICAR QUE LAS LÍNEAS 13, 15, 16, 17 y 18 SE EJECUTEN

%-Rutina con la que se llevan a cabo los experimentos
M = 6;
particles.N = 60;
model.L = 350;
failure_i.stdev = 0.10;
model.pc = 1;

% fractura = [fractura = 4.5, p(fractura) <= 0.05; p(fractura) <= 0.1]
fractura1 = zeros(M,3); % filtro clásico
fractura2 = zeros(M,3); % sensible al riesgo con d = 0.01
fractura3 = zeros(M,3); % sensible al riesgo con d = 0.05
fractura4 = zeros(M,3); % sensible al riesgo con d = 0.1
disp('Filtro 1')
for r = 1:M
  MAIN;
```

```
  fractura1(r,:) = failure;
end
disp('Filtro 2')
noise_p2.d = -2*sqrt(0.00015);
for r = 1:M
  alt_MAINrs;
  fractura2(r,:) = failure;
end
disp('Filtro 3')
noise_p2.d = -3*sqrt(0.00015);
for r = 1:M
  alt_MAINrs;
  fractura3(r,:) = failure;
end
disp('Filtro 4')
noise_p2.d = -4*sqrt(0.00015);
for r = 1:M
  alt_MAINrs;
  fractura4(r,:) = failure;
end

t_fractura = [sum(fractura1)/M; sum(fractura2)/M;...
    sum(fractura3)/M; sum(fractura4)/M];

A1 = t_fractura
save alt_EXP1N060L350stdev010M06;
```

B.5. failure_prognosis.m

```
%-Cálculo de la probabilidad de falla en base a una distribución normal en
%   torno al umbral de falla
function probability_of_failure = failure_prognosis(particles,failure_param)
  for j = 1:particles.N
    if particles.value(1,j) >= failure_param.max
      failure_param.prob(1,j) = 1;
    elseif particles.value(1,j) < failure_param.min
      failure_param.prob(1,j) = 0;
    else
      failure_param.prob(1,j) =...
          0.5*(1+erf((particles.value(1,j)-failure_param.mu)/...
          (failure_param.stdev*sqrt(2))));
    end
  end
  probability_of_failure = failure_param.prob*particles.weights';
end
```

B.6. filter_data.m

```
%-Función encargada de suavizar una señal de entrada
function est_signal = filter_data(signal,windowSize)
  est_signal = filter(ones(1,windowSize)/windowSize,1,signal);
end
```

B.7. gauss_likelihood.m

```
%-Cálculo de la verosimilitud en base a una distribución normal
function likelihood = gauss_likelihood(oNoise, error)
  nov = size(error,2);
  normfact = (2*pi)^(size(oNoise.mu,1)/2);
  XX = error - oNoise.mu(:,ones(1,nov));
  S = chol(oNoise.cov)';
  foo = S \ XX;
  likelihood = exp(-0.5*sum(foo.*foo, 1))./(normfact*abs(prod(diag(S))));
end
```

B.8. gauss_sample.m

```
%-Obtención de una muestra de ruido que distribuye como una normal
function sample = gauss_sample(noise, N)
  M = size(noise.mu,1);
  sample = chol(noise.cov)*randn(M,N) + noise.mu(:,ones(1,N));
end
```

B.9. gausspdf.m

```
%-PDF normal
function y = gausspdf(x,mu,sigma)
  y = exp(-(x-mu).^2/(2*sigma^2))/sqrt(2*pi*sigma^2);
end
```

B.10. output_mean.m

```
%-Salida calculada como la media de las partículas y sus pesos
function [crack, alpha] = output_mean(particles)
  crack = sum(particles.value(1,:).*particles.weights);
  alpha = sum(particles.value(2,:).*particles.weights);
end
```

B.11. resampling.m

```
%-Remuestreo de partículas
function particles = resampling(particles, threshold)
  St = round(threshold*particles.N);
  if sum(particles.weights.^2) >= 1/St
    outIndex          = residualresample(1:particles.N,particles.weights);
    particles.value   = particles.value(:,outIndex);
    particles.weights = ones(1,particles.N)/particles.N;
  end
end
```

B.12. residualresample.m

Dado que `residualresample.m` no es un código que haya sido generado por el autor de este texto, se prefirió no publicarlo.

B.13. sog_sample.m

```
%-Obtención de una muestra de ruido que distribuye como suma de dos normales
function sample = sog_sample(N,noise_p1,noise_p2)
  options = optimset('Display','off','TolFun',1e-8);
  mu1 = noise_p1.mu;
  mu2 = noise_p2.mu;
  y = rand(1,N); x = zeros(1,N); x0 = 0;
  for i = 1:N;
    if(y(i) <= 0.5);
      x0 = -0.025;
    else
      x0 = 0.025;
    end;
    while(1)
      [x1,fval,exitflag] = fsolve(@(x)(noise_p1.weight*(0.5*(1+erf((x-mu1(2))/...
          sqrt(2*noise_p1.cov(2,2)))))+noise_p2.weight*(0.5*(1+erf((x-mu2(2))/...
          sqrt(2*noise_p2.cov(2,2)))))-y(i)),x0,options);
      if(exitflag == 1)
        break;
      else
        y(i) = rand; i
        if(y(i) <= 0.5);
          x0 = -0.025;
        else
          x0 = 0.025;
        end;
      end;
    end;
    x(i) = x1;
  end;
  sample = x;
end
```

B.14. update_particles.m

```
%-Actualización del valor de las partículas y sus pesos
function particles = update_particles(particles, model, load_u, obs, n)
  % n es un parámetro que dice si se actualizan los pesos, n=1, o no, n=0
  K1 = interp1(model.Lengths,model.K_1,particles.value(1,:)./2,'spline');
  K2 = interp1(model.Lengths,model.K_2,particles.value(1,:)./2,'spline');
  K1(K1<10) = 10; K2(K2>100) = 100;

  U = interp1([93 100 120],[model.U_93 model.U_100 model.U_120],...
      load_u*ones(1,particles.N),'linear','extrap');

  DELTA_K = model.param.C.*((U.*K1).^model.param.m+(U.*K2).^model.param.m);

  if(n)
    pNoise = gauss_sample(model.noise_p,particles.N);
  else
    mu = model.noise_p.mu;
    pNoise = mu(:,ones(1,particles.N));
  end;

  particles.value = ...
      particles.value + [1; 0]*(particles.value(2,:).*DELTA_K) + pNoise;

  particles.value(1,particles.value(1,:)<=0) = 1;
  particles.value(2,particles.value(2,:)<=0) = 1;

  if(n)
    error = obs - particles.value(1,:);
    likelihood = gauss_likelihood(model.noise_o,error) + 1e-99;

    particles.weights = particles.weights.*likelihood;
    particles.weights = particles.weights/sum(particles.weights);
  end
end
```

B.15. update_particlesrs.m

```
%-Actualización del valor de las partículas y sus pesos (risk-sensitive)
function particles = update_particlesrs(particles, model, load_u, obs, n, pc)
  % n es un parámetro que dice si se actualizan los pesos, n=1, o no, n=0
  K1 = interp1(model.Lengths,model.K_1,particles.value(1,:)./2,'spline');
  K2 = interp1(model.Lengths,model.K_2,particles.value(1,:)./2,'spline');
  K1(K1<10) = 10; K2(K2>100) = 100;

  U = interp1([93 100 120],[model.U_93 model.U_100 model.U_120],...
      load_u*ones(1,particles.N),'linear','extrap');

  DELTA_K = model.param.C.*((U.*K1).^model.param.m+(U.*K2).^model.param.m);
```

```
  mu1_temp = model.noise_p1.mu(1);
  if(n)           % filtraje
    pNoise(1,:) = sqrt(model.noise_p1.cov(1,1))*randn(1,particles.N)+...
        mu1_temp(:,ones(1,particles.N));
    pNoise(2,:) = sog_sample(particles.N,model.noise_p1,model.noise_p2);
  elseif(pc == 1) % predicción caso 1
    pNoise = zeros(2,particles.N);
  elseif(pc == 2) % predicción caso 2
    pNoise(1,:) = sqrt(model.noise_p1.cov(1,1))*randn(1,particles.N)+...
        mu1_temp(:,ones(1,particles.N));
    pNoise(2,:) = zeros(1,particles.N);
  end;

  particles.value = ...
      particles.value + [1; 0]*(particles.value(2,:).*DELTA_K) + pNoise;

  particles.value(1,particles.value(1,:)<=0) = 1;
  particles.value(2,particles.value(2,:)<=0) = 1;

  if(n)
    error = obs - particles.value(1,:);
    likelihood = gauss_likelihood(model.noise_o,error) + 1e-99;

    particles.weights = particles.weights.*likelihood.*...
        (exp(-(pNoise(2,:)).^2/(2*model.noise_p1.cov(2,2))./...
        (model.noise_p1.weight*exp(-(pNoise(2,:)-model.noise_p1.mu(2)).^2/...
         (2*model.noise_p1.cov(2,2)))+...
         model.noise_p2.weight*exp(-(pNoise(2,:)-model.noise_p2.mu(2)).^2/...
         (2*model.noise_p2.cov(2,2)))));
    particles.weights = particles.weights/sum(particles.weights);
  end
end
```