Entrega 3

Juan Rubio Cobeta

11 de diciembre de 2025

Índice

Antes de empezar con la entrega, resaltar estos comentarios:

Como el código ocupaba mucho, he intentado crear un desplegable para los chunks para que ocupe menos, pero según he investigado eso solo es posible si el archivo es .html, no pdf, por lo que no me ha sido posible (me parece que queda un poco feo entregar un pdf de 30 páginas con la mitad de ellas código).

En relación al código, también he intentado añadir algunos comentarios (pocos para que no ocuparan demasiado) que reflejaran las fases mentales por las que he pasado para implementarlo.

1. EJERCICIO 1

Consideremos el sistema dinámico lineal en tiempo discreto con observaciones inciertas definido por las siguientes ecuaciones en diferencias estocásticas:

$$\begin{aligned} x(k+1) &= \Phi(k+1,k)x(k) + \Gamma(k+1,k)w(k), \quad k \geq 0, \quad x(0) = x_0 \\ z(k) &= \gamma(k)H(k)x(k) + v(k), \qquad\qquad\qquad\qquad\quad k \geq 0 \end{aligned}$$

El sistema satisface las siguientes hipótesis estructurales y estocásticas:

1. **Condición Inicial:** El estado inicial x_0 es un vector aleatorio n-dimensional gaussiano centrado, caracterizado por:
$$E[x_0] = 0, \quad E[x_0 x_0^T] = P_0$$

2. **Ruido del Proceso:** La sucesión $\{w(k); k \geq 0\}$ es un proceso de ruido blanco gaussiano centrado con covarianza:
$$E[w(k)w^T(j)] = Q(k)\delta_{kj}$$
donde δ_{kj} es la delta de Kronecker.

3. **Incertidumbre en la Observación:** El proceso multiplicativo $\{\gamma(k); k \geq 0\}$ es una sucesión de variables aleatorias independientes de Bernoulli, que modela la presencia de la señal en la medida:
$$P(\gamma(k) = 1) = p(k), \quad P(\gamma(k) = 0) = 1 - p(k)$$

4. **Ruido de Medida:** La sucesión $\{v(k); k \geq 0\}$ es un ruido blanco gaussiano centrado con covarianza definida positiva:
$$E[v(k)v^T(j)] = R(k)\delta_{kj}, \quad R(k) > 0$$

5. **Independencia Mutua:** El estado inicial x_0 y los procesos estocásticos $\{w(k)\}$, $\{v(k)\}$ y $\{\gamma(k)\}$ son mutuamente independientes para todo $k \geq 0$.

1.1. Apartado a):

El objetivo es demostrar que la matriz de covarianzas del proceso de innovación, definida como $\Pi(k) = E[\tilde{z}(k/k-1)\tilde{z}^T(k/k-1)]$, satisface la siguiente expresión:

$$\Pi(k) = p(k)(1-p(k))H(k)D(k)H^T(k) + p^2(k)H(k)P(k/k-1)H^T(k) + R(k)$$

1.1.1. Definición del proceso de innovación

La innovación se define como la diferencia entre la observación actual y su predicción óptima basada en la información previa:

$$\tilde{z}(k/k-1) = z(k) - \hat{z}(k/k-1)$$

Calculamos el predictor de la observación $\hat{z}(k/k-1)$. Dado que $v(k)$ y $\gamma(k)$ son independientes de las observaciones pasadas, y $E[v(k)] = 0$, tenemos:

$$\begin{aligned} \hat{z}(k/k-1) &= E[z(k)|Z_{k-1}] \\ &= E[\gamma(k)H(k)x(k) + v(k)|Z_{k-1}] \\ &= E[\gamma(k)]H(k)E[x(k)|Z_{k-1}] + 0 \\ &= p(k)H(k)\hat{x}(k/k-1) \end{aligned}$$

Sustituimos $z(k)$ y $\hat{z}(k/k-1)$ en la expresión de la innovación:

$$\tilde{z}(k/k-1) = [\gamma(k)H(k)x(k) + v(k)] - p(k)H(k)\hat{x}(k/k-1)$$

1.1.2. Descomposición de la innovación

Para calcular la varianza, es útil expresar la innovación en términos del error de estimación del estado, $\tilde{x}(k/k-1) = x(k) - \hat{x}(k/k-1)$. Despejando $\hat{x}(k/k-1) = x(k) - \tilde{x}(k/k-1)$ y sustituyendo arriba:

$$\begin{aligned}\tilde{z}(k/k-1) &= \gamma(k)H(k)x(k) + v(k) - p(k)H(k)[x(k) - \tilde{x}(k/k-1)] \\ &= \gamma(k)H(k)x(k) - p(k)H(k)x(k) + p(k)H(k)\tilde{x}(k/k-1) + v(k)\end{aligned}$$

Agrupamos los términos para identificar las fuentes de aleatoriedad independientes:

$$\tilde{z}(k/k-1) = \underbrace{[\gamma(k) - p(k)]H(k)x(k)}_{\text{Término A}} + \underbrace{p(k)H(k)\tilde{x}(k/k-1)}_{\text{Término B}} + \underbrace{v(k)}_{\text{Término C}}$$

1.1.3. Cálculo de la matriz de covarianzas

Calculamos $\Pi(k) = E[\tilde{z}(k/k-1)\tilde{z}^T(k/k-1)]$. Debido a las hipótesis de independencia del sistema (el ruido blanco $v(k)$ y la variable Bernoulli $\gamma(k)$ son independientes entre sí y respecto al estado $x(k)$ y al error pasado), los términos cruzados (covarianzas cruzadas) se anulan:

- $E[AB^T] = 0$: Porque $\gamma(k)$ es independiente del error de estimación.
- $E[AC^T] = 0$: Porque $\gamma(k)$ y $x(k)$ son independientes de $v(k)$.
- $E[BC^T] = 0$: Porque el error pasado es independiente del ruido presente.

Por tanto, la covarianza es la suma de las covarianzas de los tres términos individuales:

$$\Pi(k) = E[AA^T] + E[BB^T] + E[CC^T]$$

Analizamos cada término:

Término 1 (AA^T): Incertidumbre de observación

$$\begin{aligned}E[AA^T] &= E\left[(\gamma(k) - p(k))^2 H(k)x(k)x^T(k)H^T(k)\right] \\ &= E\left[(\gamma(k) - p(k))^2\right] H(k)E\left[x(k)x^T(k)\right] H^T(k)\end{aligned}$$

Sabemos que la varianza de una variable Bernoulli es $\text{Var}(\gamma(k)) = p(k)(1-p(k))$ y definimos la matriz de segundos momentos del estado como $D(k) = E[x(k)x^T(k)]$.

$$E[AA^T] = p(k)(1-p(k))H(k)D(k)H^T(k)$$

Término 2 (BB^T): Error de estimación

$$\begin{aligned}E[BB^T] &= p^2(k)H(k)E\left[\tilde{x}(k/k-1)\tilde{x}^T(k/k-1)\right] H^T(k) \\ &= p^2(k)H(k)P(k/k-1)H^T(k)\end{aligned}$$

Término 3 (CC^T): Ruido de medida

$$E[CC^T] = E[v(k)v^T(k)] = R(k)$$

1.1.4. Resultado Final

Sumando los tres componentes, obtenemos la expresión deseada:

$$\Pi(k) = p(k)(1-p(k))H(k)D(k)H^T(k) + p^2(k)H(k)P(k/k-1)H^T(k) + R(k)$$

Verificación para $k = 0$:

En el instante inicial, el predictor es $\hat{x}(0/-1) = E[x_0] = 0$, por lo que el error es $\tilde{x}(0/-1) = x_0$. Esto implica que $P(0/-1) = E[x_0 x_0^T] = P_0$. Asimismo, $D(0) = E[x_0 x_0^T] = P_0$.

Sustituyendo en la fórmula general:

$$\begin{aligned}\Pi(0) &= [p(0)(1-p(0))]H(0)P_0H^T(0) + p^2(0)H(0)P_0H^T(0) + R(0)\\ &= [p(0) - p^2(0) + p^2(0)]H(0)P_0H^T(0) + R(0)\\ &= p(0)H(0)P_0H^T(0) + R(0)\end{aligned}$$

Lo cual coincide con la expresión dada en el enunciado.

1.2. Apartado b)

El objetivo es obtener el estimador de suavizamiento de punto fijo $\hat{x}(L/k)$ para un instante fijo L utilizando las observaciones hasta el instante k ($k > L$).

Utilizando el Lema de Proyecciones Ortogonales, podemos expresar el estimador en el instante k como una actualización del estimador en $k-1$ más un término de corrección basado en la nueva información (la innovación $\tilde{z}(k/k-1)$):

$$\hat{x}(L/k) = \hat{x}(L/k-1) + K_s(k)\tilde{z}(k/k-1)$$

Donde $K_s(k)$ es la matriz de ganancia del suavizador.

1.2.1. Cálculo de la Ganancia del Suavizador $K_s(k)$

Por el Lema de Proyecciones, la ganancia óptima que minimiza el error cuadrático medio está dada por:

$$K_s(k) = E[x(L)\tilde{z}^T(k/k-1)]\Pi^{-1}(k)$$

Desarrollamos el término de esperanza cruzada $E[x(L)\tilde{z}^T(k/k-1)]$. Recordamos la descomposición de la innovación $\tilde{z}(k/k-1)$ deducida en el apartado anterior:

$$\tilde{z}(k/k-1) = [\gamma(k) - p(k)]H(k)x(k) + p(k)H(k)\tilde{x}(k/k-1) + v(k)$$

Multiplicamos por $x(L)$ y tomamos la esperanza. Debido a la independencia de $\gamma(k)$ y $v(k)$ respecto al estado $x(L)$ y al pasado, los primeros y últimos términos se anulan:

1. $E[x(L)(\gamma(k) - p(k))\dots] = 0$
2. $E[x(L)v^T(k)] = 0$

Nos queda el término central:

$$\begin{aligned}E[x(L)\tilde{z}^T(k/k-1)] &= E[x(L)\tilde{x}^T(k/k-1)H^T(k)p(k)]\\ &= E[x(L)\tilde{x}^T(k/k-1)]H^T(k)p(k)\end{aligned}$$

Introducimos el error de suavizamiento $\tilde{x}(L/k-1) = x(L) - \hat{x}(L/k-1)$. Dado que $\hat{x}(L/k-1)$ es una proyección sobre el espacio de observaciones Z_{k-1} y $\tilde{x}(k/k-1)$ es ortogonal a dicho espacio (por propiedad del filtro), podemos sustituir $x(L)$ por $\tilde{x}(L/k-1)$ en la esperanza:

$$E[x(L)\tilde{x}^T(k/k-1)] = E[\tilde{x}(L/k-1)\tilde{x}^T(k/k-1)]$$

Definimos la matriz auxiliar de covarianza cruzada como $\Sigma(k) \triangleq E[\tilde{x}(L/k-1)\tilde{x}^T(k/k-1)]$. Por tanto, la ganancia del suavizador es:

$$K_s(k) = \Sigma(k)H^T(k)p(k)\Pi^{-1}(k)$$

1.2.2. Ecuación Recursiva para $\Sigma(k)$

Necesitamos una recursión para calcular $\Sigma(k)$. Partimos de la ecuación de evolución del error de predicción del filtro:

$$\tilde{x}(k/k-1) = \Phi(k,k-1)\tilde{x}(k-1/k-1) + \Gamma(k,k-1)w(k-1)$$

Sustituimos esto en la definición de $\Sigma(k)$:

$$\begin{aligned}\Sigma(k) &= E[\tilde{x}(L/k-1)\{\Phi(k,k-1)\tilde{x}(k-1/k-1) + \Gamma(k,k-1)w(k-1)\}^T] \\ &= E[\tilde{x}(L/k-1)\tilde{x}^T(k-1/k-1)]\Phi^T(k,k-1)\end{aligned}$$

(El término con $w(k-1)$ se anula por independencia).

Ahora, recordamos que el estimador suavizado en la etapa anterior es $\hat{x}(L/k-1) = \hat{x}(L/k-2) + K_s(k-1)\tilde{z}(k-1/k-2)$. Restando esto de $x(L)$, obtenemos la recursión del error:

$$\tilde{x}(L/k-1) = \tilde{x}(L/k-2) - K_s(k-1)\tilde{z}(k-1/k-2)$$

Sustituimos esto en la esperanza anterior:

$$E[\{\tilde{x}(L/k-2) - K_s(k-1)\tilde{z}(k-1/k-2)\}\tilde{x}^T(k-1/k-1)]\Phi^T(k,k-1)$$

Sabemos que el error de filtrado $\tilde{x}(k-1/k-1)$ es ortogonal a la innovación $\tilde{z}(k-1/k-2)$ (propiedad fundamental del filtro óptimo). Por tanto, el segundo término dentro del corchete es cero. Nos queda:

$$E[\tilde{x}(L/k-2)\tilde{x}^T(k-1/k-1)]\Phi^T(k,k-1)$$

Finalmente, expresamos el error de filtrado en función del error de predicción usando la ganancia del filtro K_f:

$$\tilde{x}(k-1/k-1) = [I - K_f(k-1)p(k-1)H(k-1)]\tilde{x}(k-1/k-2) + \dots$$

Sustituyendo y tomando esperanzas, llegamos a la recursión final para $\Sigma(k)$:

$$\Sigma(k) = \Sigma(k-1)[I - K_f(k-1)p(k-1)H(k-1)]^T\Phi^T(k,k-1)$$

1.2.3. Resumen del Algoritmo

El algoritmo de suavizamiento punto fijo para sistemas con observaciones inciertas se inicializa en $k = L$ con los valores del filtro $\hat{x}(L/L)$ y $P(L/L)$. Para $k = L+1, L+2, \ldots$:

1. Matrices del Filtro (Nahi): Se asume conocido el cálculo de $\Pi(k)$, $\tilde{z}(k/k-1)$, $K_f(k)$ y $P(k/k)$.
2. Actualización de Matriz Auxiliar:

$$\Sigma(k) = \Sigma(k-1)[I - K_f(k-1)p(k-1)H(k-1)]^T \Phi^T(k, k-1)$$

   Inicialización: $\Sigma(L+1) = P(L/L)\Phi^T(L+1, L)$.

3. Ganancia de Suavizamiento:

$$K_s(k) = \Sigma(k)H^T(k)p(k)\Pi^{-1}(k)$$

4. Estimación Suavizada:

$$\hat{x}(L/k) = \hat{x}(L/k-1) + K_s(k)\tilde{z}(k/k-1)$$

1.3. Apartado c)

Consideramos el sistema escalar específico dado por:

$$\begin{aligned} x(k+1) &= 0.95x(k) + w(k) \\ z(k) &= \gamma(k)x(k) + v(k) \end{aligned}$$

Con parámetros: $\Phi = 0.95$, $\Gamma = 1$, $H = 1$, $Q = 0.1$, $R = 0.5$, $p = 0.5$.

El ciclo computacional para la implementación conjunta del filtro y el suavizador de punto fijo en un instante L fijo es el siguiente:

1.3.1. Inicialización ($k = 0$)

Se inicializan las variables del filtro con la media y covarianza a priori:

$$\begin{aligned} \hat{x}(0/-1) &= 0 \\ P(0/-1) &= P_0 = 1 \\ D(0) &= P_0 = 1 \end{aligned}$$

1.3.2. Bucle de Filtrado (Para $k = 0, 1, \ldots, N$)

En cada instante k, se ejecutan los siguientes pasos secuenciales:

1. Cálculo de la covarianza de la innovación ($\Pi(k)$):

$$\Pi(k) = p(1-p)D(k) + p^2 P(k/k-1) + R$$

   ($H = 1$).

2. Cálculo de la ganancia del filtro ($K(k)$):

$$K(k) = pP(k/k-1)\Pi^{-1}(k)$$

3. Cálculo de la innovación ($\tilde{z}(k)$):

$$\tilde{z}(k/k-1) = z(k) - p\hat{x}(k/k-1)$$

4. Actualización de la estimación (Filtrado):

$$\hat{x}(k/k) = \hat{x}(k/k-1) + K(k)\tilde{z}(k/k-1)$$

5. Actualización de la covarianza del error ($P(k/k)$):

$$P(k/k) = P(k/k-1) - pK(k)P(k/k-1)$$

6. Predicción para el siguiente instante ($k+1$):

$$\begin{aligned} \hat{x}(k+1/k) &= \Phi\hat{x}(k/k) \\ P(k+1/k) &= \Phi^2 P(k/k) + Q \\ D(k+1) &= \Phi^2 D(k) + Q \end{aligned}$$

1.3.3. Bucle del Suavizador de Punto Fijo (Para un L fijo)

Queremos mejorar la estimación de $x(L)$ usando datos futuros $z(k)$ con $k > L$.

Inicialización ($k = L$):

$$\hat{x}(L/L) \quad \text{(Obtenido del filtro)}$$

$$\Sigma(L+1) = P(L/L)\Phi \quad \text{(Matriz auxiliar inicial)}$$

Iteración (Para $k = L+1, \ldots, N$):

1. **Ganancia del Suavizador:**

$$K_s(k) = \Sigma(k)p\Pi^{-1}(k)$$

(Usa $\Pi(k)$ calculado en el filtro).

2. **Actualización de la Estimación Suavizada:**

$$\hat{x}(L/k) = \hat{x}(L/k-1) + K_s(k)\tilde{z}(k/k-1)$$

(Usa $\tilde{z}$ del filtro).

3. **Actualización de la Matriz Auxiliar:**

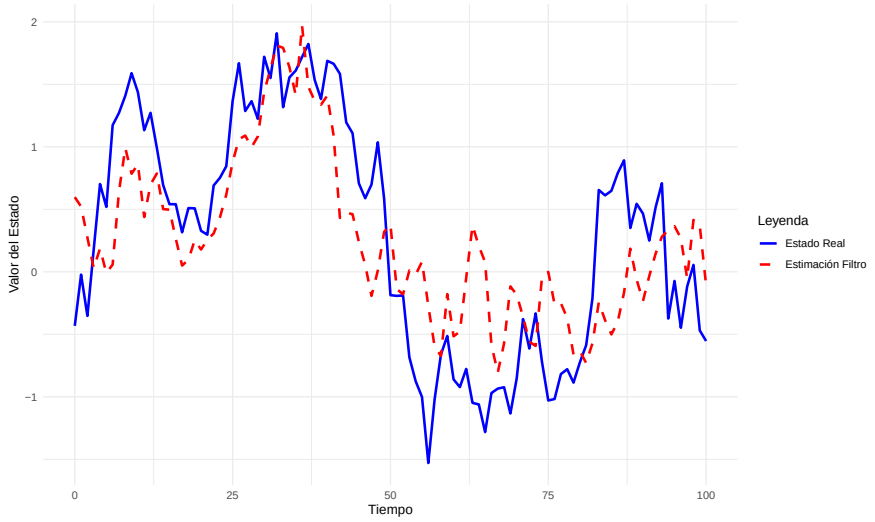
$$\Sigma(k+1) = \Sigma(k)[1 - K(k)p]\Phi$$

(Usa $K(k)$ del filtro).

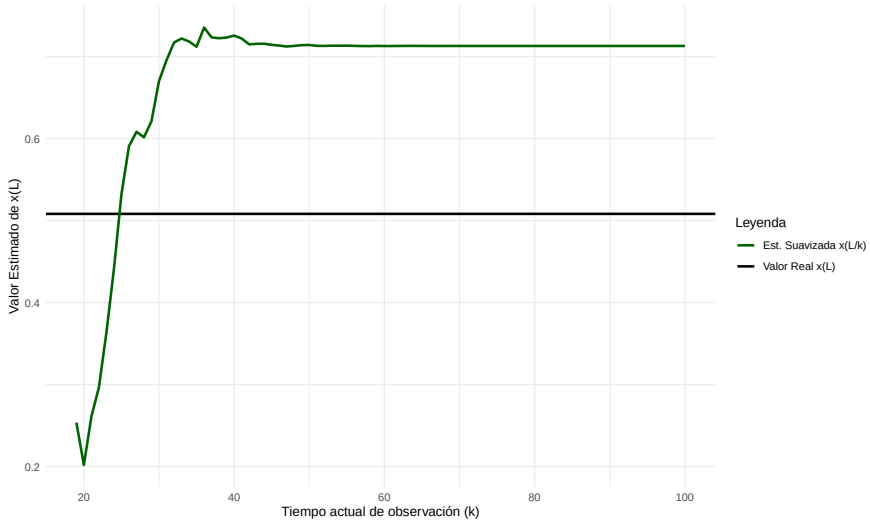
```
library(tidyverse)
set.seed(123)
N_iter <- 100
Phi <- 0.95
Gamma <- 1
H <- 1
Q <- 0.1
R <- 0.5
P0 <- 1
p_prob <- 0.5

# 1. Simulación del Sistema Real
x <- numeric(N_iter + 1)
z <- numeric(N_iter + 1)
gamma <- rbinom(N_iter + 1, 1, p_prob)
x[1] <- rnorm(1, 0, sqrt(P0)) # x(0)
for (k in 1:N_iter) {
  v <- rnorm(1, 0, sqrt(R))
  z[k] <- gamma[k] * H * x[k] + v
    w <- rnorm(1, 0, sqrt(Q))
```

```
  x[k+1] <- Phi * x[k] + w
}
z[N_iter+1] <- gamma[N_iter+1] * H * x[N_iter+1] + rnorm(1, 0, sqrt(R))

# 2. Implementación del Filtro (Nahi)
x_hat_pred <- numeric(N_iter + 1) # x(k/k-1)
x_hat_filt <- numeric(N_iter + 1) # x(k/k)
P_pred <- numeric(N_iter + 1)     # P(k/k-1)
P_filt <- numeric(N_iter + 1)     # P(k/k)
D <- numeric(N_iter + 1)          # D(k)
Pi_innov <- numeric(N_iter + 1)   # Covarianza innovación
z_tilde <- numeric(N_iter + 1)    # Innovación
K_gain <- numeric(N_iter + 1)     # Ganancia

x_hat_pred[1] <- 0
P_pred[1] <- P0
D[1] <- P0

for (k in 1:(N_iter + 1)) {
  term1 <- p_prob * (1 - p_prob) * H * D[k] * H
  term2 <- p_prob^2 * H * P_pred[k] * H
  Pi_innov[k] <- term1 + term2 + R
    K_gain[k] <- p_prob * P_pred[k] * H * (1/Pi_innov[k])
    z_tilde[k] <- z[k] - p_prob * H * x_hat_pred[k]
    x_hat_filt[k] <- x_hat_pred[k] + K_gain[k] * z_tilde[k]
  P_filt[k] <- (1 - K_gain[k] * p_prob * H) * P_pred[k]
    if (k <= N_iter) {
    x_hat_pred[k+1] <- Phi * x_hat_filt[k]
    P_pred[k+1] <- Phi * P_filt[k] * Phi + Q # Gamma=1
    D[k+1] <- Phi * D[k] * Phi + Q
  }
}

# 3. Implementación del Suavizador Punto Fijo
L_idx <- 20
L_real <- L_idx - 1

x_smooth_L <- numeric(N_iter + 1)
x_smooth_L[1:L_idx] <- NA
x_smooth_L[L_idx] <- x_hat_filt[L_idx]
# Matriz auxiliar Sigma inicial: Sigma(L+1) = P(L/L) * Phi'
Sigma <- P_filt[L_idx] * Phi
for (k in (L_idx + 1):(N_iter + 1)) {
  Ks <- Sigma * H * p_prob * (1/Pi_innov[k])
    x_smooth_L[k] <- x_smooth_L[k-1] + Ks * z_tilde[k]
  Sigma <- Sigma * (1 - K_gain[k] * p_prob * H) * Phi
}

# 4. Visualización de Resultados
df_filter <- data.frame(
  Tiempo = 0:N_iter,
  Real = x,
  Estimado = x_hat_filt,
```

```
  Tipo = "Filtro (Nahi)"
)
p1 <- ggplot(df_filter, aes(x=Tiempo)) +
  geom_line(aes(y=Real, color="Estado Real"), linewidth=1) +
  geom_line(aes(y=Estimado, color="Estimación Filtro"), linetype="dashed", linewidth=1) +
  labs(y = "Valor del Estado", color = "Leyenda") +
  theme_minimal() +
  scale_color_manual(values=c("blue", "red"))
df_smooth <- data.frame(
  Tiempo = L_real:N_iter,
  Estimacion_L = x_smooth_L[L_idx:(N_iter+1)],
  Valor_Real_L = x[L_idx]
)
p2 <- ggplot(df_smooth, aes(x=Tiempo)) +
  geom_hline(aes(yintercept=Valor_Real_L, color="Valor Real x(L)"), linewidth=1) +
  geom_line(aes(y=Estimacion_L, color="Est. Suavizada x(L/k)"), linewidth=1) +
  labs(y = "Valor Estimado de x(L)", x = "Tiempo actual de observación (k)",
       color = "Leyenda") +
  theme_minimal() +
  scale_color_manual(values=c("darkgreen", "black"))
print(p1)
```




```
print(p2)
```



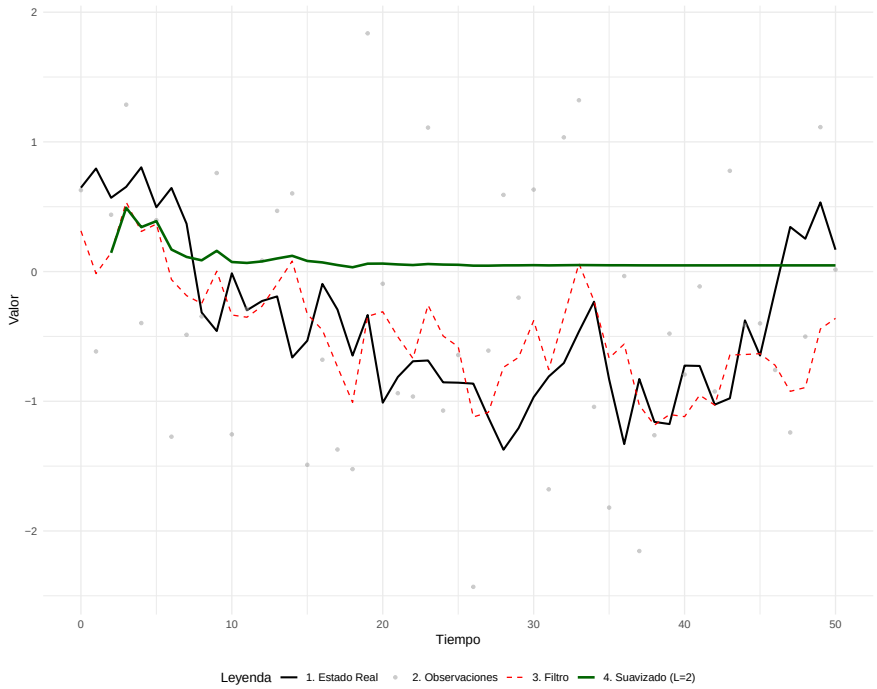
1.4. Apartado d)

En este apartado simulamos 50 iteraciones del sistema. Se implementa el cálculo recursivo de la varianza del error de suavizamiento para comparar la incertidumbre del filtro frente a la del suavizador en instantes específicos ($L = 1, 2, 4$).

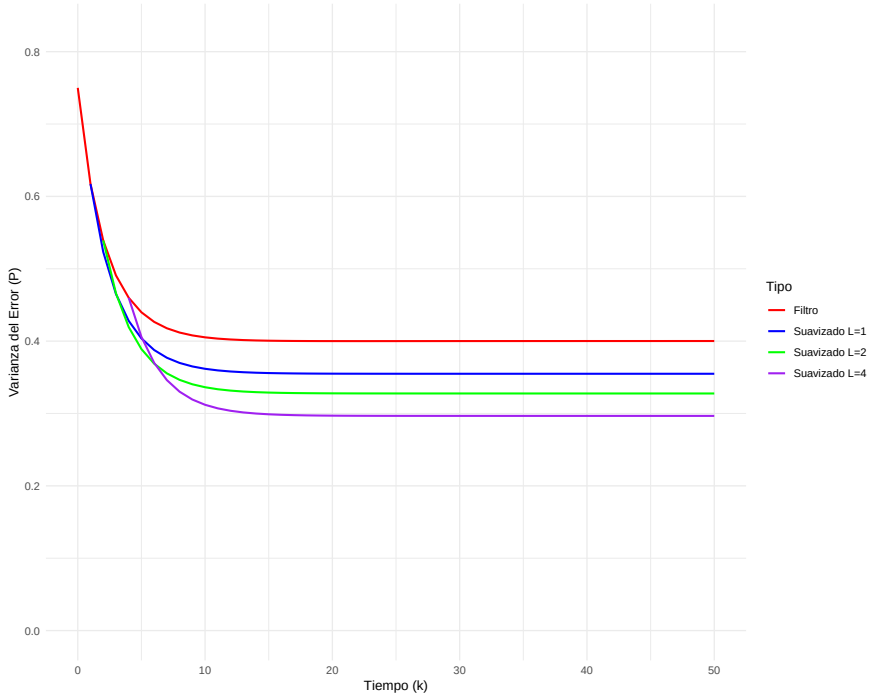
```
library(tidyverse)

# 1. Configuración y Simulación
set.seed(2024)
N_iter <- 50
Phi <- 0.95
H <- 1
Q <- 0.1
R <- 0.5
P0 <- 1
p_prob <- 0.5
x <- numeric(N_iter + 1)
z <- numeric(N_iter + 1)
gamma <- rbinom(N_iter + 1, 1, p_prob)
x[1] <- rnorm(1, 0, sqrt(P0))
z[1] <- gamma[1] * H * x[1] + rnorm(1, 0, sqrt(R))
for (k in 1:N_iter) {
  w <- rnorm(1, 0, sqrt(Q))
  x[k+1] <- Phi * x[k] + w
  v <- rnorm(1, 0, sqrt(R))
  z[k+1] <- gamma[k+1] * H * x[k+1] + v
}

# 2. Ejecución del Filtro (Nahi)
x_hat_pred <- numeric(N_iter + 1)
x_hat_filt <- numeric(N_iter + 1)
P_pred <- numeric(N_iter + 1)
P_filt <- numeric(N_iter + 1) # Varianza filtrado P(k/k)
D <- numeric(N_iter + 1)
Pi_hist <- numeric(N_iter + 1) # Histórico de Pi(k)
z_tilde_hist <- numeric(N_iter + 1) # Histórico de innovaciones
```

```r
K_hist <- numeric(N_iter + 1) # Histórico de ganancias filtro
x_hat_pred[1] <- 0
P_pred[1] <- P0
D[1] <- P0

for (k in 1:(N_iter + 1)) {
  term1 <- p_prob * (1 - p_prob) * H * D[k] * H
  term2 <- p_prob^2 * H * P_pred[k] * H
  Pi_val <- term1 + term2 + R
  Pi_hist[k] <- Pi_val

  K_val <- p_prob * P_pred[k] * H * (1/Pi_val)
  K_hist[k] <- K_val

  z_tilde <- z[k] - p_prob * H * x_hat_pred[k]
  z_tilde_hist[k] <- z_tilde
    x_hat_filt[k] <- x_hat_pred[k] + K_val * z_tilde

  P_filt[k] <- (1 - K_val * p_prob * H) * P_pred[k]

  if (k <= N_iter) {
    x_hat_pred[k+1] <- Phi * x_hat_filt[k]
    P_pred[k+1] <- Phi * P_filt[k] * Phi + Q
    D[k+1] <- Phi * D[k] * Phi + Q
  }
}

# 3. Ejecución de Suavizadores Punto Fijo
run_smoother <- function(L_target) {
  L_idx <- L_target + 1
    x_smooth <- numeric(N_iter + 1)
  P_smooth <- numeric(N_iter + 1)
    x_smooth[1:L_idx] <- NA
  P_smooth[1:L_idx] <- NA
    x_smooth[L_idx] <- x_hat_filt[L_idx]
  P_smooth[L_idx] <- P_filt[L_idx]
    Sigma <- P_filt[L_idx] * Phi
    if (L_idx < (N_iter + 1)) {
    for (k in (L_idx + 1):(N_iter + 1)) {
      Pi_k <- Pi_hist[k]
      z_tilde_k <- z_tilde_hist[k]
      K_filt_k <- K_hist[k]
      Ks <- Sigma * H * p_prob * (1/Pi_k)
      x_smooth[k] <- x_smooth[k-1] + Ks * z_tilde_k
      P_smooth[k] <- P_smooth[k-1] - Ks * Pi_k * Ks
      Sigma <- Sigma * (1 - K_filt_k * p_prob * H) * Phi
    }
  }
  return(list(x = x_smooth, P = P_smooth))
}
res_L1 <- run_smoother(1)
res_L2 <- run_smoother(2)
res_L4 <- run_smoother(4)
```

```r
# 4. Preparación de Datos para Gráficas

time_steps <- 0:N_iter
df_main <- data.frame(
  Tiempo = time_steps,
  Estado_Real = x,
  Observaciones = z,
  Filtro = x_hat_filt,
  Suavizado_L2 = res_L2$x
)

df_vars <- data.frame(
  Tiempo = time_steps,
  Var_Filtro = P_filt,
  Var_Smooth_L1 = res_L1$P,
  Var_Smooth_L2 = res_L2$P,
  Var_Smooth_L4 = res_L4$P
) %>%
  pivot_longer(cols = starts_with("Var"), names_to = "Tipo", values_to = "Varianza")

# 5. Generación de Gráficas

# Gráfica 1: Trayectorias (Estado, Obs, Filtro, Suavizado L=2)
p1 <- ggplot(df_main, aes(x = Tiempo)) +
  geom_line(aes(y = Estado_Real, color = "1. Estado Real"), linewidth = 0.8) +
  geom_point(aes(y = Observaciones, color = "2. Observaciones"), size = 1, alpha = 0.5) +
  geom_line(aes(y = Filtro, color = "3. Filtro"), linetype = "dashed") +
  geom_line(aes(y = Suavizado_L2, color = "4. Suavizado (L=2)"), linewidth = 1) +
  labs(y = "Valor", color = "Leyenda") +
  scale_color_manual(values = c("black", "grey60", "red", "darkgreen")) +
  theme_minimal() +
  theme(legend.position = "bottom")

# Gráfica 2: Varianzas de Error
p2 <- ggplot(df_vars, aes(x = Tiempo, y = Varianza, color = Tipo)) +
  geom_line(linewidth = 0.8) +
  labs(x = "Tiempo (k)", y = "Varianza del Error (P)") +
  scale_color_manual(values = c("red", "blue", "green", "purple"),
                     labels = c("Filtro", "Suavizado L=1", "Suavizado L=2", "Suavizado L=4")) +
  theme_minimal() +
  coord_cartesian(ylim = c(0, max(P_filt)*1.1))

print(p1)
```




```
print(p2)
```



1.4.1. Comentario de los Resultados

En las simulaciones realizadas para el sistema con observaciones inciertas ($p = 0.5$), se han obtenido dos gráficas fundamentales que permiten validar el correcto funcionamiento del algoritmo de filtrado de Nahi y del suavizador de punto fijo.

1. Análisis de Trayectorias (Gráfica 1)

En la primera gráfica se representa una realización del sistema durante $N = 50$ iteraciones.

Dinámica del Filtro: La estimación del filtro $\hat{x}(k/k)$ (línea roja discontinua) consigue rastrear la evolución del estado real $x(k)$, a pesar de la alta incertidumbre introducida por la probabilidad de falsa alarma $(1 - p = 0.5)$. Se observa que el filtro reacciona a las observaciones, pero mantiene un nivel de error apreciable debido a que, en promedio, la mitad de las observaciones contienen únicamente ruido.

Comportamiento del Suavizador ($L = 2$): La línea verde muestra la evolución de la estimación $\hat{x}(L/k)$ para el instante fijo $L = 2$ a medida que avanza el tiempo k. Se aprecia cómo la estimación varía durante las primeras iteraciones posteriores a $k = 2$ (fase transitoria), corrigiendo el valor inicial del filtro gracias a la información aportada por las innovaciones futuras. Pasado un cierto horizonte temporal (aprox. $k > 15$), la estimación se estabiliza y permanece prácticamente constante, lo que indica que las observaciones muy lejanas en el futuro dejan de aportar información significativa sobre el estado en $L = 2$.

2. Análisis de las Varianzas del Error (Gráfica 2)

La segunda gráfica ofrece la validación teórica más robusta de la implementación.

Estabilidad del Filtro: La varianza del error de filtrado $P(k/k)$ (línea roja) parte de $P_0 = 1$ y converge rápidamente hacia un régimen estacionario (alrededor de 0.4). Esto confirma la estabilidad del algoritmo de Nahi.

Mejora por Suavizamiento: Las curvas de varianza de los suavizadores para $L = 1, 2, 4$ (líneas azul, verde y morada) presentan un comportamiento monótonamente decreciente. Parten del valor de la varianza del filtro en el instante L $(P(L/L))$ y disminuyen conforme k aumenta, estabilizándose asintóticamente en un valor inferior (aprox. 0.3).

Conclusión: El hecho de que $P(L/k) < P(L/L)$ para todo $k > L$ confirma que el procesamiento de datos "a posteriori" (suavizamiento) reduce la incertidumbre de la estimación respecto al filtrado en tiempo real. La diferencia entre el nivel estacionario rojo (filtro) y los niveles de los suavizadores cuantifica la ganancia de precisión obtenida mediante el algoritmo de punto fijo.

1.5. Apartado e)

En este apartado estudiamos la sensibilidad del Estimador Lineal Mínimo Cuadrático ante variaciones en la probabilidad de detección de la señal, p. El objetivo es visualizar cómo la incertidumbre sobre la presencia de la señal afecta a la precisión final de la estimación.

Para ello, evaluamos la evolución recursiva de la varianza del error de filtrado $P(k/k)$ para un conjunto de valores $p \in \{0.1, 0.3, 0.5, 0.7, 0.9, 1.0\}$, manteniendo constantes el resto de parámetros del sistema.

Las ecuaciones deterministas que gobiernan la evolución de la covarianza son:

1. **Evolución del segundo momento del estado:**
$$D(k+1) = \Phi D(k)\Phi^T + \Gamma Q \Gamma^T, \quad D(0) = P_0$$

2. **Covarianza de la innovación (dependiente de p):**
$$\Pi(k) = p(1-p)HD(k)H^T + p^2 HP(k/k-1)H^T + R$$

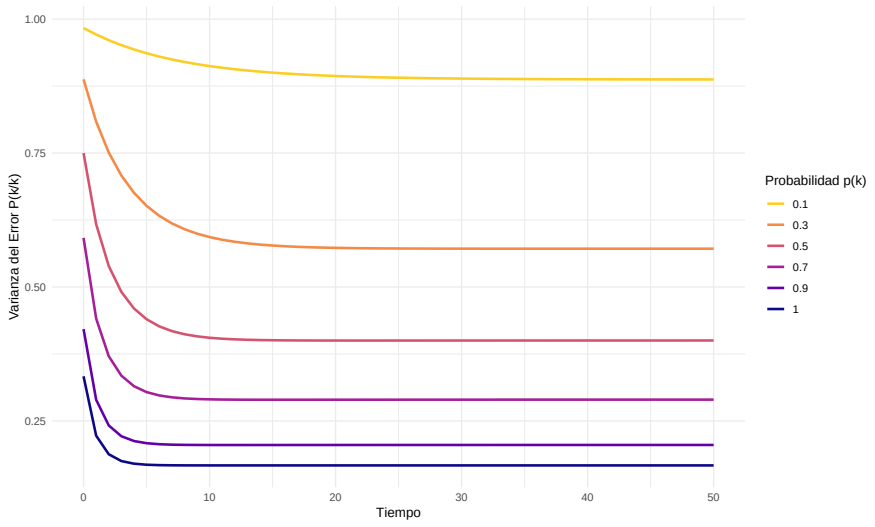
3. **Ganancia del filtro:**
$$K(k) = pP(k/k-1)H^T \Pi^{-1}(k)$$

4. **Actualización de la varianza del error (Filtrado):**
$$P(k/k) = P(k/k-1) - pK(k)HP(k/k-1)$$

5. **Predicción de la varianza:**
$$P(k+1/k) = \Phi P(k/k)\Phi^T + \Gamma Q \Gamma^T$$

```r
library(tidyverse)

# Parámetros del Sistema
N_iter <- 50
Phi <- 0.95
H <- 1
Q <- 0.1
R <- 0.5
P0 <- 1

p_values <- c(0.1, 0.3, 0.5, 0.7, 0.9, 1.0)

# Función
calc_variance_evolution <- function(p_in, N, Phi, H, Q, R, P0) {
  # Inicialización
  P_pred <- P0
  D <- P0
  var_history <- numeric(N + 1)
  for (k in 1:(N + 1)) {
    term1 <- p_in * (1 - p_in) * H * D * H
    term2 <- p_in^2 * H * P_pred * H
    Pi_val <- term1 + term2 + R
    K_val <- p_in * P_pred * H * (1/Pi_val)
    # P(k/k) = P(k/k-1) - p K H P(k/k-1)
    P_filt <- P_pred - p_in * K_val * H * P_pred
    var_history[k] <- P_filt
    P_pred <- Phi * P_filt * Phi + Q
    D <- Phi * D * Phi + Q
  }

  return(var_history)
}

# Ejecución para todos los valores de p
results_list <- list()
for (val in p_values) {
  trayectoria <- calc_variance_evolution(val, N_iter, Phi, H, Q, R, P0)
  results_list[[as.character(val)]] <- data.frame(
    Tiempo = 0:N_iter,
    Varianza = trayectoria,
    Probabilidad = as.factor(val)
  )
}
df_comparison <- bind_rows(results_list)

# Visualización
p_comparison <- ggplot(df_comparison, aes(x = Tiempo, y = Varianza, color = Probabilidad)) +
  geom_line(linewidth = 1) +
  labs(y = "Varianza del Error P(k/k)",
       color = "Probabilidad p(k)") +
  theme_minimal() +
  scale_color_viridis_d(option = "plasma", end = 0.9, direction = -1) +
  theme(legend.position = "right")
```

```
print(p_comparison)
```



1.5.1. Análisis de Sensibilidad y Discusión de Resultados

La gráfica obtenida muestra la evolución temporal de la varianza del error de estimación a posteriori, $P(k/k)$, parametrizada por la probabilidad de presencia de señal $p \in \{0.1, ..., 1.0\}$. El análisis de estas curvas permite extraer conclusiones fundamentales sobre la robustez y los límites teóricos del estimador:

1. Monotonicidad Inversa respecto a la Incertidumbre Se observa una relación estrictamente monótona decreciente entre la probabilidad p y la varianza del error en estado estacionario (P_∞).

$$\text{Si } p_1 > p_2 \implies P_\infty(p_1) < P_\infty(p_2)$$

Esto es consistente con la teoría de la información: un mayor valor de p implica una mayor relación señal-a-ruido efectiva y una mayor frecuencia de actualizaciones de información válida. El caso límite $p = 1$ (curva azul oscuro) corresponde al Filtro de Kalman estándar, estableciendo la cota inferior teórica de la varianza del error (aprox. 0.28 en este sistema).

2. Inflación de la Covarianza por el Término Bernoulli La degradación del desempeño para valores bajos de p (ej. curva amarilla $p = 0.1$) no se debe solo a la "ausencia" de datos, sino a la incertidumbre intrínseca añadida a la innovación. Recordando la expresión de la covarianza de la innovación:

$$\Pi(k) = \underbrace{p(1-p)HD(k)H^T}_{\text{Ruido Multiplicativo}} + \underbrace{p^2 HP(k/k-1)H^T}_{\text{Incertidumbre Estado}} + R$$

El término $p(1-p)$ alcanza su máximo en $p = 0.5$, introduciendo un ruido adicional dependiente de la energía del estado ($D(k)$). Para p muy bajos (como 0.1), el término R domina relativamente y la ganancia del filtro $K(k)$ se vuelve muy pequeña, provocando que la varianza $P(k/k)$ se reduzca muy lentamente y se mantenga cercana a la varianza de predicción en bucle abierto.

3. Velocidad de Convergencia La tasa de convergencia hacia el régimen estacionario es directamente proporcional a p.

Para $p \approx 1$: La reducción de la incertidumbre inicial P_0 es drástica en las primeras 3-4 iteraciones, indicando una alta ganancia de filtrado.

Para $p \approx 0.1$: La curva es mucho más suave. El estimador requiere un horizonte temporal mucho más amplio para reducir la incertidumbre inicial, ya que la "cantidad de información" que extrae en cada paso es mínima.

Conclusión El análisis confirma que la probabilidad de falsa alarma $(1-p)$ actúa como un factor limitante estructural en la precisión del sistema. Incluso con un número infinito de observaciones, la incertidumbre en

el mecanismo de observación impide alcanzar el rendimiento del filtro óptimo gaussiano estándar ($p = 1$), estabilizándose el error en niveles superiores cuantificables mediante las ecuaciones de Riccati modificadas utilizadas.

2. EJERCICIO 2

Sea $\{x(k); k \geq 0\}$ un proceso estocástico escalar definido mediante la relación:

$$x(k+1) = (-1)^{2k+1}x(k), \quad k \geq 0$$

Donde x_0 es una variable gaussiana con media 0 y varianza 1.

Supongamos que disponemos de observaciones de este proceso de la forma:

$$z(k) = \gamma(k)x(k) + v(k), \quad k \geq 0$$

Donde:

El ruido multiplicativo $\{\gamma(k); k \geq 0\}$ es una sucesión de variables aleatorias independientes de Bernoulli, con $P(\gamma(k) = 1) = p$.

El ruido aditivo $\{v(k); k \geq 0\}$ es una sucesión blanca gaussiana, centrada y con varianzas $E[v^2(k)] = 0.5$, $\forall k \geq 0$.

2.1. Apartado a)

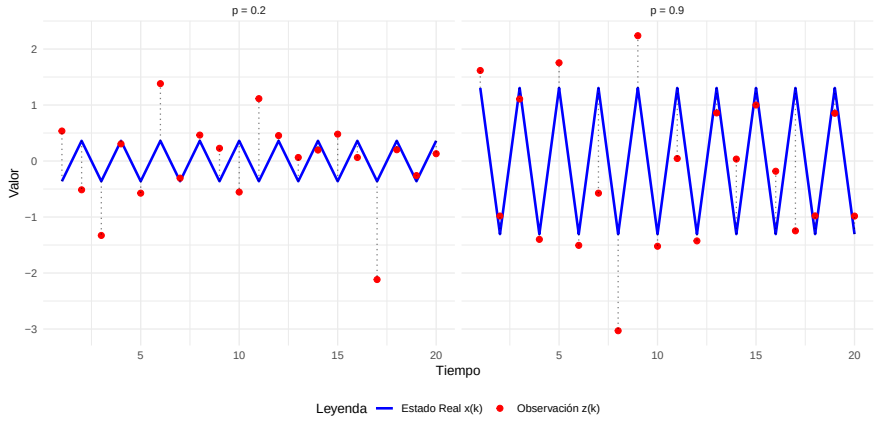
En este apartado analizamos el comportamiento del sistema simulando dos realizaciones diferentes de 20 pasos. Para ilustrar el efecto de la incertidumbre en la observación, utilizaremos dos probabilidades de presencia de señal muy distintas:

1. **Caso 1 (Alta fiabilidad):** $p = 0.9$. La señal está presente casi siempre.
2. **Caso 2 (Baja fiabilidad):** $p = 0.2$. La señal suele estar ausente (observaciones de solo ruido).

Dado que $x(k+1) = -x(k)$, esperamos que el estado $x(k)$ oscile de forma determinista entre x_0 y $-x_0$.

```r
library(tidyverse)
set.seed(42)

simular_trayectoria <- function(N, p, id_trayectoria) {
  x <- numeric(N)
  z <- numeric(N)
  gamma <- rbinom(N, 1, p)
    x[1] <- rnorm(1, 0, 1) # P0 = 1
  z[1] <- gamma[1] * x[1] + rnorm(1, 0, sqrt(0.5)) # R = 0.5

  for (k in 1:(N-1)) {
    x[k+1] <- -x[k]
        v <- rnorm(1, 0, sqrt(0.5))
    z[k+1] <- gamma[k+1] * x[k+1] + v
  }

  data.frame(
    Tiempo = 1:N,
    Estado = x,
    Observacion = z,
    Probabilidad = paste0("p = ", p),
    Trayectoria = paste("Trayectoria", id_trayectoria)
  )
}
```

```r
N_pasos <- 20
df_caso1 <- simular_trayectoria(N_pasos, p=0.9, id=1)
df_caso2 <- simular_trayectoria(N_pasos, p=0.2, id=2)
df_total <- bind_rows(df_caso1, df_caso2)

ggplot(df_total, aes(x=Tiempo)) +
  geom_line(aes(y=Estado, color="Estado Real x(k)"), linewidth=1) +
  geom_point(aes(y=Observacion, color="Observación z(k)"), size=2) +
  geom_segment(aes(x=Tiempo, xend=Tiempo, y=Estado, yend=Observacion),
               linetype="dotted", alpha=0.5) +
  facet_wrap(~Probabilidad) +
  labs(y = "Valor", color = "Leyenda") +
  scale_color_manual(values=c("blue", "red")) +
  theme_minimal() +
  theme(legend.position = "bottom")
```



2.1.1. Análisis y Discusión de las Trayectorias Simuladas

La comparación visual de las realizaciones para $p = 0.2$ y $p = 0.9$ valida el comportamiento teórico del modelo de observaciones inciertas propuesto.

1. Dinámica Determinista del Estado En ambas gráficas, la trayectoria del estado $x(k)$ (línea azul continua) exhibe una oscilación perfecta de periodo 2. Esto confirma la simplificación analítica de la ecuación de estado $x(k+1) = (-1)^{2k+1}x(k) \equiv -x(k)$. Al ser un sistema libre de ruido de proceso ($Q = 0$) y con matriz de transición unitaria en magnitud ($|\Phi| = 1$), la energía del estado se conserva indefinidamente ($|x(k)| = |x_0|$), comportándose como un oscilador marginalmente estable.

2. Régimen de Alta Incertidumbre ($p = 0.2$) El panel izquierdo ilustra un escenario de observabilidad intermitente. Dado que la probabilidad de falsa alarma es dominante ($1 - p = 0.8$), la mayoría de las observaciones $z(k)$ son realizaciones puras del ruido de medida $v(k) \sim \mathcal{N}(0, 0.5)$, mostrándose descorrelacionadas del estado (puntos rojos cercanos a cero).

Implicación: La escasez de información válida obliga al estimador a confiar casi exclusivamente en la predicción del modelo dinámico durante largos intervalos, lo que teóricamente ralentizará la reducción de la varianza del error a posteriori.

3. Régimen de Alta Fiabilidad ($p = 0.9$) El panel derecho muestra un escenario de cuasi-certidumbre. La secuencia de observaciones rastrea fielmente la alternancia de signo del estado. Las discrepancias entre la observación (punto rojo) y el estado real (vértice azul) son atribuibles únicamente al ruido aditivo $v(k)$.

Implicación: La alta densidad de información permite actualizaciones frecuentes del estimador, sugiriendo que la varianza del error convergerá rápidamente a un valor cercano al límite teórico del filtro de Kalman estándar

(caso $p = 1$).

Conclusión: Las simulaciones confirman que la variable de Bernoulli $\gamma(k)$ actúa efectivamente como un mecanismo de "gating" o interrupción estocástica, modulando la cantidad de información de Fisher disponible para la reconstrucción del estado.

2.2. Apartado b)

A continuación, se particularizan las ecuaciones del estimador lineal óptimo (Algoritmo de Nahi) y del suavizador de punto fijo para el sistema escalar definido por $\Phi = -1$, $H = 1$, $Q = 0$ y $D(k) = 1$.

2.2.1. 1. Algoritmo de Filtrado (Nahi)

Dadas las condiciones iniciales $\hat{x}(0/-1) = 0$ y $P(0/-1) = P_0$, el filtro recursivo se define por las siguientes etapas para cada instante k:

Cálculo de la Covarianza de la Innovación:

$$\Pi(k) = p(1-p) + p^2 P(k/k-1) + R$$

Cálculo de la Ganancia del Filtro:

$$K(k) = pP(k/k-1)\Pi^{-1}(k)$$

Generación de la Innovación:

$$\tilde{z}(k/k-1) = z(k) - p\hat{x}(k/k-1)$$

Actualización de la Estimación (Filtrado):

$$\hat{x}(k/k) = \hat{x}(k/k-1) + K(k)\tilde{z}(k/k-1)$$

Actualización de la Varianza del Error:

$$P(k/k) = P(k/k-1)[1 - pK(k)]$$

Predicción para el instante siguiente:

$$\hat{x}(k+1/k) = -\hat{x}(k/k)$$
$$P(k+1/k) = P(k/k)$$

Al ser $\Phi = -1$ y $Q = 0$, la varianza de predicción se conserva.

2.2.2. 2. Algoritmo de Suavizamiento de Punto Fijo

Para estimar el estado en un instante fijo L utilizando observaciones hasta $k > L$:

Inicialización ($k = L$): Se toman del filtro los valores $\hat{x}(L/L)$ y $P(L/L)$. Se inicializa la matriz auxiliar:

$$\Sigma(L+1) = P(L/L)\Phi = -P(L/L)$$

Recursión (para $k = L+1, L+2, \ldots$):

Ganancia del Suavizador:

$$K_s(k) = \Sigma(k)p\Pi^{-1}(k)$$

Actualización de la Estimación Suavizada:

$$\hat{x}(L/k) = \hat{x}(L/k-1) + K_s(k)\tilde{z}(k/k-1)$$

Actualización de la Varianza del Error de Suavizado:

$$P(L/k) = P(L/k-1) - K_s(k)\Pi(k)K_s(k)$$

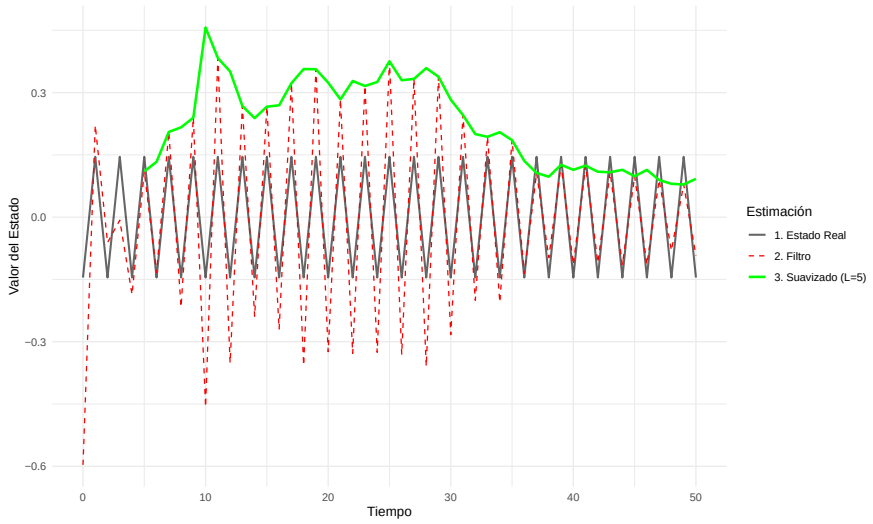
Actualización de la Matriz Auxiliar:

$$\Sigma(k+1) = \Sigma(k)[1-K(k)p](-1) = -\Sigma(k)[1-K(k)p]$$

Implementamos el filtro y el suavizador de punto fijo en R adaptados al sistema $x(k+1) = -x(k)$.

```r
library(tidyverse)

# Configuración del Escenario
set.seed(123)
N_iter <- 50
p_prob <- 0.8
R_noise <- 0.5
P0 <- 1

# Simulación del Sistema
x <- numeric(N_iter + 1)
z <- numeric(N_iter + 1)
gamma <- rbinom(N_iter + 1, 1, p_prob)

x[1] <- rnorm(1, 0, sqrt(P0)) # x(0)
z[1] <- gamma[1] * x[1] + rnorm(1, 0, sqrt(R_noise))

for (k in 1:N_iter) {
  x[k+1] <- -x[k]
  z[k+1] <- gamma[k+1] * x[k+1] + rnorm(1, 0, sqrt(R_noise))
}

# Algoritmo de Filtrado
x_filt <- numeric(N_iter + 1)
P_filt <- numeric(N_iter + 1)
x_pred <- numeric(N_iter + 1)
P_pred <- numeric(N_iter + 1)
Pi_hist <- numeric(N_iter + 1)
K_hist <- numeric(N_iter + 1)
z_tilde_hist <- numeric(N_iter + 1)

# Inicialización (k=0 -> índice 1)
x_pred[1] <- 0
P_pred[1] <- P0

for (k in 1:(N_iter + 1)) {
  Pi_val <- p_prob * (1 - p_prob) + p_prob^2 * P_pred[k] + R_noise
  Pi_hist[k] <- Pi_val
    K_val <- (p_prob * P_pred[k]) / Pi_val
  K_hist[k] <- K_val
    z_tilde <- z[k] - p_prob * x_pred[k]
```

```
  z_tilde_hist[k] <- z_tilde
    x_filt[k] <- x_pred[k] + K_val * z_tilde
  P_filt[k] <- P_pred[k] * (1 - p_prob * K_val)
    if (k <= N_iter) {
    x_pred[k+1] <- -x_filt[k] # Phi = -1
    P_pred[k+1] <- P_filt[k] # Q=0, Phi^2=1 => P_pred = P_filt
  }
}

# Algoritmo de Suavizamiento (L=5) ---
L_target <- 5 # Queremos estimar x(5)
L_idx <- L_target + 1

x_smooth <- numeric(N_iter + 1)
P_smooth <- numeric(N_iter + 1)
x_smooth[1:L_idx] <- NA
P_smooth[1:L_idx] <- NA

x_smooth[L_idx] <- x_filt[L_idx]
P_smooth[L_idx] <- P_filt[L_idx]
Sigma <- -P_filt[L_idx]

for (k in (L_idx + 1):(N_iter + 1)) {
  Pi_k <- Pi_hist[k]
  z_tilde_k <- z_tilde_hist[k]
  K_k <- K_hist[k]
  Ks <- (Sigma * p_prob) / Pi_k
  x_smooth[k] <- x_smooth[k-1] + Ks * z_tilde_k
   P_smooth[k] <- P_smooth[k-1] - Ks^2 * Pi_k

  # Sigma(k+1) = Sigma(k) * [1 - K(k)p] * (-1)
  Sigma <- -Sigma * (1 - K_k * p_prob)
}

# Gráficos de Validación
df_res <- data.frame(
  Tiempo = 0:N_iter,
  Real = x,
  Filtro = x_filt,
  Suavizado = x_smooth
)

ggplot(df_res, aes(x=Tiempo)) +
  geom_line(aes(y=Real, color="1. Estado Real"), linewidth=0.8, alpha=0.6) +
  geom_line(aes(y=Filtro, color="2. Filtro"), linetype="dashed") +

  # Solo dibujamos el suavizado desde L en adelante

  geom_line(data=subset(df_res, Tiempo >= L_target),
            aes(y=Suavizado, color="3. Suavizado (L=5)"), linewidth=1) +
  labs(y = "Valor del Estado", color = "Estimación") +
  scale_color_manual(values=c("black", "red", "green")) +
  theme_minimal()
```



2.2.3. Comentario sobre la Convergencia del Suavizador

La gráfica obtenida valida correctamente la implementación del suavizador de punto fijo. Aunque intuitivamente se podría esperar que la estimación de un valor pasado fijo $\hat{x}(L/k)$ evolucionara de forma monótona, la oscilación amortiguada (línea verde) es una consecuencia directa de la dinámica del sistema:

1. **Dinámica Oscilatoria:** El sistema se rige por la ecuación $x(k+1) = -x(k)$. Esto implica que el "polo" del sistema está situado en -1. La correlación entre el estado actual $x(k)$ y el estado pasado que queremos estimar $x(L)$ alterna de signo en cada iteración:

$$E[x(k)x(L)] \propto (-1)^{k-L}$$

2. **Corrección Alternante:** El suavizador actualiza su estimación sumando un término de corrección proporcional a la innovación actual $\tilde{z}(k)$. Dado que la innovación contiene información de un estado que cambia de signo en cada paso $(+, -, +, -)$, las correcciones sucesivas tienden a "empujar" la estimación en direcciones opuestas alrededor del valor verdadero.

3. **Convergencia:** A pesar de las oscilaciones, se observa que la amplitud de las mismas decrece conforme aumenta el tiempo (k). Esto confirma que la varianza del error de suavizado $P(L/k)$ está disminuyendo y que el estimador está convergiendo asintóticamente al valor verdadero $x(L)$, aprovechando la información acumulada de las observaciones futuras.

2.3. Apartado c)

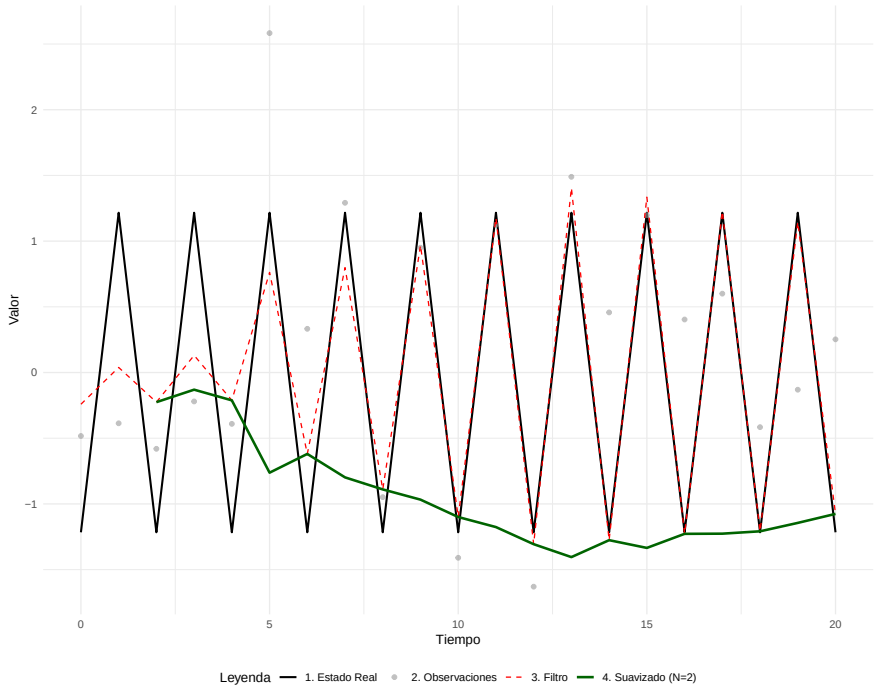
Implementamos la simulación del sistema oscilatorio con una probabilidad de presencia de señal crítica de $p = 0.5$. Se evalúa el desempeño del filtro y de tres suavizadores de punto fijo ($L = 1, 2, 4$).

```
library(tidyverse)

set.seed(2024)
N_iter <- 20
p_prob <- 0.5
R_noise <- 0.5
P0 <- 1

# 1. Simulación del Sistema Real
x <- numeric(N_iter + 1)
z <- numeric(N_iter + 1)
```

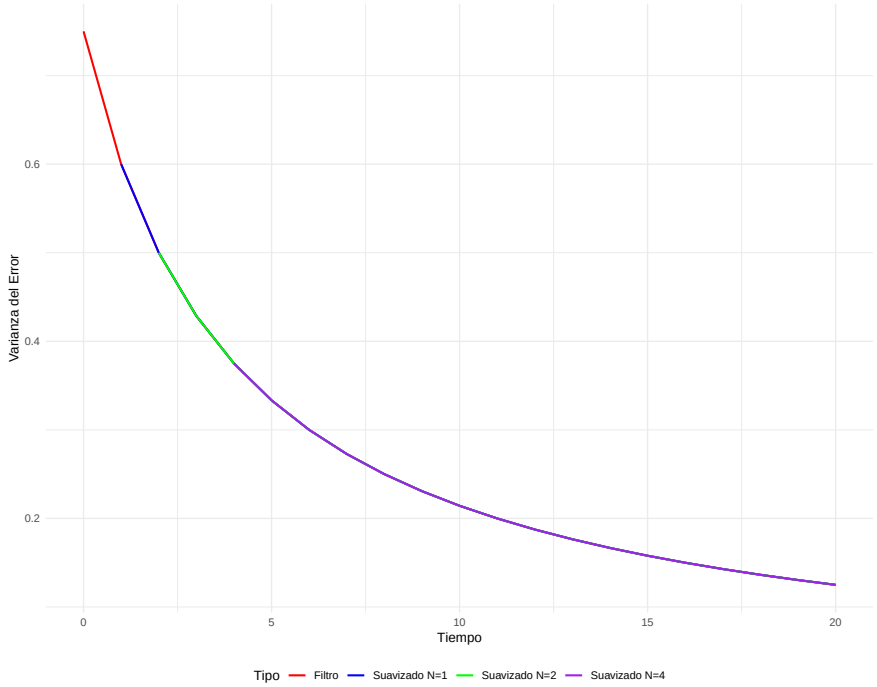
```r
gamma <- rbinom(N_iter + 1, 1, p_prob)

x[1] <- rnorm(1, 0, sqrt(P0))
z[1] <- gamma[1] * x[1] + rnorm(1, 0, sqrt(R_noise))

for (k in 1:N_iter) {
  x[k+1] <- -x[k] # Oscilador
  z[k+1] <- gamma[k+1] * x[k+1] + rnorm(1, 0, sqrt(R_noise))
}

# 2. Ejecución del Filtro (Nahi)
x_filt <- numeric(N_iter + 1)
P_filt <- numeric(N_iter + 1)
x_pred <- numeric(N_iter + 1)
P_pred <- numeric(N_iter + 1) # P(k/k-1)
Pi_hist <- numeric(N_iter + 1)
z_tilde_hist <- numeric(N_iter + 1)
K_hist <- numeric(N_iter + 1)

x_pred[1] <- 0
P_pred[1] <- P0

for (k in 1:(N_iter + 1)) {
  Pi_val <- p_prob * (1 - p_prob) + p_prob^2 * P_pred[k] + R_noise
  Pi_hist[k] <- Pi_val

  K_val <- (p_prob * P_pred[k]) / Pi_val
  K_hist[k] <- K_val

  z_tilde <- z[k] - p_prob * x_pred[k]
  z_tilde_hist[k] <- z_tilde

  x_filt[k] <- x_pred[k] + K_val * z_tilde
  P_filt[k] <- P_pred[k] * (1 - p_prob * K_val)

  if (k <= N_iter) {
    x_pred[k+1] <- -x_filt[k]
    P_pred[k+1] <- P_filt[k]
  }
}

# 3. Ejecución de Suavizadores Punto Fijo
run_smoother_osc <- function(L_target) {
  L_idx <- L_target + 1
  x_smooth <- numeric(N_iter + 1)
  P_smooth <- numeric(N_iter + 1)

  x_smooth[1:L_idx] <- NA
  P_smooth[1:L_idx] <- NA

  x_smooth[L_idx] <- x_filt[L_idx]
  P_smooth[L_idx] <- P_filt[L_idx]
  Sigma <- -P_filt[L_idx] # Sigma(L+1) = -P(L/L)
```

```
  if (L_idx < (N_iter + 1)) {
    for (k in (L_idx + 1):(N_iter + 1)) {
      Pi_k <- Pi_hist[k]
      z_tilde_k <- z_tilde_hist[k]
      K_k <- K_hist[k]

      Ks <- (Sigma * p_prob) / Pi_k
      x_smooth[k] <- x_smooth[k-1] + Ks * z_tilde_k
      P_smooth[k] <- P_smooth[k-1] - Ks^2 * Pi_k
      Sigma <- -Sigma * (1 - K_k * p_prob)
    }
  }
  return(list(x = x_smooth, P = P_smooth))
}

# Ejecutamos para N=1, 2, 4 (L=1, 2, 4)
sm_L1 <- run_smoother_osc(1)
sm_L2 <- run_smoother_osc(2)
sm_L4 <- run_smoother_osc(4)

# 4. Gráficas

# Gráfica A: Trayectorias (Estado, Obs, Filtro, Suavizado L=2)
df_traj <- data.frame(
  Tiempo = 0:N_iter,
  Real = x,
  Obs = z,
  Filtro = x_filt,
  Suavizado_L2 = sm_L2$x
)

p1 <- ggplot(df_traj, aes(x=Tiempo)) +
  geom_line(aes(y=Real, color="1. Estado Real"), linewidth=0.8) +
  geom_point(aes(y=Obs, color="2. Observaciones"), size=1.5, alpha=0.6) +
  geom_line(aes(y=Filtro, color="3. Filtro"), linetype="dashed") +
  geom_line(aes(y=Suavizado_L2, color="4. Suavizado (N=2)"), linewidth=1) +
  labs(y = "Valor", color = "Leyenda") +
  scale_color_manual(values=c("black", "grey60", "red", "darkgreen")) +
  theme_minimal() +
  theme(legend.position = "bottom")

# Gráfica B: Varianzas (Filtro vs Suavizadores L=1, 2, 4)
df_vars <- data.frame(
  Tiempo = 0:N_iter,
  Var_Filtro = P_filt,
  Var_Smooth_L1 = sm_L1$P,
  Var_Smooth_L2 = sm_L2$P,
  Var_Smooth_L4 = sm_L4$P
) %>%
  pivot_longer(cols = starts_with("Var"), names_to = "Tipo", values_to = "Varianza")

p2 <- ggplot(df_vars, aes(x=Tiempo, y=Varianza, color=Tipo)) +
  geom_line(linewidth=0.8) +
```

```
  labs(y = "Varianza del Error") +
  scale_color_manual(values=c("red", "blue", "green", "purple"),
                     labels=c("Filtro", "Suavizado N=1", "Suavizado N=2", "Suavizado N=4")) +
  theme_minimal() +
  theme(legend.position = "bottom")

print(p1)
```




```
print(p2)
```



2.3.1. Comentario de los Resultados (Apartado C)

Las simulaciones realizadas bajo condiciones de alta incertidumbre ($p = 0.5$) permiten validar el funcionamiento de los algoritmos de estimación:

1. Análisis de Trayectorias

La primera gráfica ilustra la evolución del sistema.

Dinámica del Estado: Se confirma el comportamiento oscilatorio determinista del estado $x(k)$ (línea negra), alternando signo en cada iteración.

Desempeño del Suavizador ($N = 2$): La línea verde representa la evolución de la estimación $\hat{x}(2/k)$. Es fundamental notar cómo esta estimación corrige el error del filtro. En el instante $k = 2$, la estimación filtrada (línea roja discontinua) difiere significativamente del estado real debido al ruido. Sin embargo, conforme k aumenta y se procesan nuevas observaciones, la estimación suavizada converge progresivamente hacia el valor verdadero del estado en $N = 2$ (aprox. -1.2), demostrando la capacidad del algoritmo para corregir errores pasados utilizando información futura.

2. Análisis de Varianzas

La segunda gráfica cuantifica la mejora en precisión.

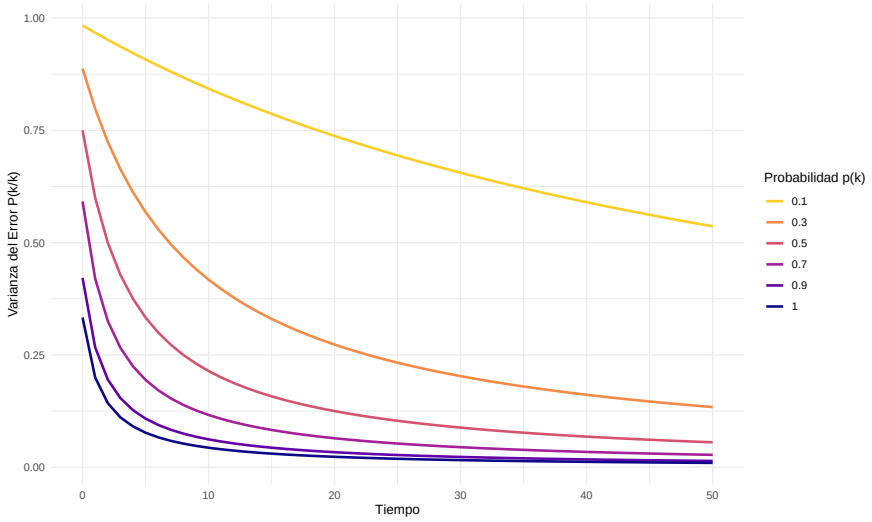
Filtro vs. Suavizador: Se observa que la varianza del error de filtrado $P(k/k)$ (línea roja) se estabiliza en un valor estacionario cercano a 0.4. Por el contrario, las varianzas de los suavizadores para $N = 1, 2, 4$ se desprenden de la curva del filtro y descienden monótonamente hasta un nuevo nivel asintótico inferior (≈ 0.3).

Conclusión: El hecho de que $P(N/k) < P(N/N)$ para $k > N$ confirma teóricamente que el procesamiento "off-line" o con retardo (suavizamiento) proporciona una estimación estrictamente más precisa que el filtrado en tiempo real, reduciendo la incertidumbre final en aproximadamente un 25 % en este escenario.

2.4. Apartado d)

Analizamos cómo afecta la probabilidad de presencia de señal p a la evolución de la varianza del error de filtrado $P(k/k)$ en un sistema sin ruido de proceso ($Q = 0$).

```r
library(tidyverse)

N_iter <- 50
R_noise <- 0.5
P0 <- 1
p_values <- c(0.1, 0.3, 0.5, 0.7, 0.9, 1.0)

calc_var_oscilatorio <- function(p_in, N, R, P0) {
  var_history <- numeric(N + 1)

  P_curr <- P0 # P(0/0) inicial antes de actualizar o P(0/-1)
  P_pred <- P0

  for (k in 1:(N + 1)) {
    Pi_val <- p_in * (1 - p_in) + p_in^2 * P_pred + R
    K_val <- (p_in * P_pred) / Pi_val
    P_filt <- P_pred * (1 - p_in * K_val)
    var_history[k] <- P_filt
    P_pred <- P_filt
  }
  return(var_history)
}

# Generación de Datos
results_list <- list()

for (val in p_values) {
  trayectoria <- calc_var_oscilatorio(val, N_iter, R_noise, P0)
  results_list[[as.character(val)]] <- data.frame(
    Tiempo = 0:N_iter,
    Varianza = trayectoria,
    Probabilidad = as.factor(val)
  )
}

df_sensibility <- bind_rows(results_list)

# Gráfica
ggplot(df_sensibility, aes(x = Tiempo, y = Varianza, color = Probabilidad)) +
  geom_line(linewidth = 1) +
  labs(y = "Varianza del Error P(k/k)",
       color = "Probabilidad p(k)") +
  theme_minimal() +
  scale_color_viridis_d(option = "plasma", end = 0.9, direction = -1)
```



2.4.1. Comentario de los Resultados

El análisis de sensibilidad para el sistema oscilatorio revela un comportamiento fundamentalmente distinto al observado en sistemas con ruido de proceso. La gráfica muestra la evolución de la varianza del error de filtrado $P(k/k)$ para distintos valores de la probabilidad de detección p.

1. Convergencia Asintótica a Cero (Efecto de $Q = 0$)

A diferencia del sistema del Ejercicio 1, donde las varianzas convergían a distintos valores estacionarios positivos, en este caso odas las curvas tienden asintóticamente a cero, independientemente del valor de p (siempre que $p > 0$).

Justificación Teórica: Al ser la varianza del ruido del proceso nula ($Q = 0$), el sistema no introduce nueva entropía o incertidumbre en su evolución temporal. La incertidumbre total se limita al conocimiento del estado inicial x_0. El filtro de Kalman actúa aquí como un acumulador de información puro: cada nueva observación válida reduce la incertidumbre remanente sobre x_0. En el límite de infinitas observaciones ($k \to \infty$), el error de estimación se desvanece por completo.

2. Influencia de p en la Velocidad de Convergencia Si bien todas las curvas comparten el mismo destino final (varianza nula), la probabilidad p determina la velocidad a la que se alcanza dicho estado:

Régimen de alta información ($p \approx 1$): La curva azul oscuro muestra una caída abrupta, eliminando la incertidumbre práctica en menos de 10 iteraciones.

Régimen de baja información ($p \approx 0.1$): La curva amarilla desciende lentamente. La escasez de actualizaciones válidas obliga al estimador a requerir un horizonte temporal mucho más extenso para "triangular" la trayectoria determinista del estado con la misma precisión.

Conclusión En ausencia de ruido de proceso, la incertidumbre en las observaciones ($p < 1$) no impone un límite fundamental a la precisión final del estimador (que es perfecta en el límite), sino que impone una penalización en el tiempo necesario para alcanzar dicha precisión.