



Descubrimiento de
conocimiento en secuencias
de valores

Luciano Sánchez
Universidad de Oviedo

Ideas generales

- Queremos **predecir los valores futuros** de las variables observables de un sistema, para simular y comparar la realidad con el modelo, en diferentes circunstancias:
 - Cuando sabemos poco acerca del problema
 - Cuando conocemos alguna propiedad o restricción
- O bien no queremos predecir las variables, sino **detectar un evento** que no sea evidente, como un cambio en la dinámica del sistema.



Queremos
adivinar el futuro

Modelos predictivos

Sabemos poco
del problema

Pure data-driven methods



Modelos lineales: Algoritmos de realización (ERA)

$$\begin{aligned}x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) + Du(k)\end{aligned}$$

$$x(0) = 0$$

$$y(0) = Du(0),$$

$$x(1) = Bu(0),$$

$$y(1) = CBu(0) + Du(1),$$

$$x(2) = ABu(0) + Bu(1),$$

$$y(2) = CABu(0) + CBu(1) + Du(2),$$

⋮

$$x(k) = \sum_{i=1}^k A^{i-1} Bu(k-i),$$

$$y(k) = \sum_{i=1}^k CA^{i-1} Bu(k-i) + Du(k)$$

Modelos lineales con observador (ERA-OKID)

$$x(k+1) = Ax(k) + Bu(k) + Gy(k) - Gy(k)$$

$$\begin{aligned}x(k+1) &= Ax(k) + Bu(k) + G(Cx(k) + Du(k)) - G(Cx(k) + Du(k)) \\ &= (A + GC)x(k) + (B + GD)u(k) - Gy(k)\end{aligned}$$

$$\begin{aligned}\bar{A} &= A + GC \\ \bar{B} &= [B + GD \quad -G] \\ v(k) &= \begin{bmatrix} u(k) \\ y(k) \end{bmatrix}\end{aligned}$$

Observador

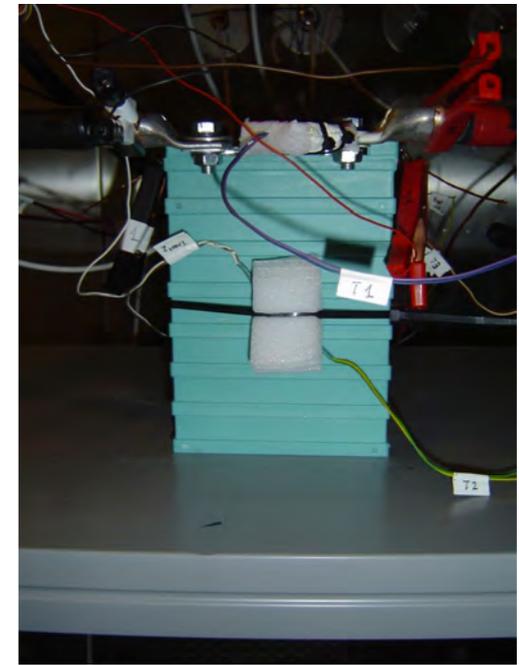
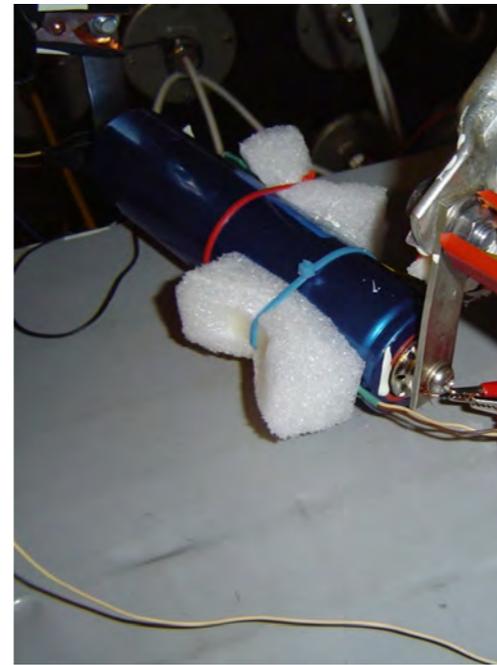
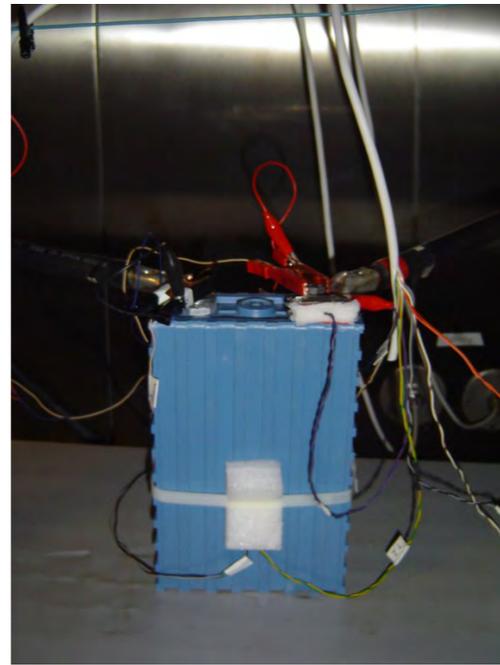
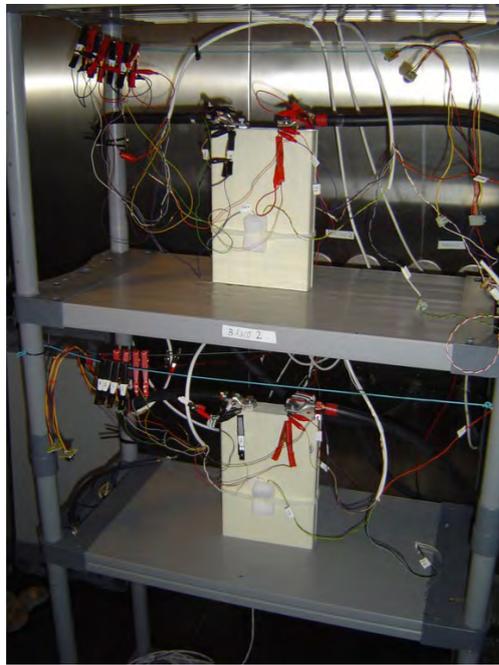


$$\begin{aligned}x(k+1) &= \bar{A}x(k) + \bar{B}v(k) \\ y(k) &= Cx(k) + Du(k)\end{aligned}$$

$$\begin{aligned}\hat{x}(k+1) &= A\hat{x}(k) + Bu(k) - G[y(k) - \hat{y}(k)] \\ \hat{y}(k) &= C\hat{x}(k) + Du(k)\end{aligned}$$

Ejemplo: Modelo de baterías recargables para automoción

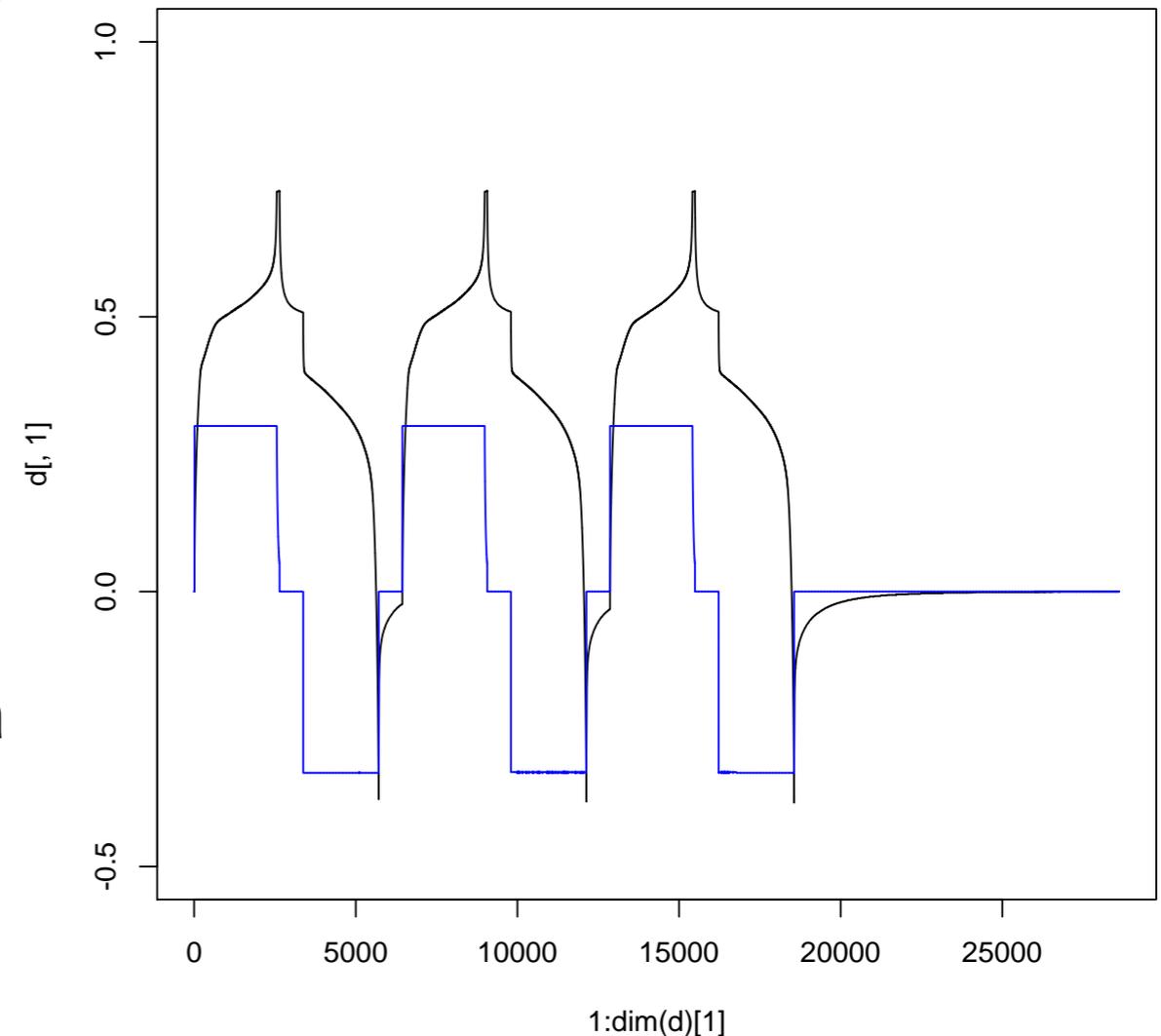
Descripción de los ensayos



Baterías modeladas. De izquierda a derecha:
HUANYU (100Ah), CALB-SE (100Ah), HEADWAY (16Ah), GBS
ENERGY (60 Ah)

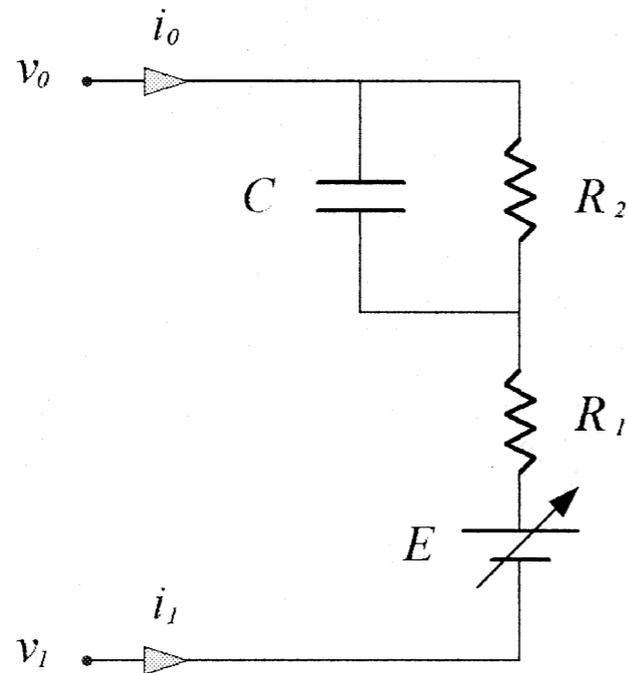
Ejemplo: Modelo de baterías recargables para automoción

- Se realizan varios ciclos de cargas y descargas completas con reposos intermedios
- La cargas se hacen a corriente constante hasta que la batería alcanza una tensión de 3.6V, y después a tensión constante
- Evolución de la tensión (negro) y la corriente de carga/descarga (azul) en un ensayo a 0.3C de una batería de LiFePO_4

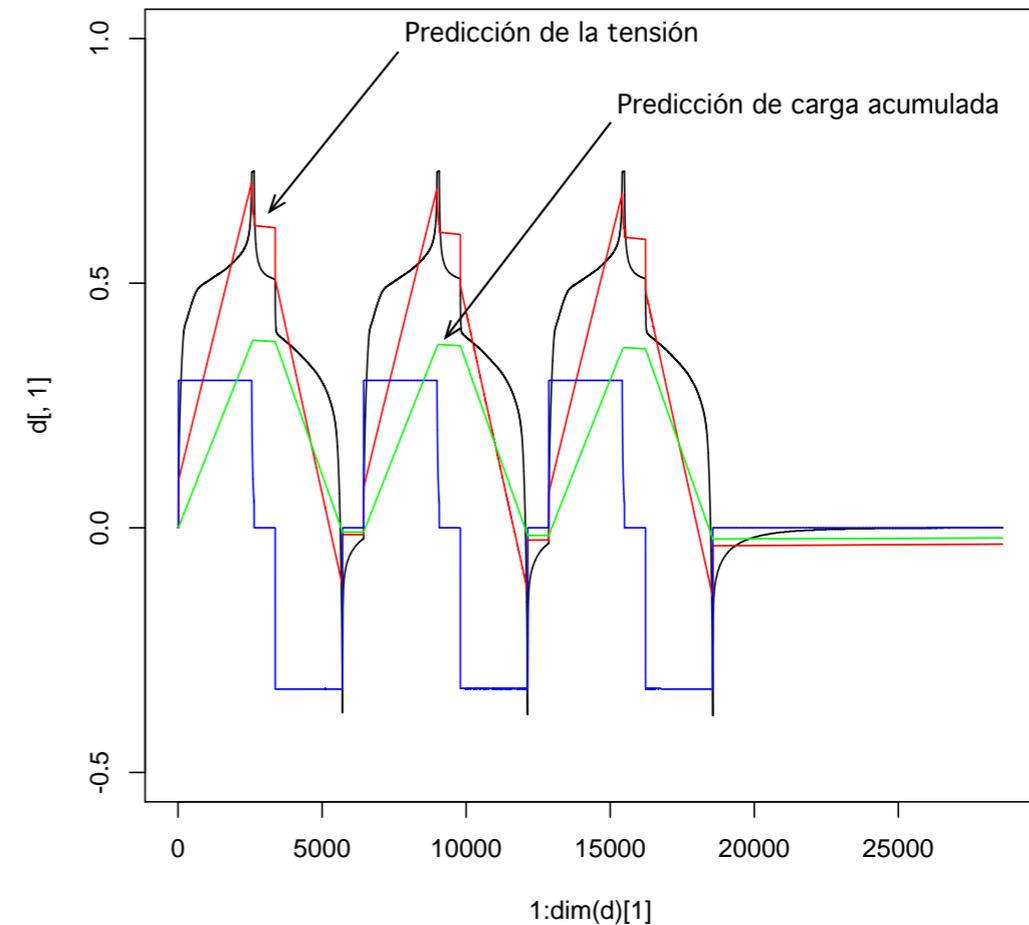


Ejemplo: Modelo de baterías recargables para automoción

Modelo lineal

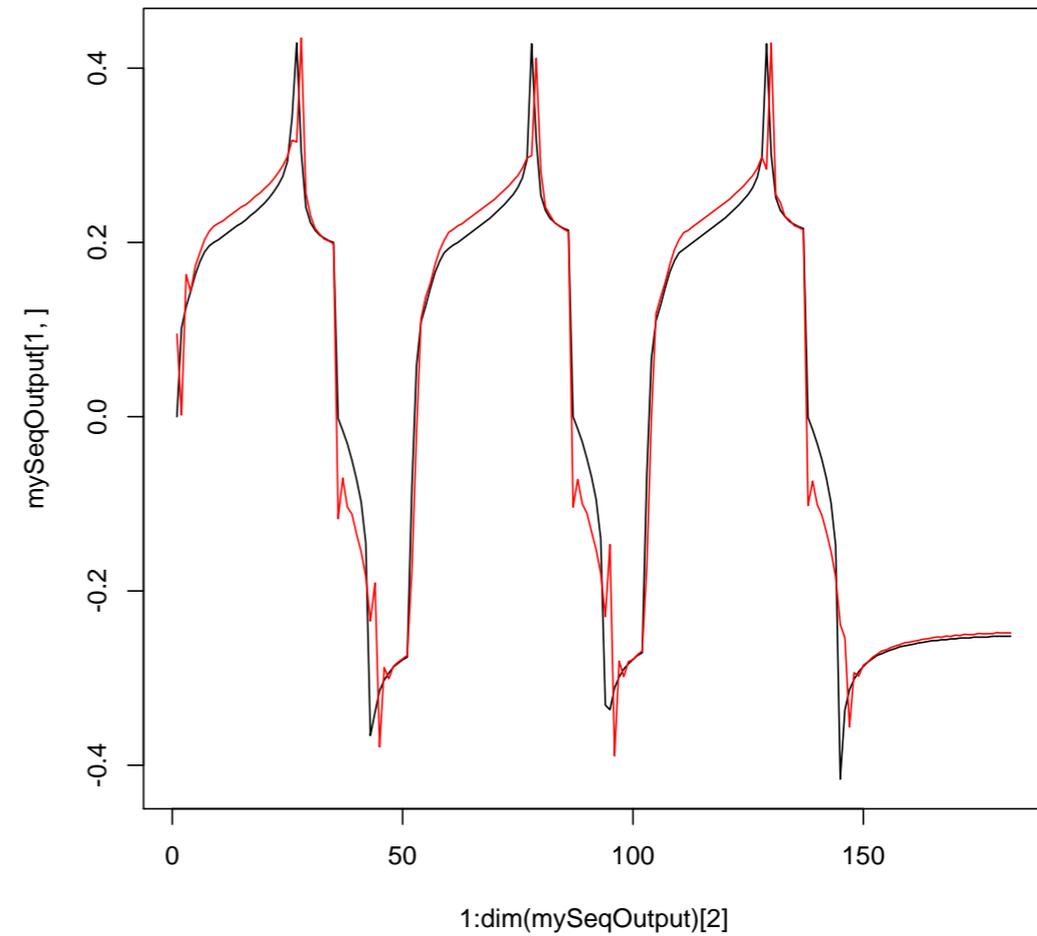
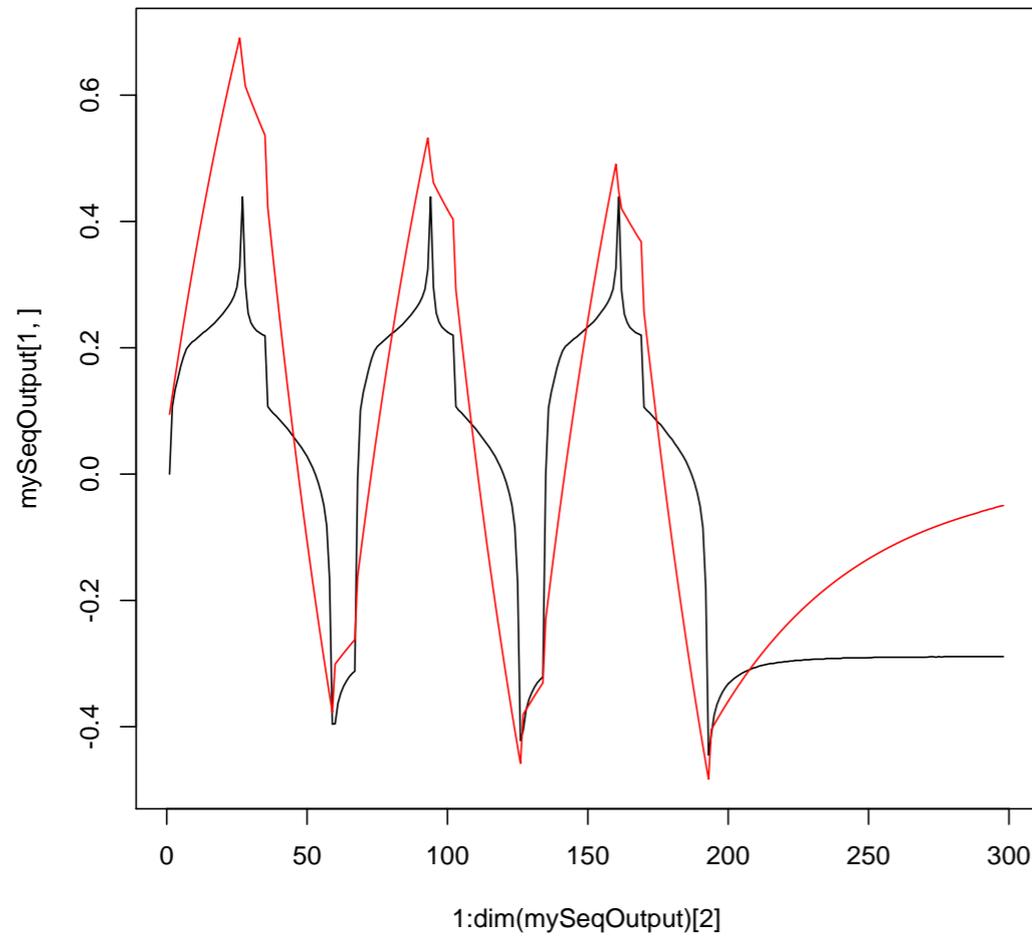


Modelo eléctrico equivalente de una batería de Li-Ion



Tres ciclos de carga y descarga a C/3 de la batería CALB-SE.

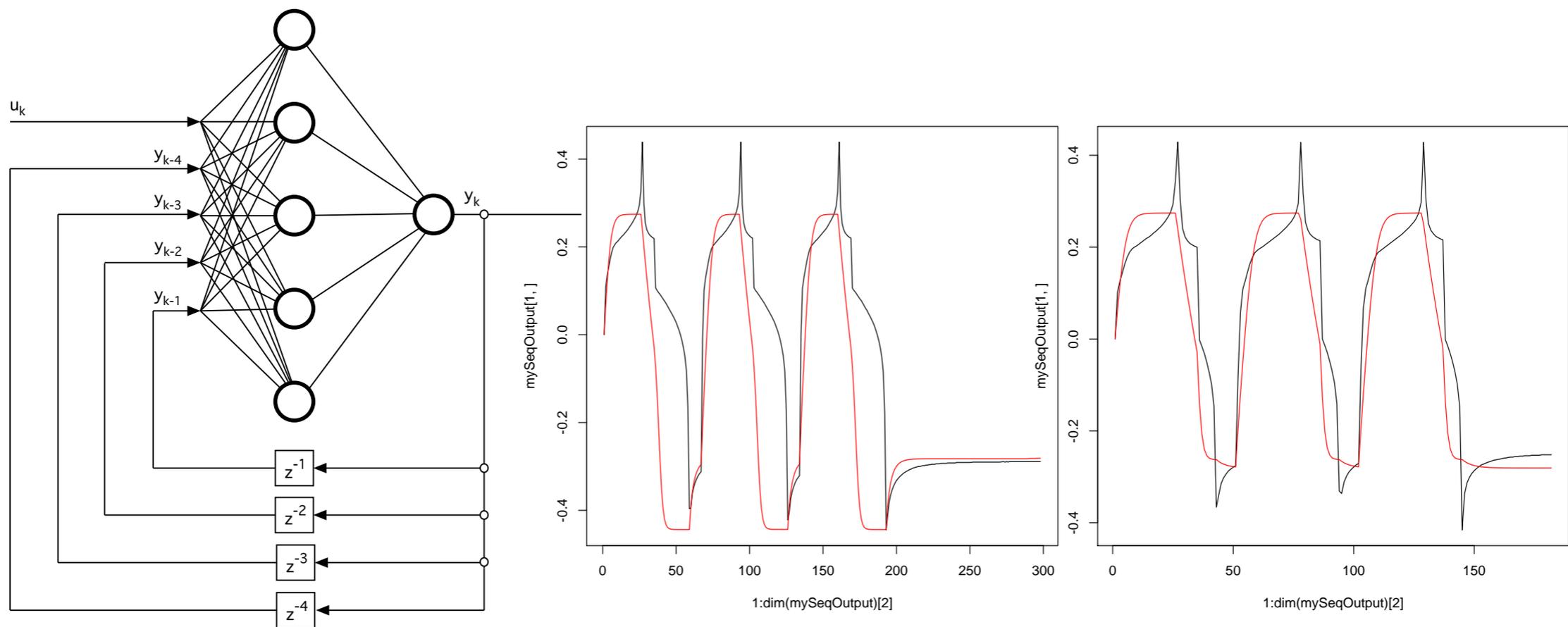
Modelos ERA y ERA/OKID para baterías



- Izquierda: Realización ERA, estado inicial conocido. Derecha: ERA/OKID

Modelos no lineales: Redes neuronales NARX

Redes neuronales

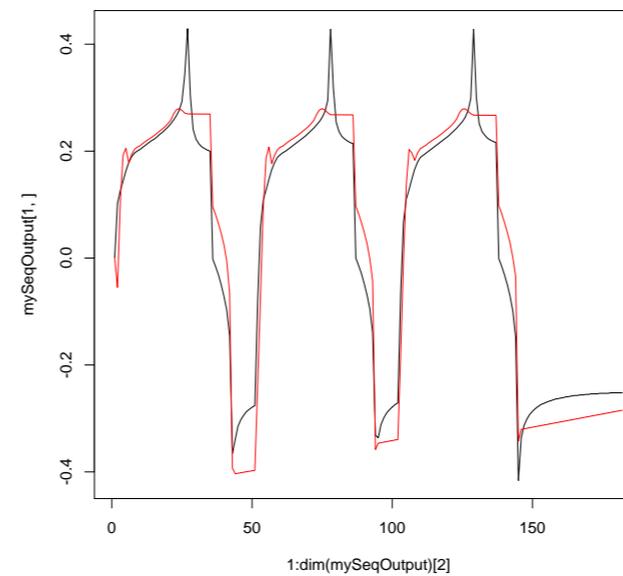
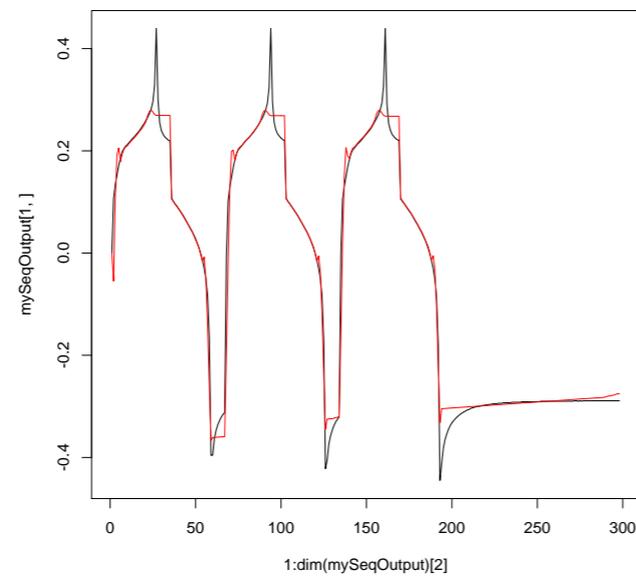
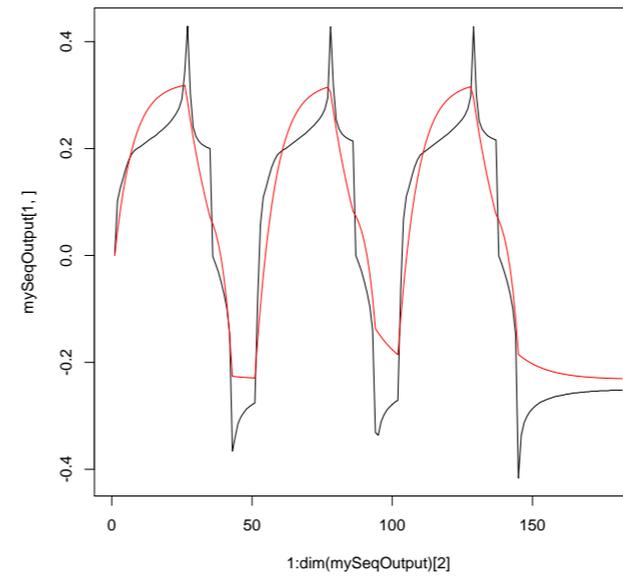
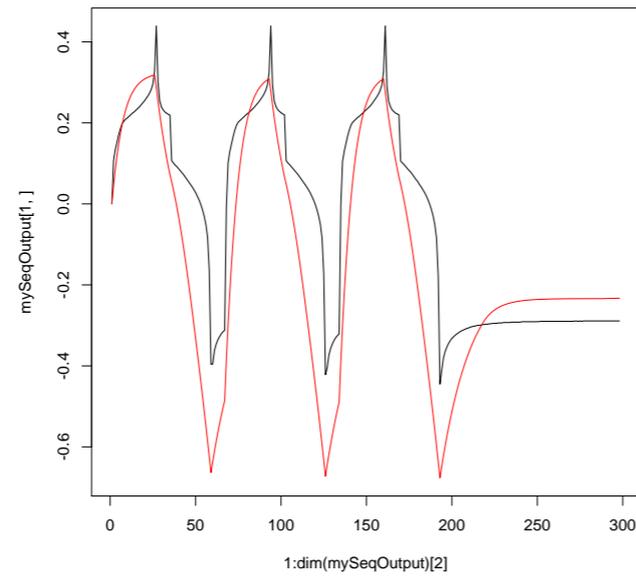


Red neuronal recurrente en topología NARX
Batería CALB-SE; entrenamiento a C3, test a C

Modelos basados en reglas o en reglas borrosas

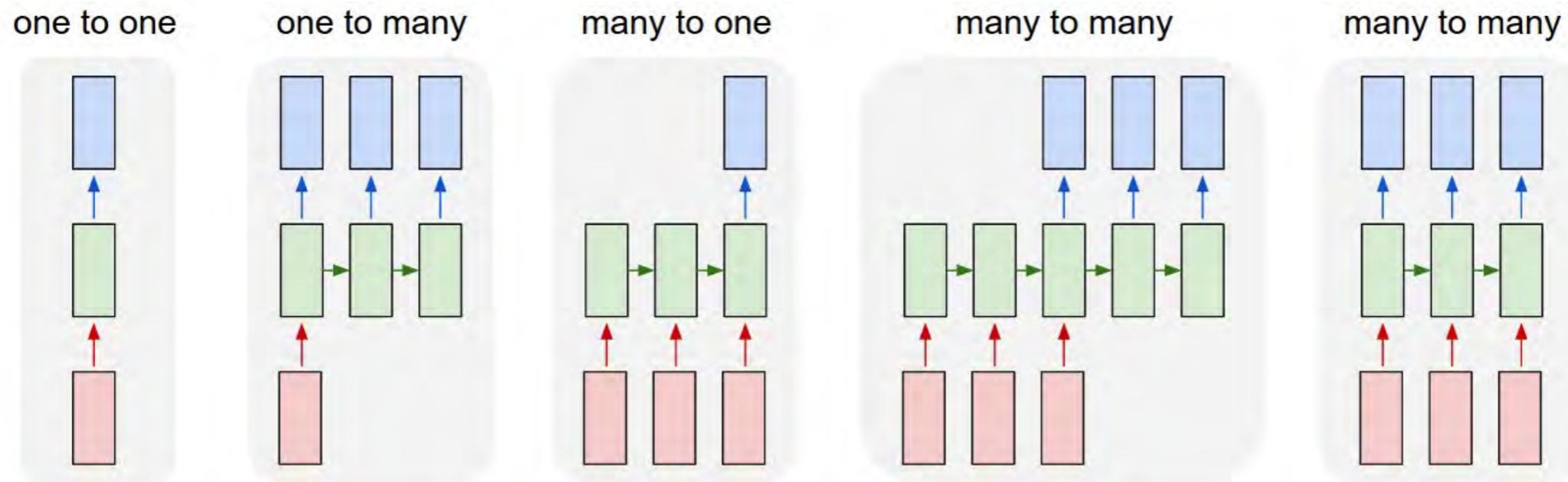
- Compuestos de reglas sí-entonces
- A veces son útiles para añadir información a mano
- A veces sirven para explicar por qué el modelo toma una decisión
- Complicados de paralelizar, más lentos que las redes neuronales
- Permiten hacer correcciones
- Pueden ser muy eficientes en problemas pequeños

Fuzzy TSK



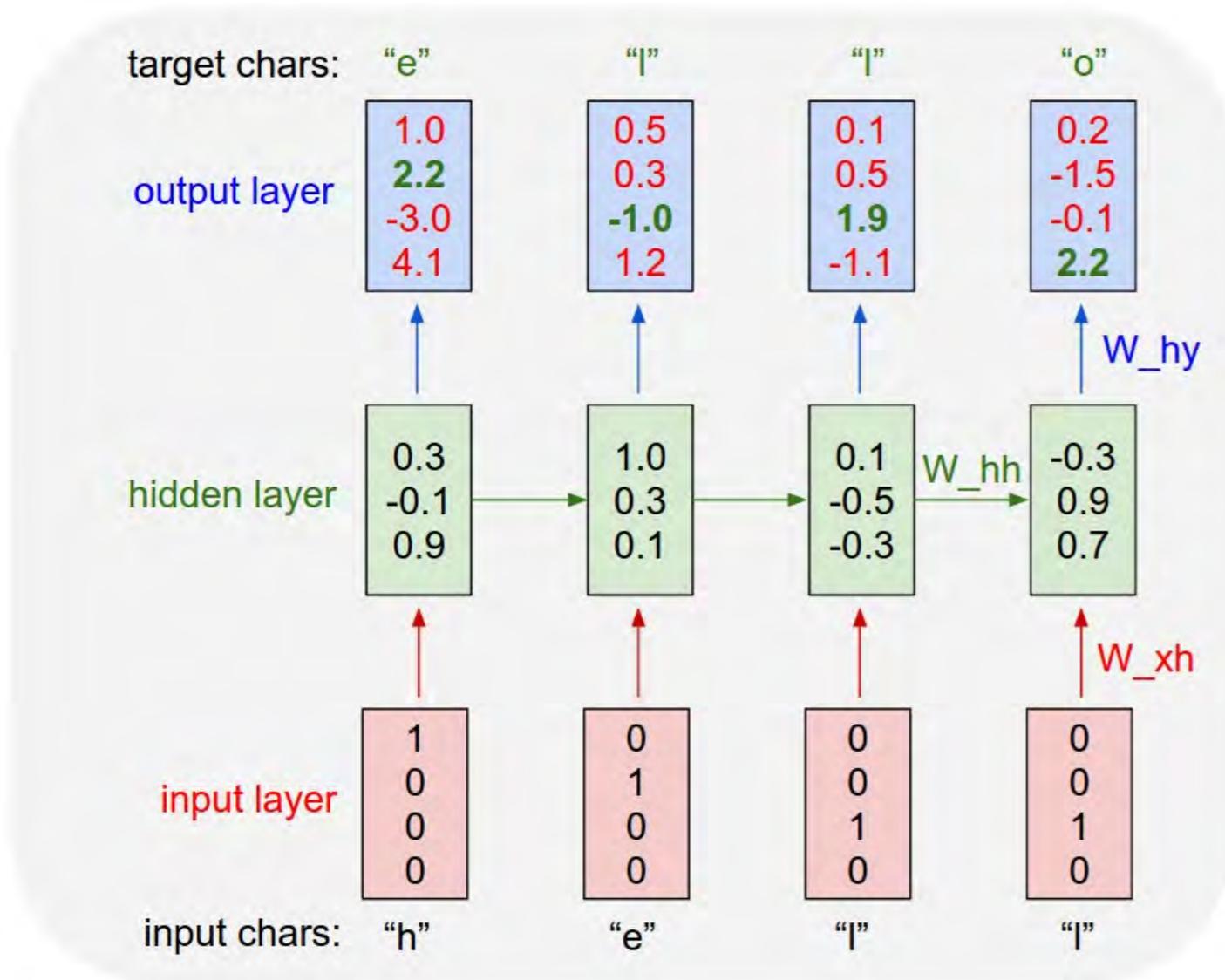
Parte superior: CALB-SE C3 y C Fuzzy TSK recurrente, 2x2 labels/partición.
Parte inferior: CALB-SE C3 y C Fuzzy TSK no recurrente, 7x2 labels/partición

Redes recurrentes: LSTM



Each rectangle is a vector and arrows represent functions (e.g. matrix multiply). Input vectors are in red, output vectors are in blue and green vectors hold the RNN's state (more on this soon). From left to right: **(1)** Vanilla mode of processing without RNN, from fixed-sized input to fixed-sized output (e.g. image classification). **(2)** Sequence output (e.g. image captioning takes an image and outputs a sentence of words). **(3)** Sequence input (e.g. sentiment analysis where a given sentence is classified as expressing positive or negative sentiment). **(4)** Sequence input and sequence output (e.g. Machine Translation: an RNN reads a sentence in English and then outputs a sentence in French). **(5)** Synced sequence input and output (e.g. video classification where we wish to label each frame of the video). Notice that in every case are no pre-specified constraints on the lengths sequences because the recurrent transformation (green) is fixed and can be applied as many times as we like

Redes recurrentes: LSTM



- La capa oculta está evolucionando permanentemente y depende del estado \mathbf{x} y de la entrada \mathbf{u}

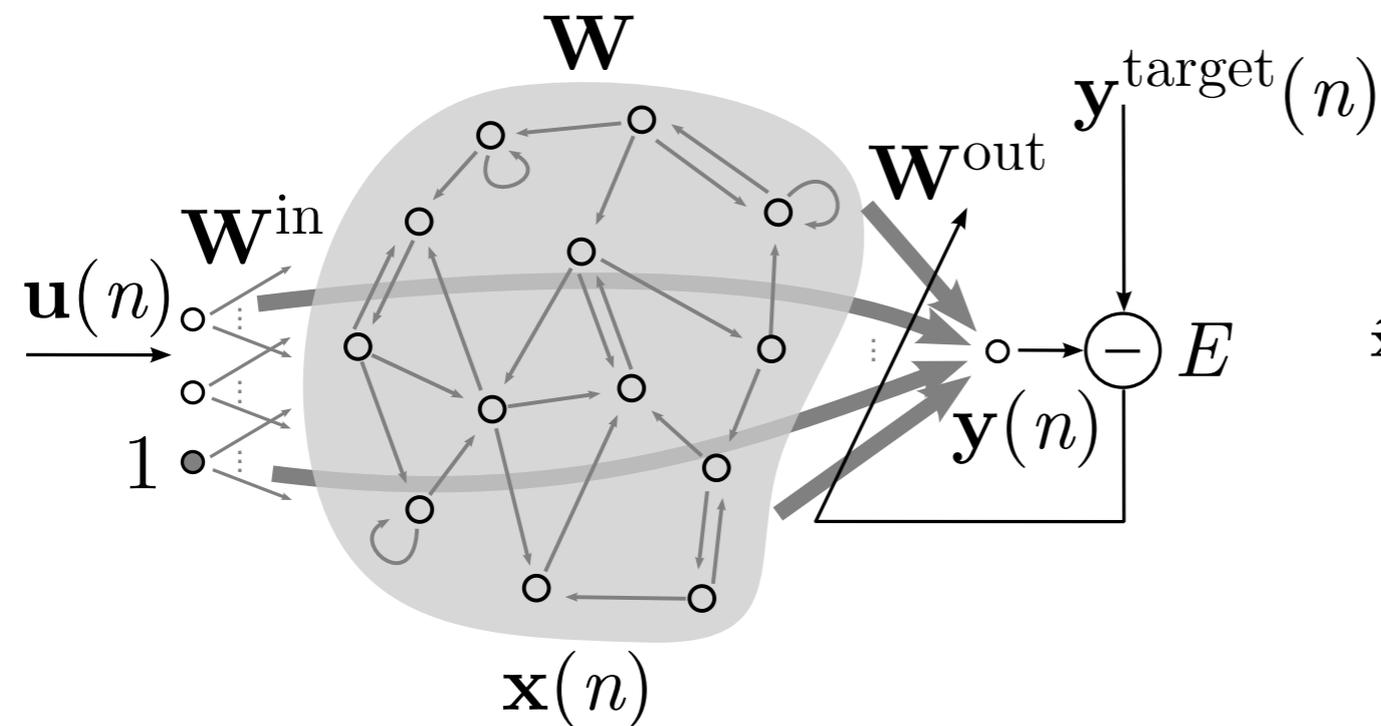
$$\tilde{\mathbf{x}}(n) = \tanh(\mathbf{W}^{\text{in}}[1; \mathbf{u}(n)] + \mathbf{W}\mathbf{x}(n-1)),$$

$$\mathbf{x}(n) = (1 - \alpha)\mathbf{x}(n-1) + \alpha\tilde{\mathbf{x}}(n),$$

- La salida depende del estado y en algunos modelos también de la entrada

$$\mathbf{y}(n) = \mathbf{W}^{\text{out}}[1; \mathbf{u}(n); \mathbf{x}(n)],$$

Redes recurrentes: ESN

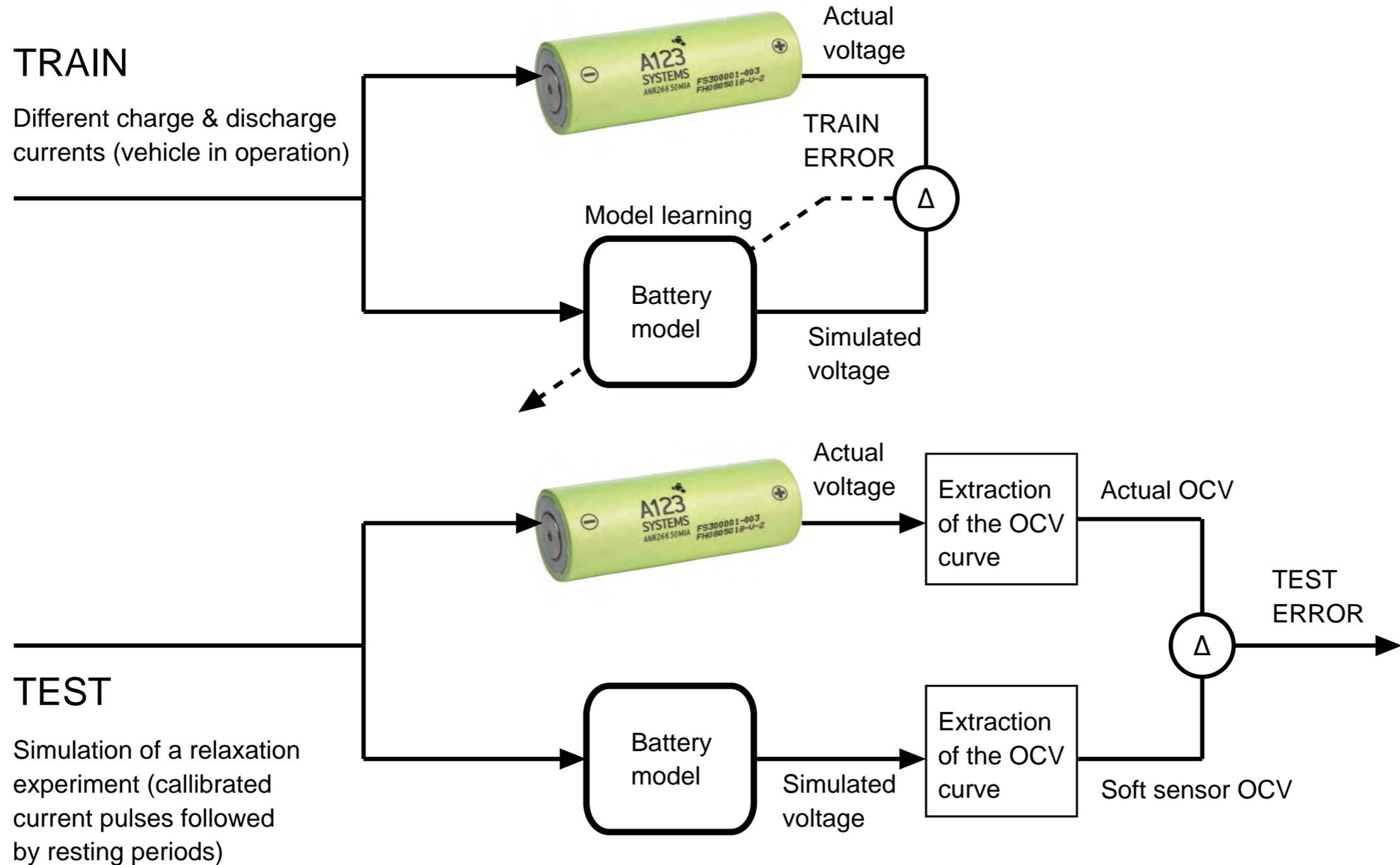


$$\begin{aligned}\tilde{\mathbf{x}}(n) &= \tanh(\mathbf{W}^{\text{in}}[1; \mathbf{u}(n)] + \mathbf{W}\mathbf{x}(n-1)), \\ \mathbf{x}(n) &= (1 - \alpha)\mathbf{x}(n-1) + \alpha\tilde{\mathbf{x}}(n), \\ \mathbf{y}(n) &= \mathbf{W}^{\text{out}}[1; \mathbf{u}(n); \mathbf{x}(n)],\end{aligned}$$

• Echo State Networks (ESNs)

1. generate a large random reservoir RNN ($\mathbf{W}^{\text{in}}, \mathbf{W}, \alpha$);
2. run it using the training input $\mathbf{u}(n)$ and collect the corresponding reservoir activation states $\mathbf{x}(n)$;
3. compute the linear readout weights \mathbf{W}^{out} from the reservoir using linear regression, minimizing the MSE between $\mathbf{y}(n)$ and $\mathbf{y}^{\text{target}}(n)$;
4. use the trained network on new input data $\mathbf{u}(n)$ computing $\mathbf{y}(n)$ by employing the trained output weights \mathbf{W}^{out} .

Ejemplo: modelo de baterías



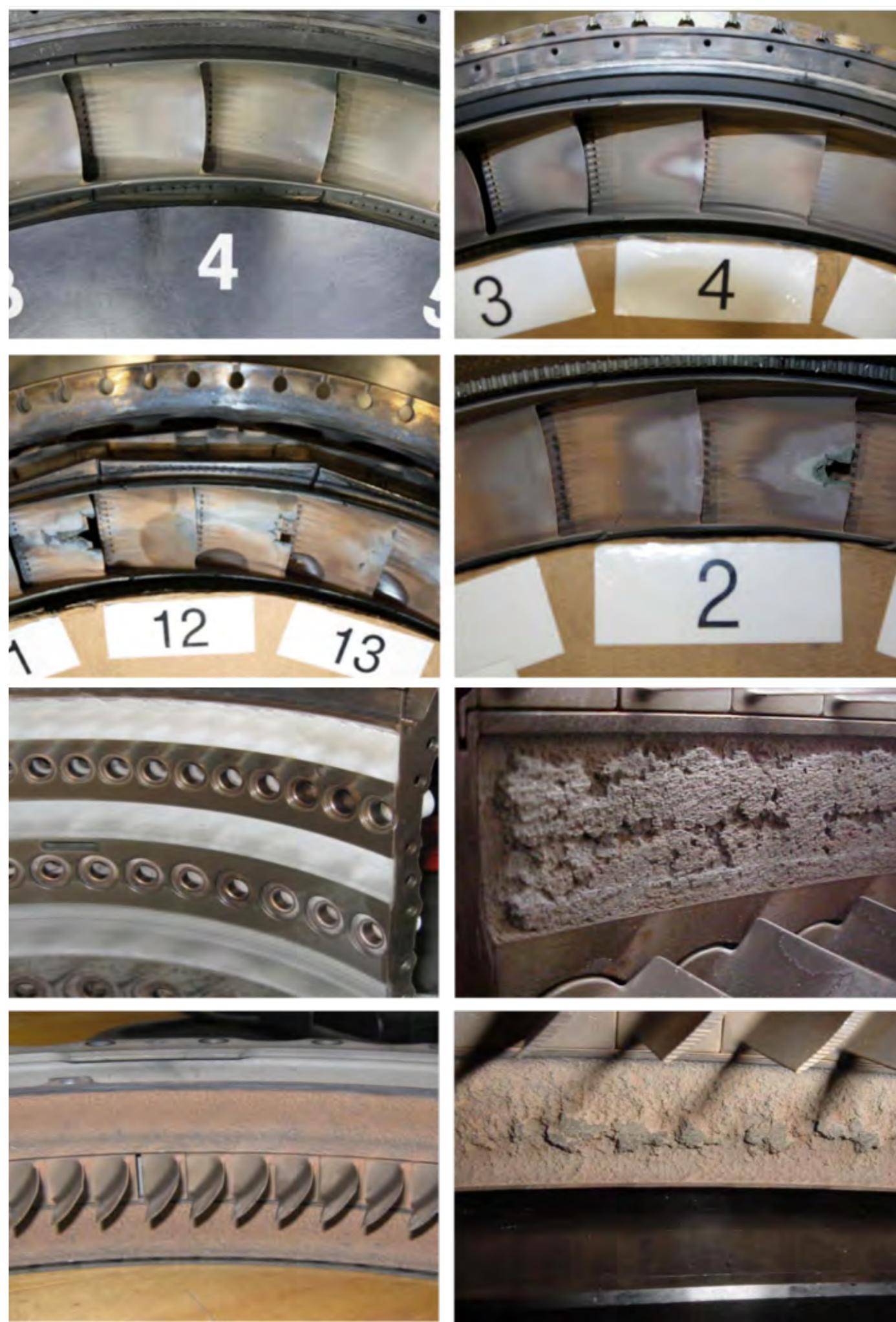
Resultados numéricos

Table 1. Influence of the charging current. Average quadratic error of OCV, obtained from RNN, LSTM, ESN, γ -ESN, ANFIS, and ARIMAX

	C25	C5	C3	C2	C1
Abu-Sharkh	0.0003	0.0094	0.0080	0.0084	0.0110
Xu	0.0006	0.0086	0.0146	0.0153	0.0073
LSTM	0.0077	0.0301	0.0070	0.0066	0.0064
LSTM-dropout	0.0100	0.0295	0.0067	0.0083	0.0093
ESN	0.0056	0.0326	0.0083	0.0106	0.0083
γ -ESN	0.1553	0.0854	0.0279	0.0132	0.0212
ANFIS	0.2026	0.2016	0.0731	0.0595	0.0334
ARIMAX(2,1)	0.0127	0.0391	1.0153	0.0165	0.0202

Conocemos parte del problema

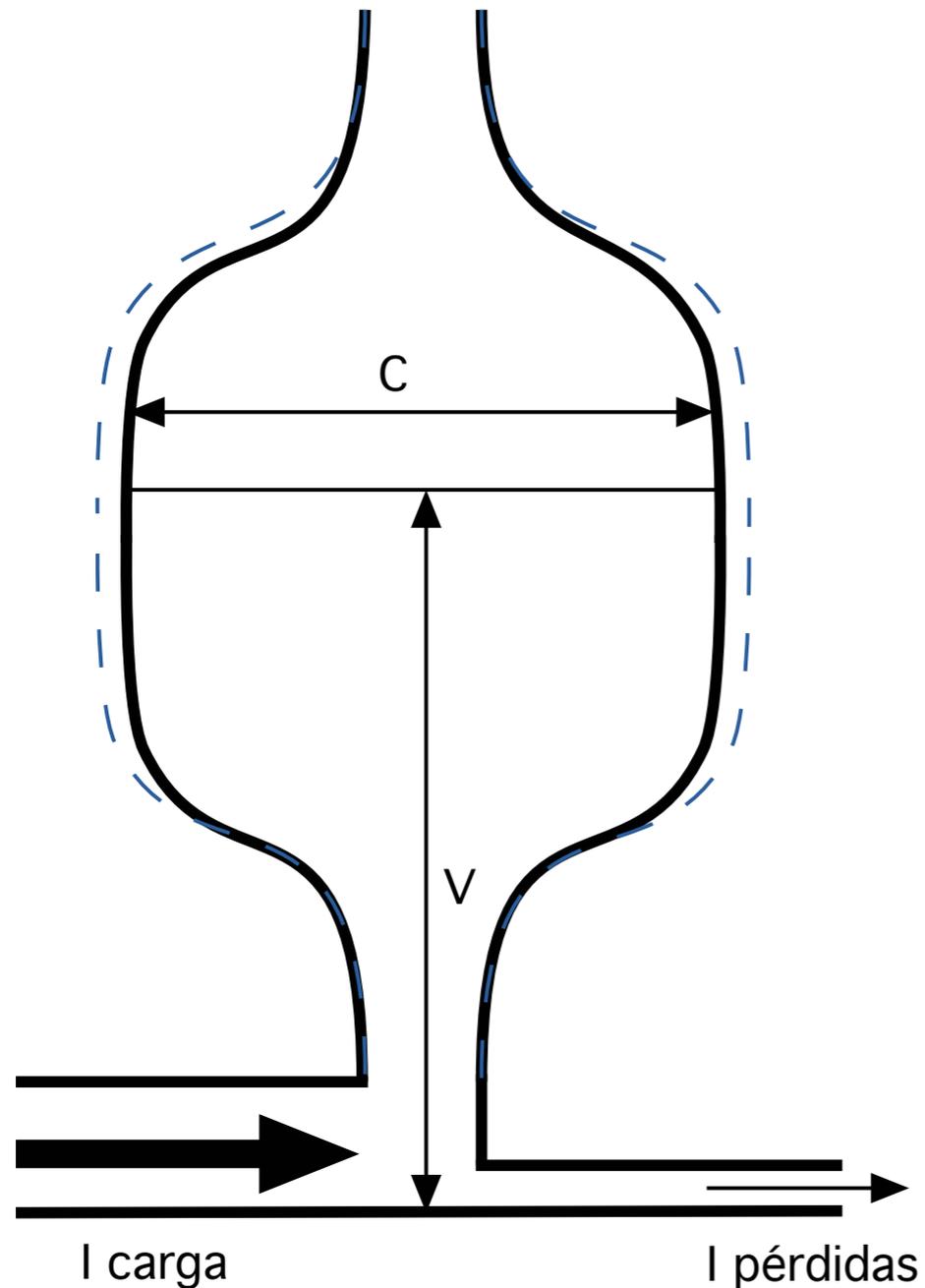
Integración de conocimiento acerca del dominio del problema y métodos guiados por datos



Modelos semifísicos

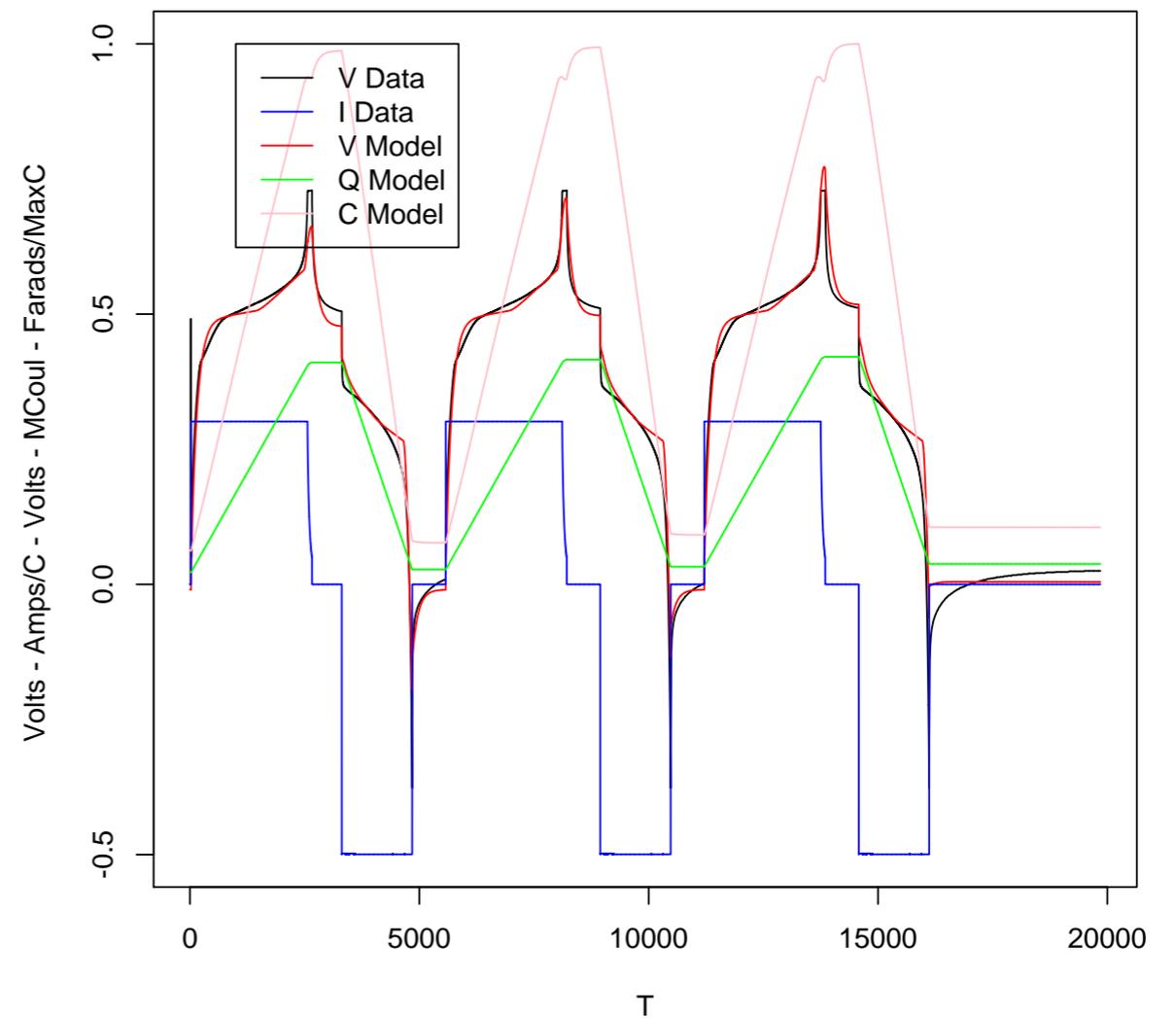
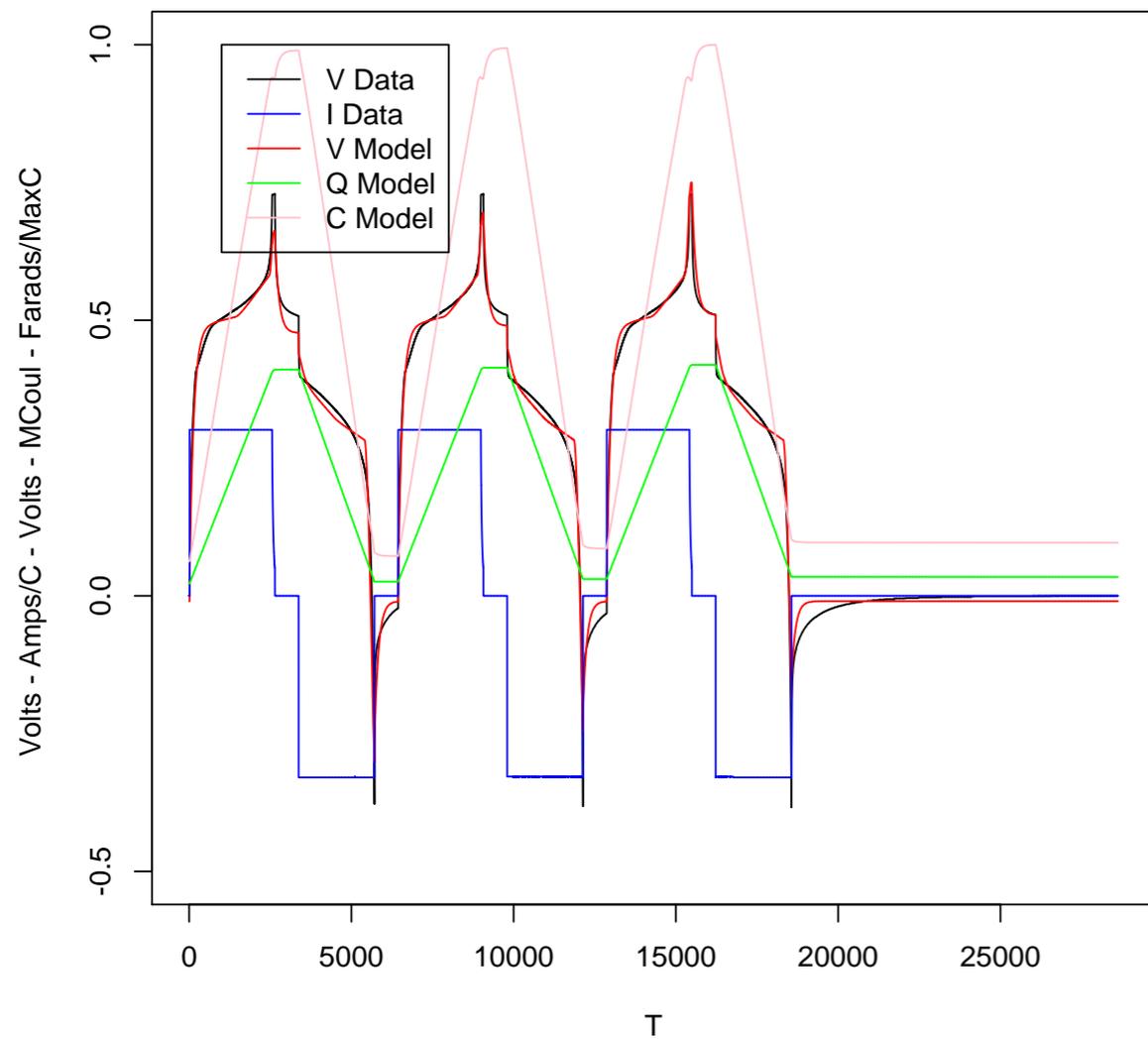
- Introducimos conocimiento del problema mediante una analogía semifísica, normalmente mediante un sistema de ecuaciones diferenciales
- Paralelizarlos es un gran problema para nosotros porque cada predicción depende de la predicción en el instante anterior

Analogía física

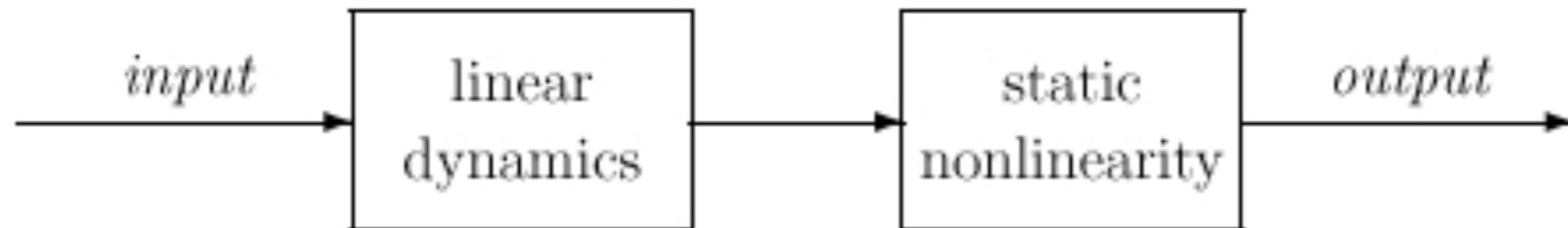


- Analogía física de una batería: el caudal de entrada es la intensidad de carga, el caudal de salida son las pérdidas, la altura es la tensión en bornes de la batería, la carga es el volumen de líquido almacenado y la capacidad, cociente entre la carga y la tensión, es el área de la base del depósito
- El depósito es flexible y al llenarse tiende a aumentar su área C (modelo I) o a disminuir la altura del líquido V (modelo II)

Modelo semi-físico tipo II

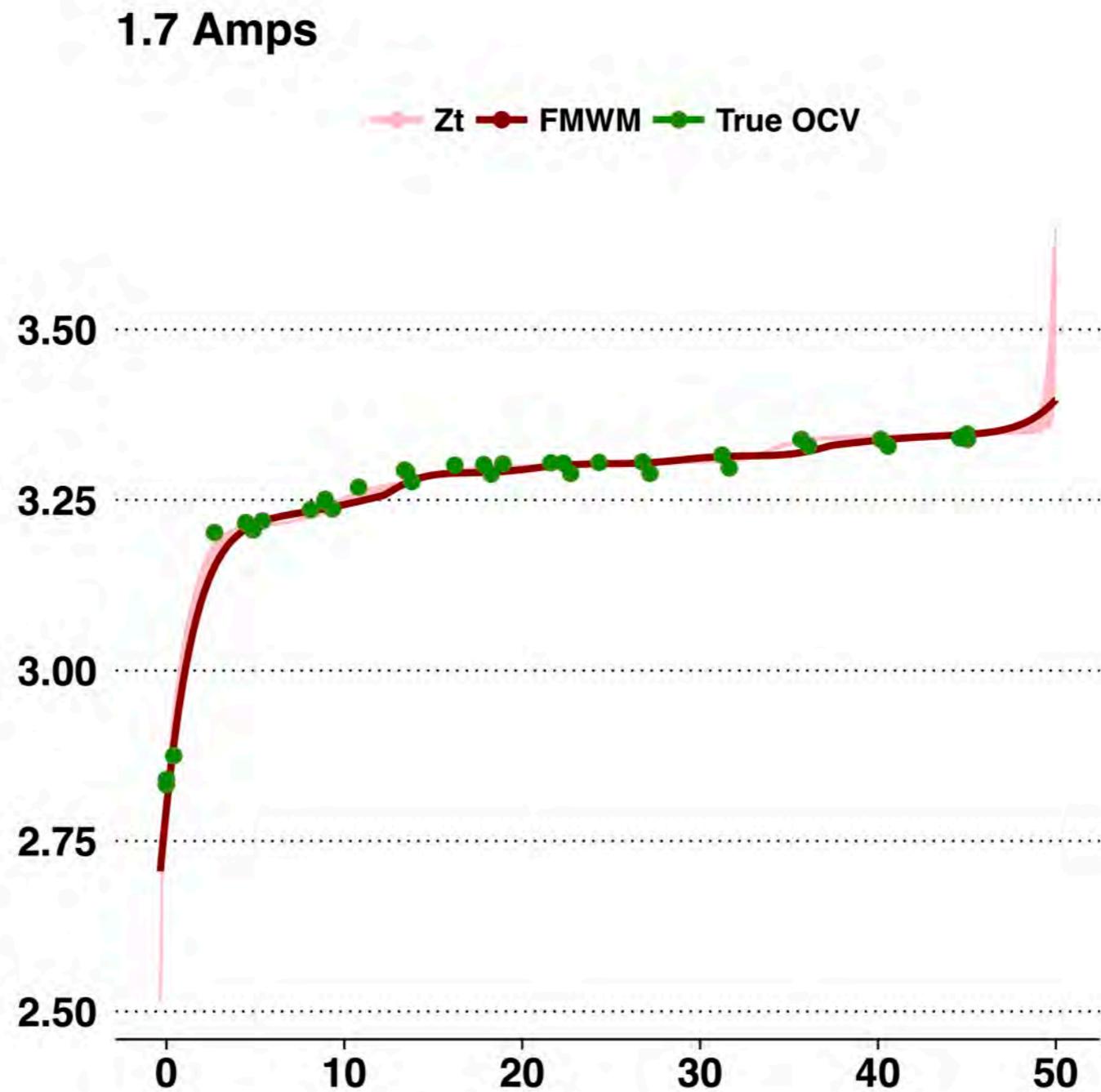


Modelos de Wiener monótonos



- Los modelos de transformación monótona no minimizan el error cuadrático sino la correlación entre la salida de la parte lineal y la respuesta deseada
- Una vez se obtiene la parte lineal, la no linealidad se obtiene con un algoritmo de interpolación, p.e. PCHIP

Resultado de la aplicación a baterías





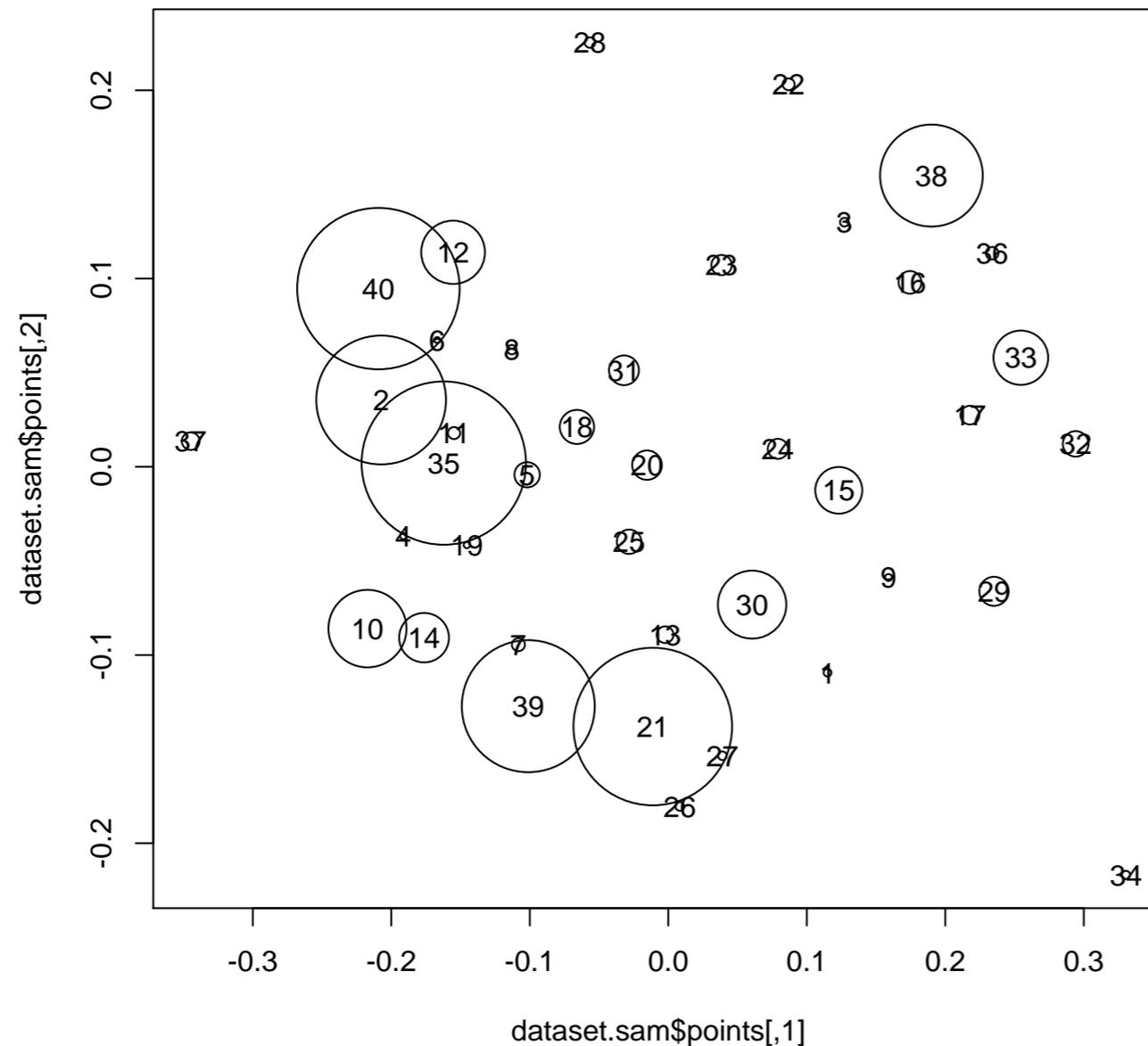
No queremos predecir,
sólo buscar eventos

Múltiples métodos

Modelos lineales

- Clasificación / clustering en el espacio paramétrico
- No nos importa cuándo, sino que haya habido cambios
- Por ejemplo, fallos en refrigeración de multiplicadoras en aerogeneradores

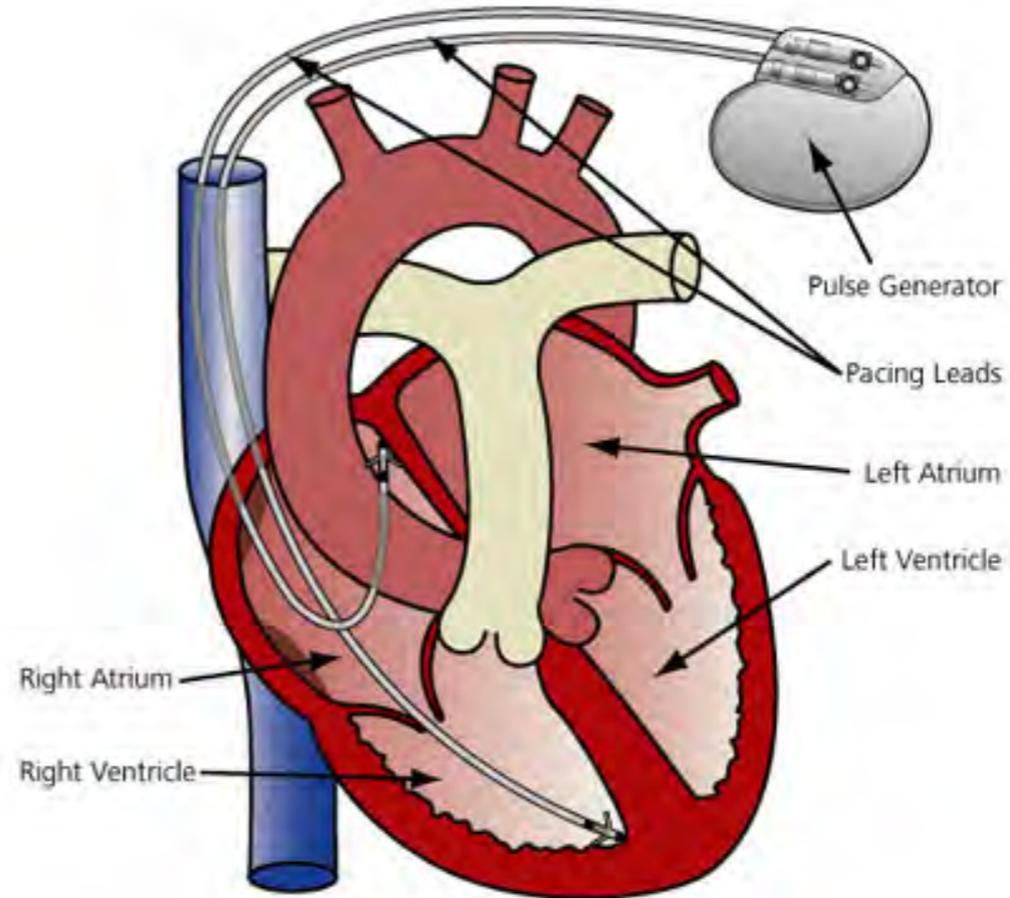
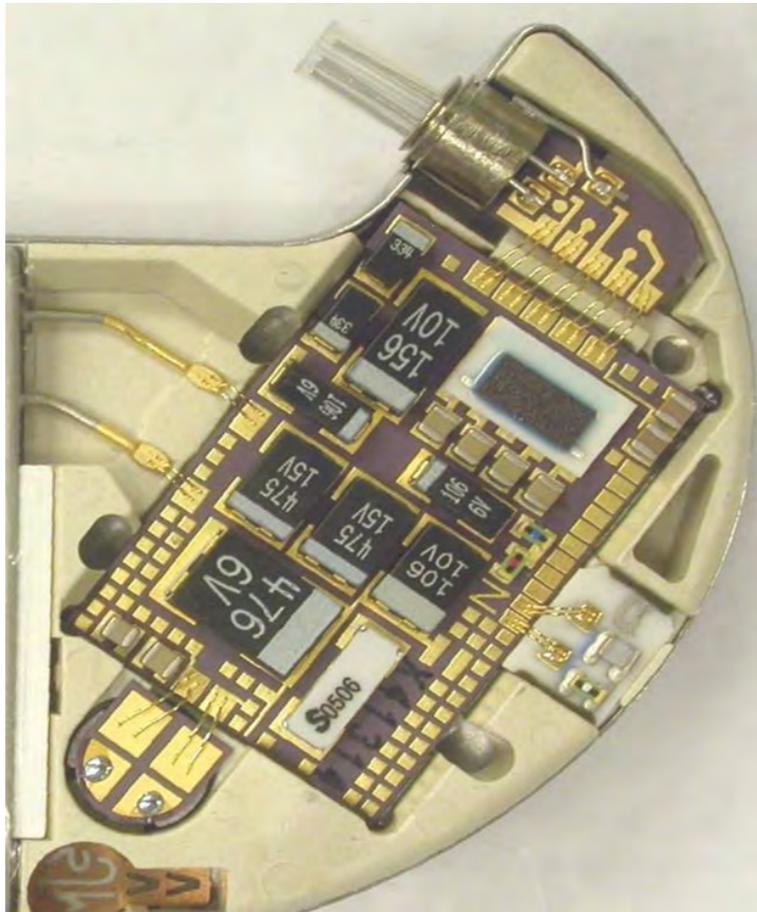
Clustering de los parámetros del modelo en espacio de estados



- Modelo lineal de la temperatura frente a la potencia generada para cada aerogenerador
- Se proyectan los parámetros del modelo de cada aerogenerador (escalado de Sammon)

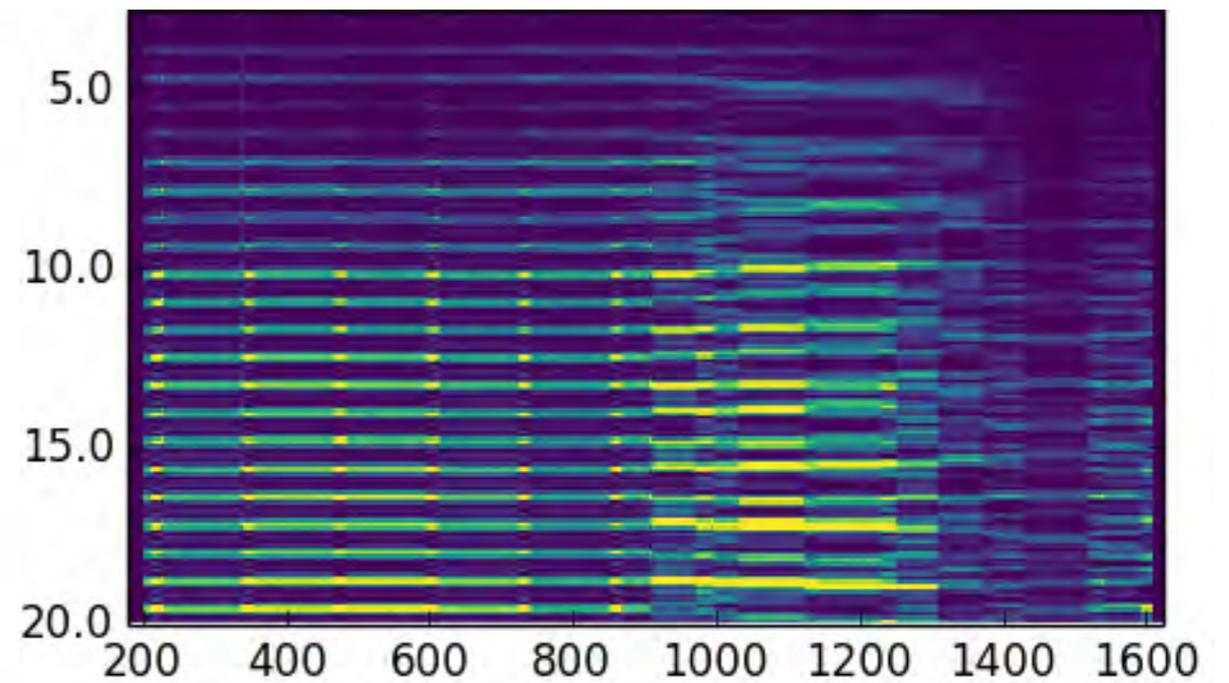
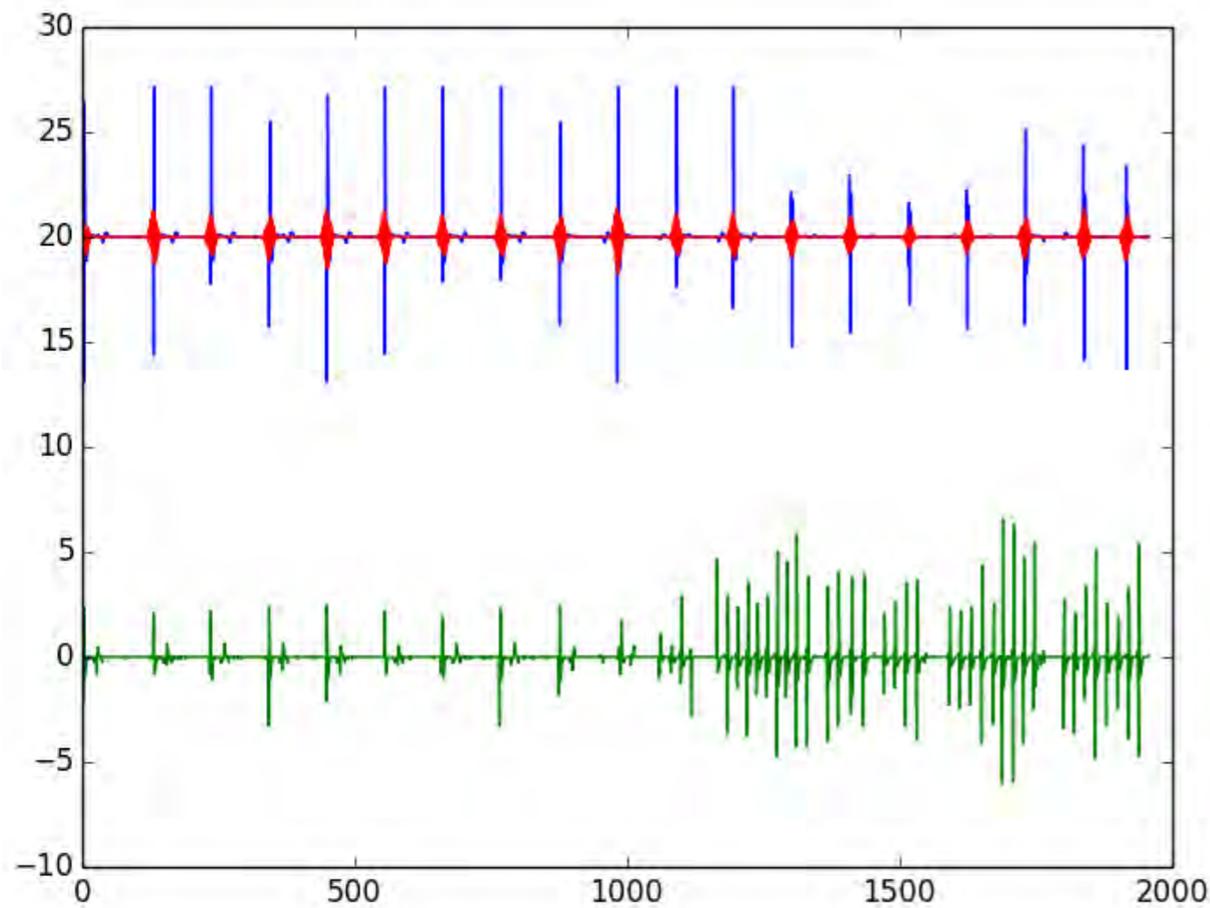
Figura 5.1: Análisis exploratorio gráfico de los parámetros del modelo en espacio de estados de la temperatura frente a la potencia. El área de cada círculo es proporcional al coste de las incidencias en 2009.

Técnicas espectrales localizadas en el tiempo



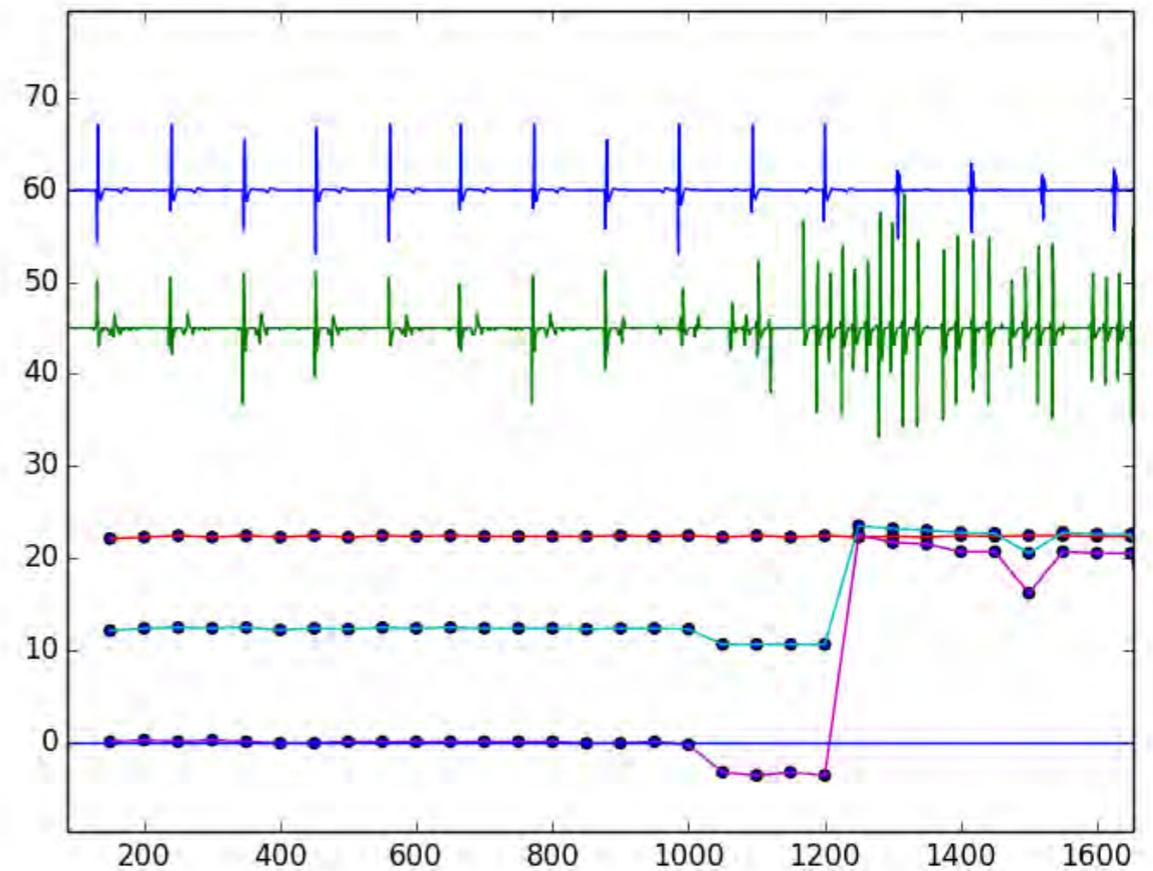
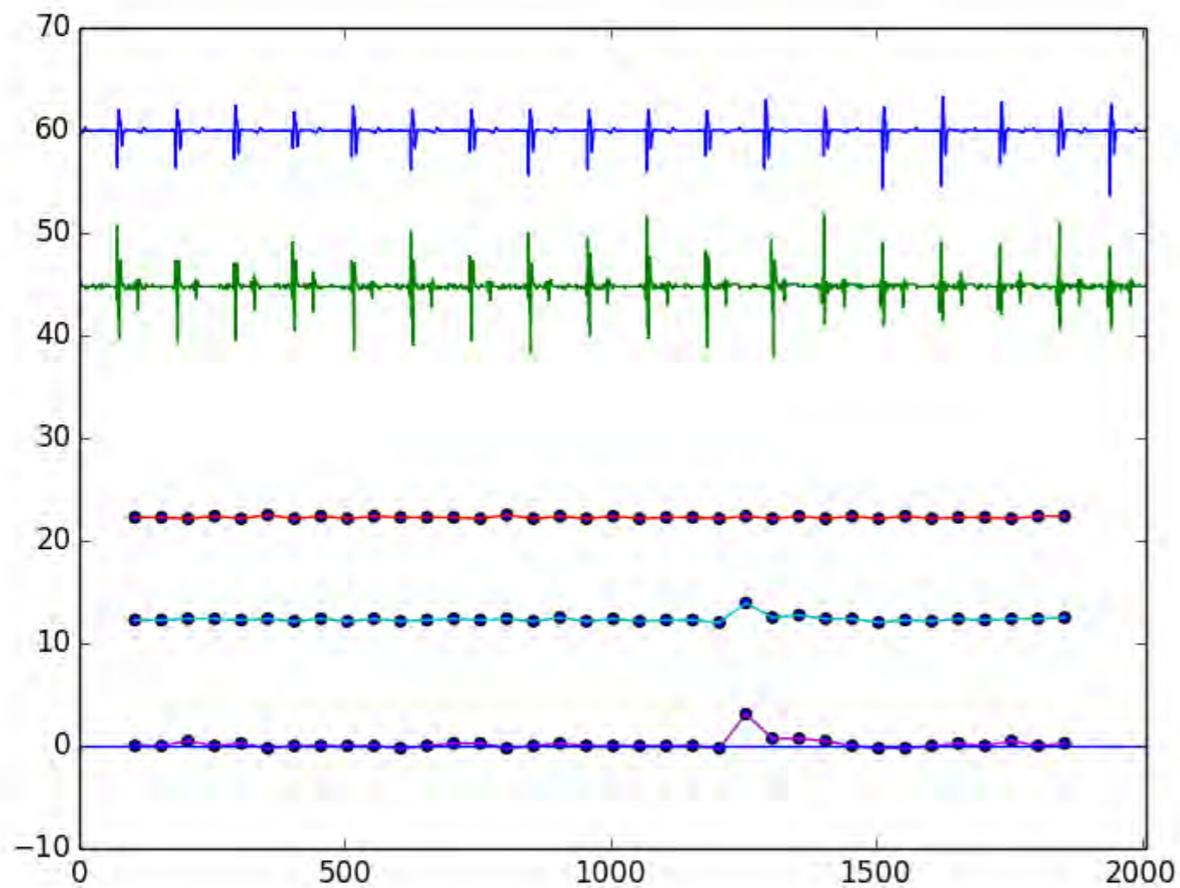
- Detección de fibrilación atrial ignorando la señal de campo lejano (eco ventricular)

Análisis espectral



- Izquierda: señal de aurícula (azul) y ventrículo (verde). Derecha: transformada de Fourier ventaneada

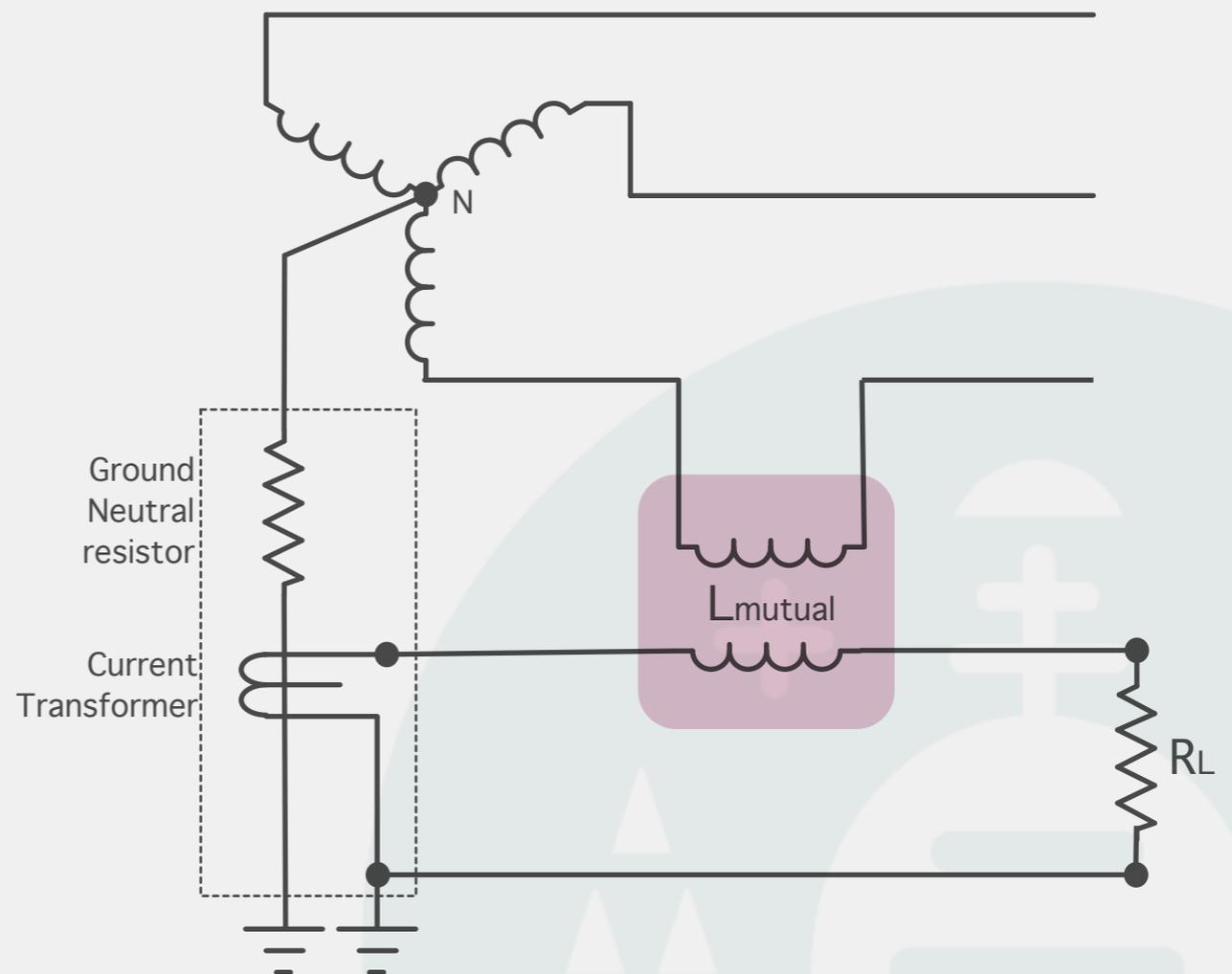
Time warping



- Se distorsiona la señal de la aurícula hasta que su espectro de frecuencias sea plano (duplicando o eliminando periodos). El espectro de la señal del ventrículo deja de ser plano en los episodios de arritmia

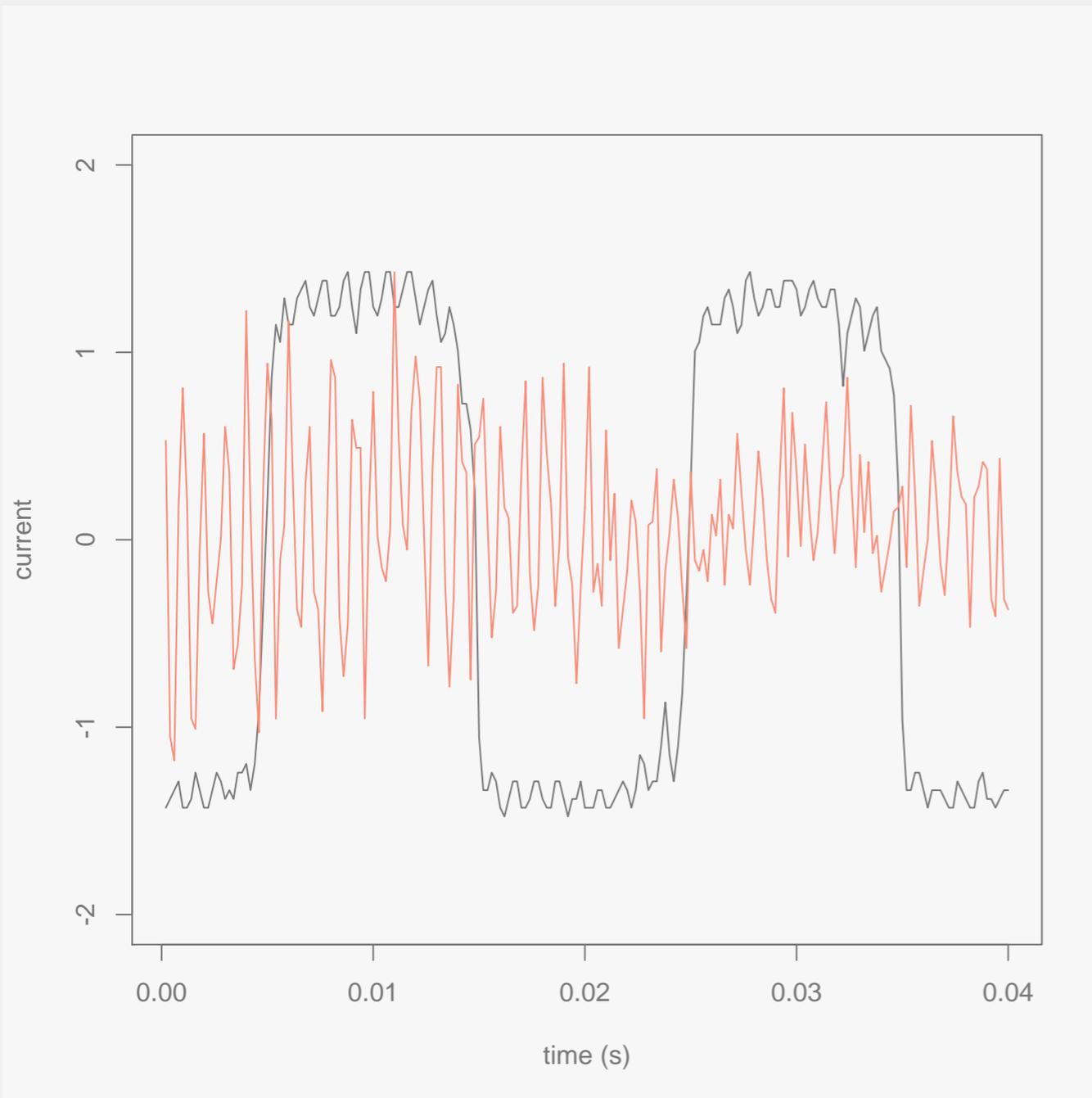
Filtrado adaptativo

- The ambient noise is very high and **the transmission cable is not appropriately shielded**
- Kalman filtering removes Gaussian noise. Ambient noise is not.
- We embed a model of the noise in the system model so we can proceed as if the ambient noise were Gaussian



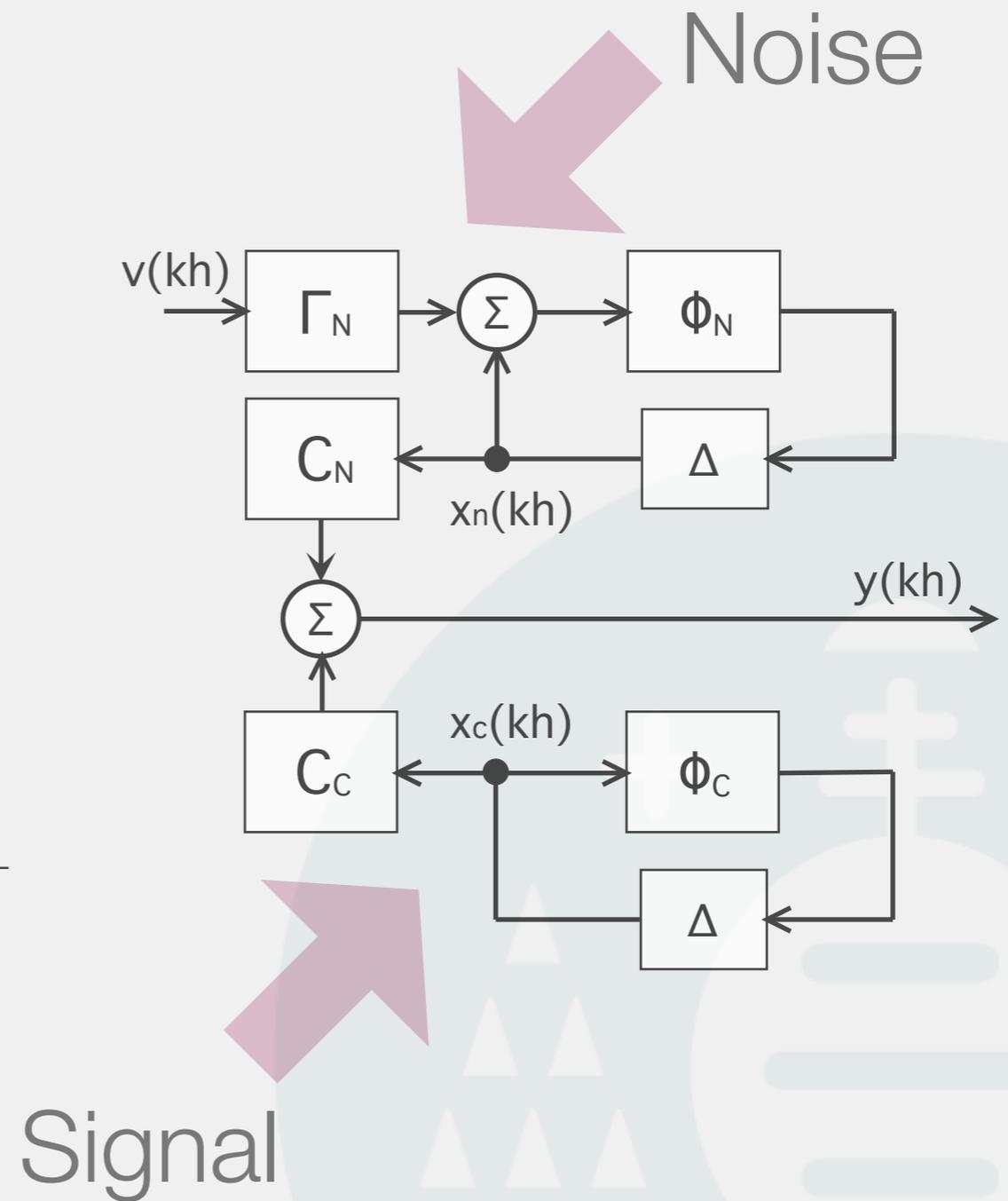
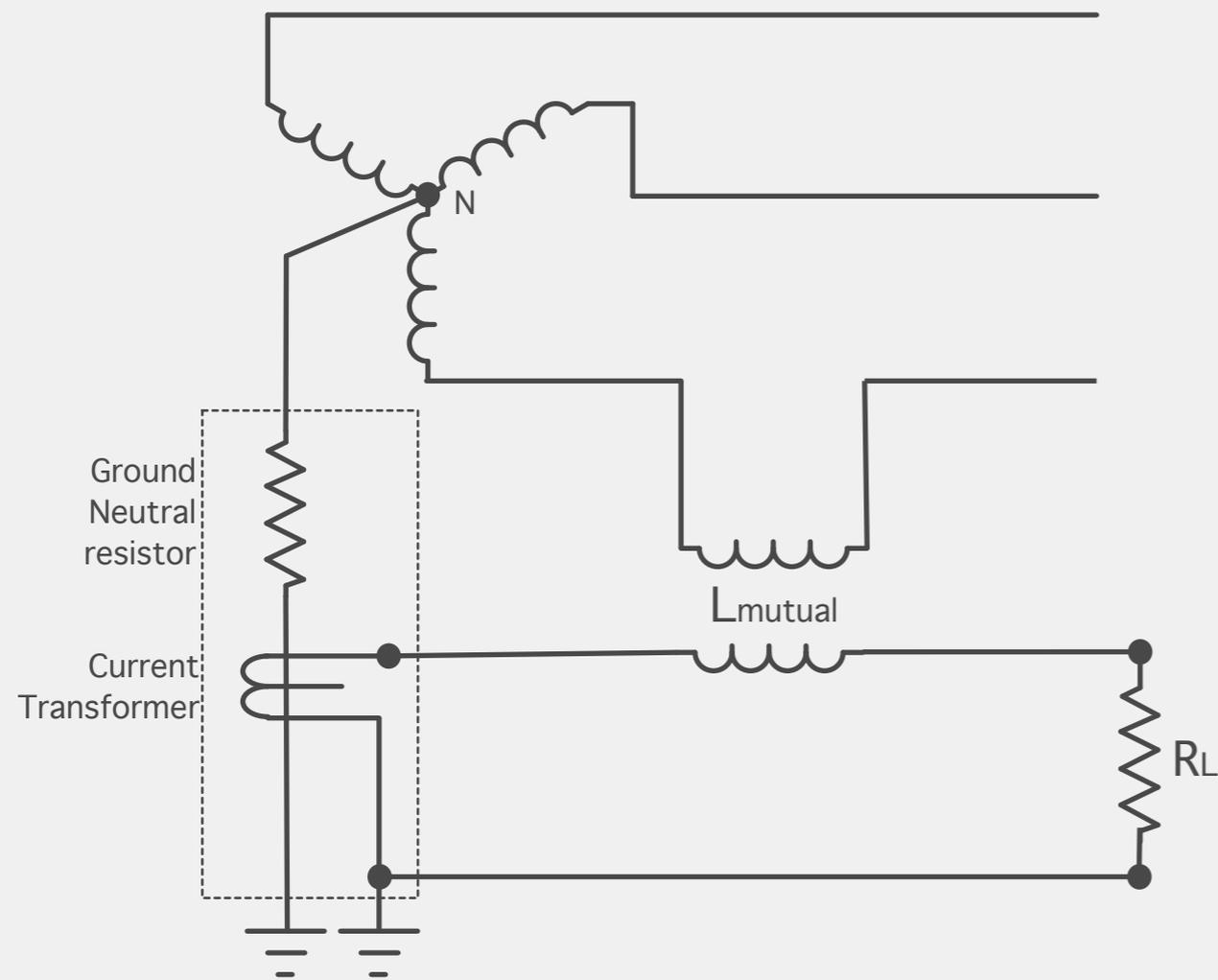


Datos medidos



- Real-world data: measured current (red) and ambient noise (black) before low pass filtering of NGR current

Measurement system and ambient noise



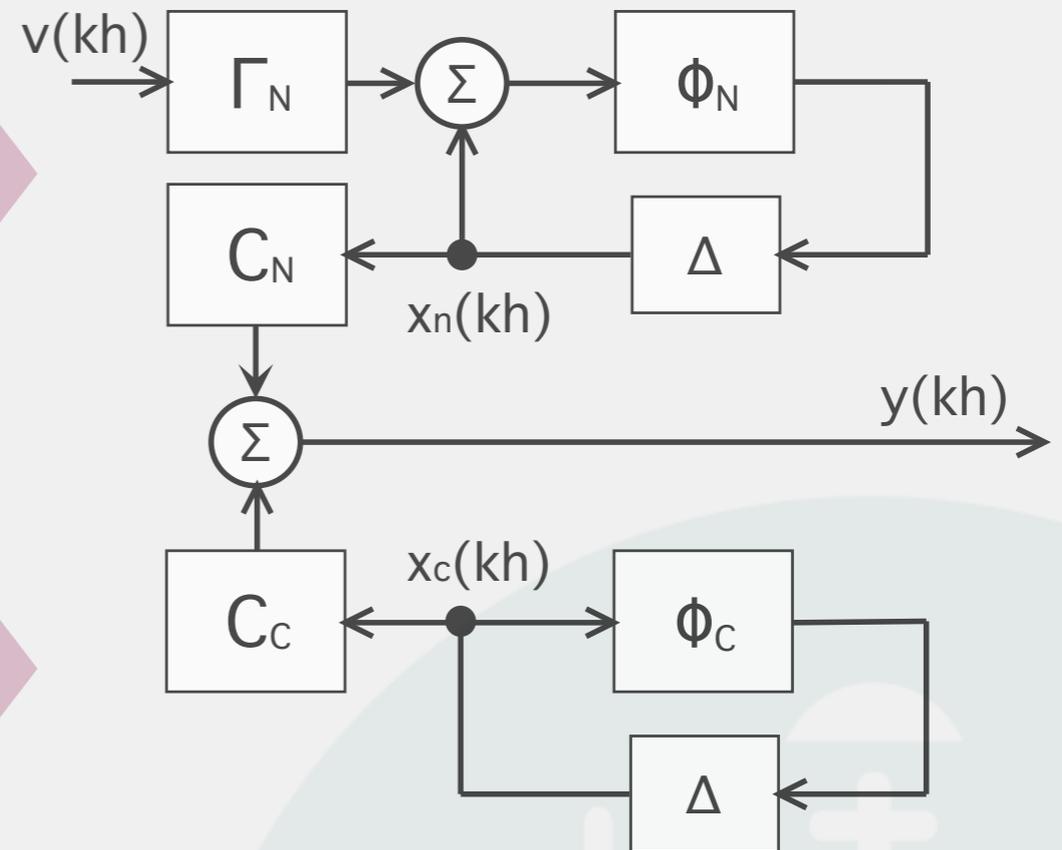
Augmented model and shaping filter

$$x_N(kh + h) = \Phi_N x_N(kh) + \Gamma_N v(k)$$

$$y_N(xk) = C_N x_N(kh)$$

$$x_C(kh + h) = \Phi_C x_N(kh)$$

$$y_C(xk) = C_C x_C(kh)$$

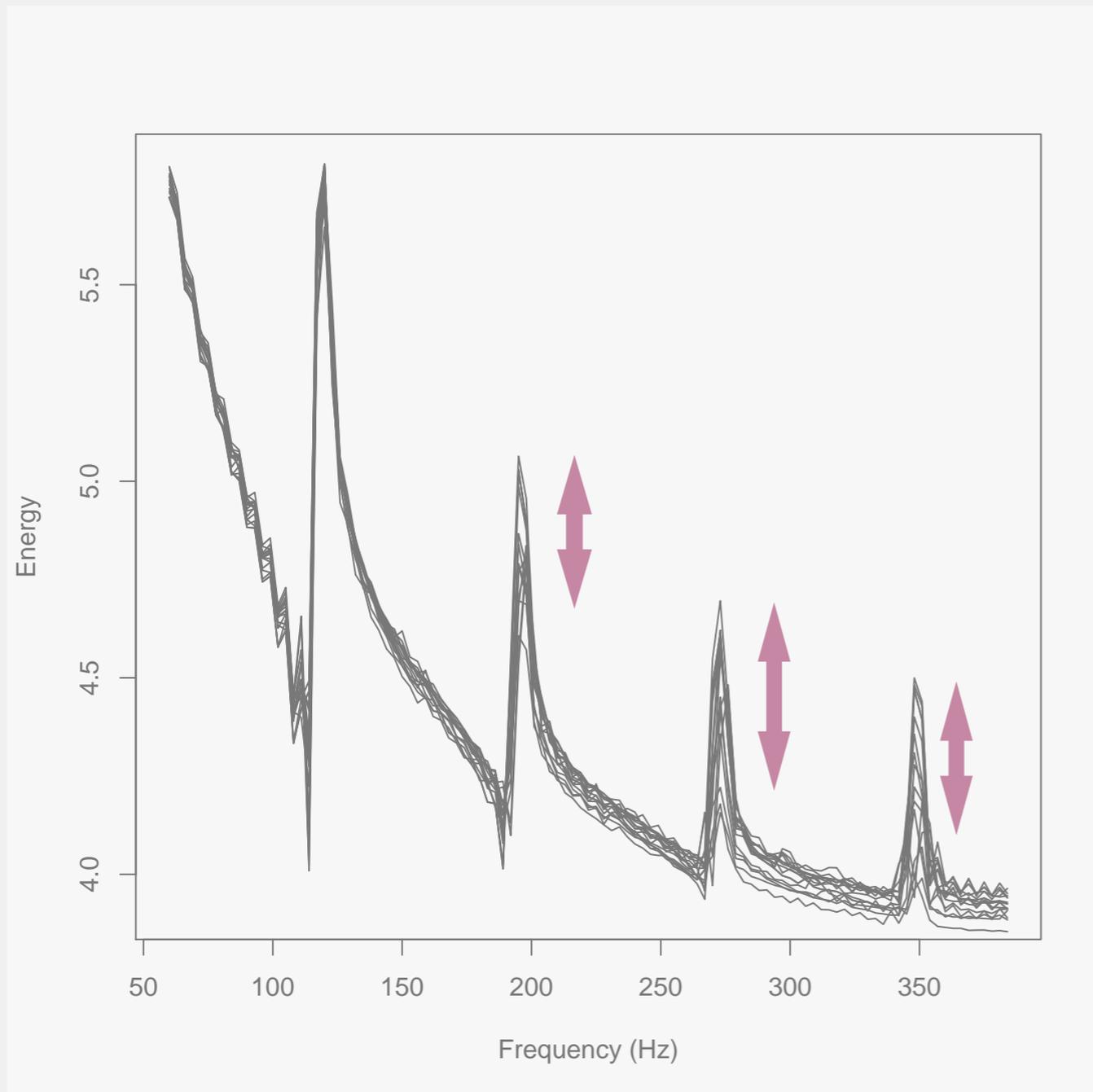


Augmented model

$$x(kh + h) = \Phi x(kh) + e(k)$$

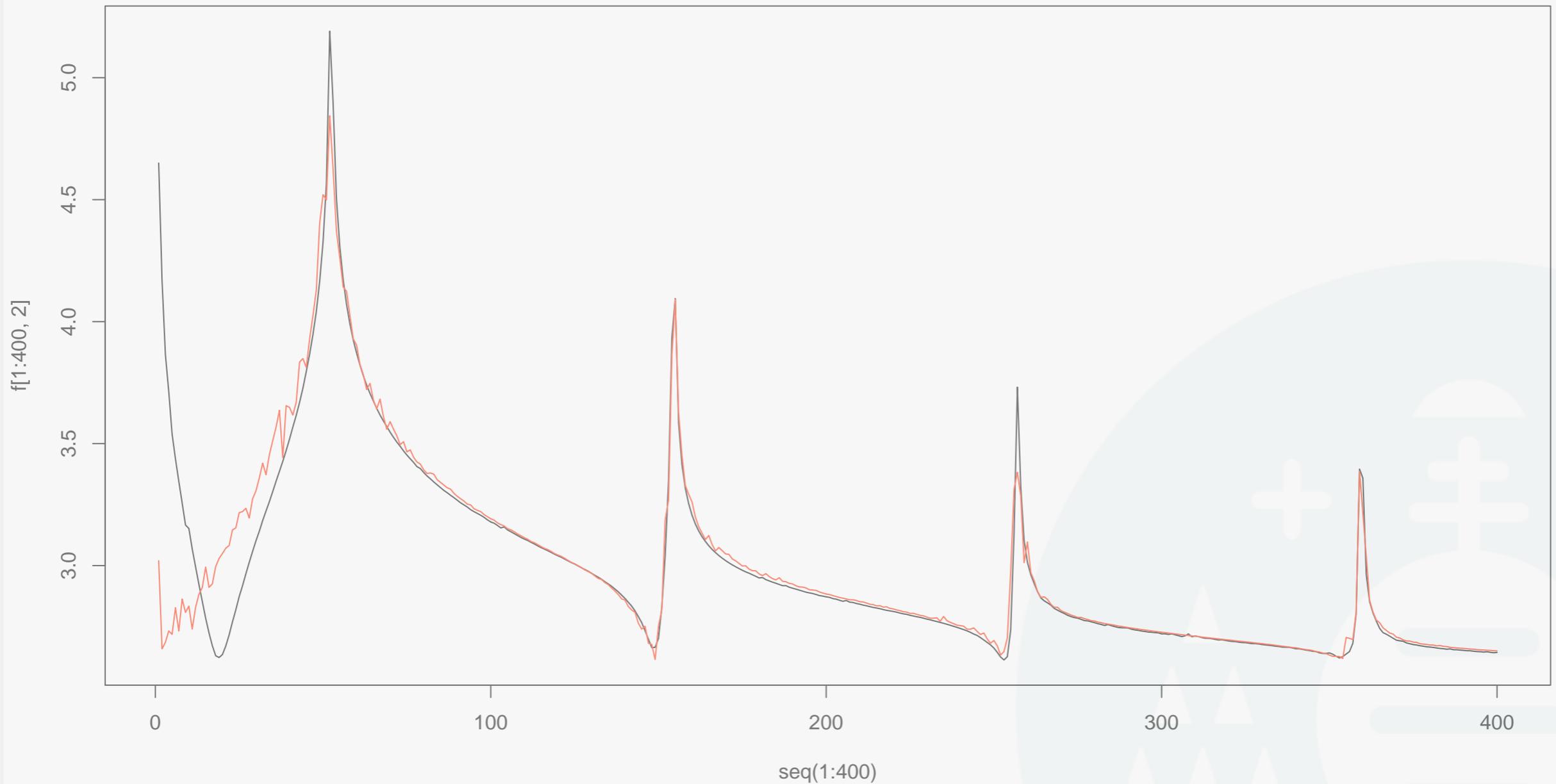
$$y(xk) = C_N x_N(kh)$$

Genetic optimization of a shaping filter: Fitness function (I)

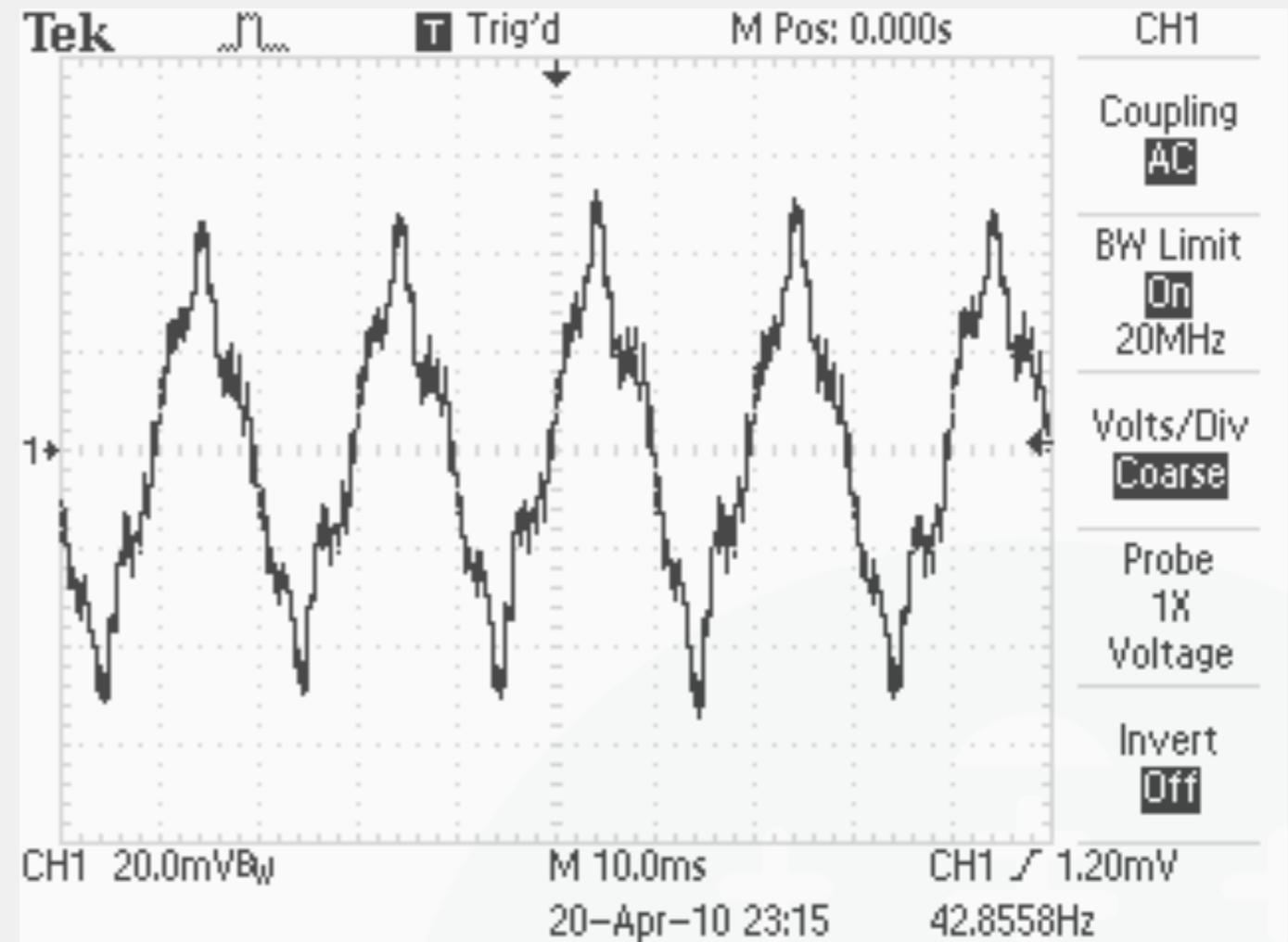
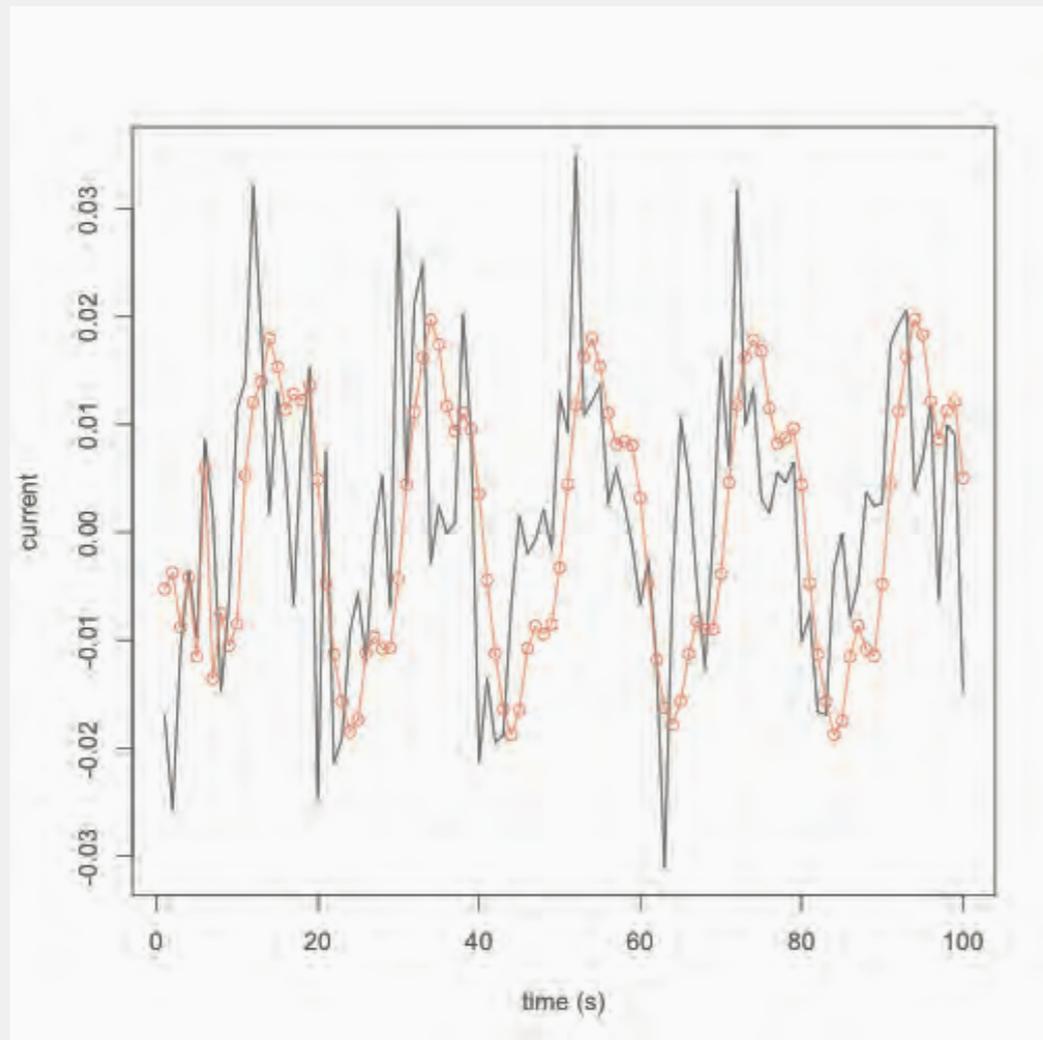


- The PSD was computed as the Fourier transform of the autocorrelation of the time series formed by a sequence of measurements
- We define 10 bands of 50Hz each, and compute the energy at each band, and its dispersion
- The energy of a band is represented by either an interval or a fuzzy set

Genetic Optimization (Example of modelled and actual PSDs)

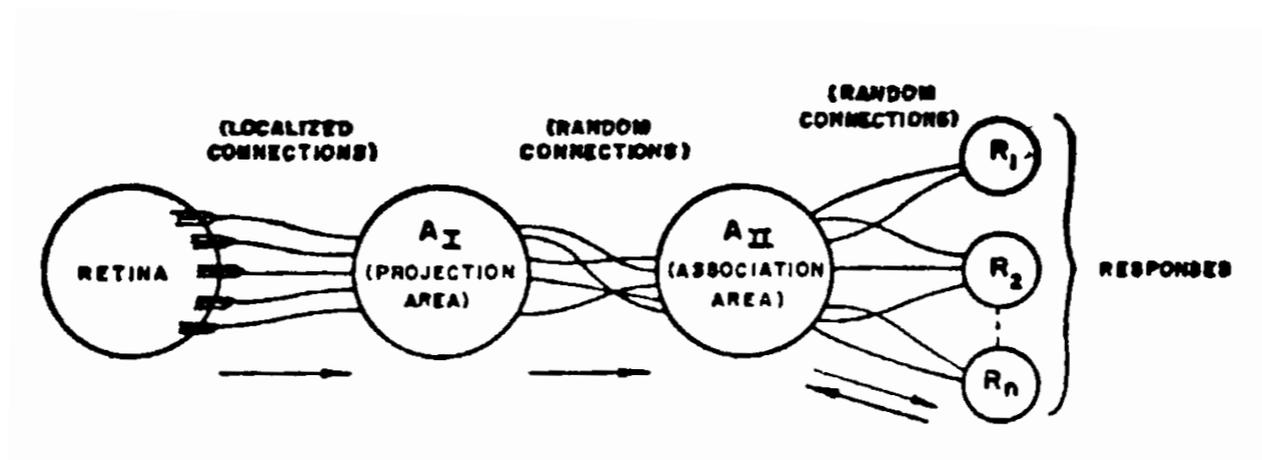
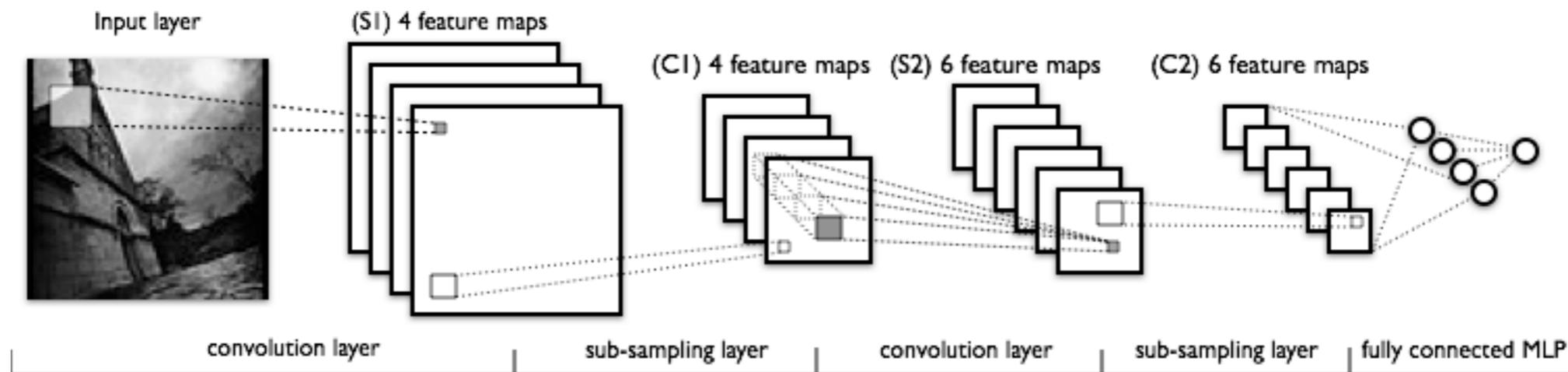


Numerical Results (IV)

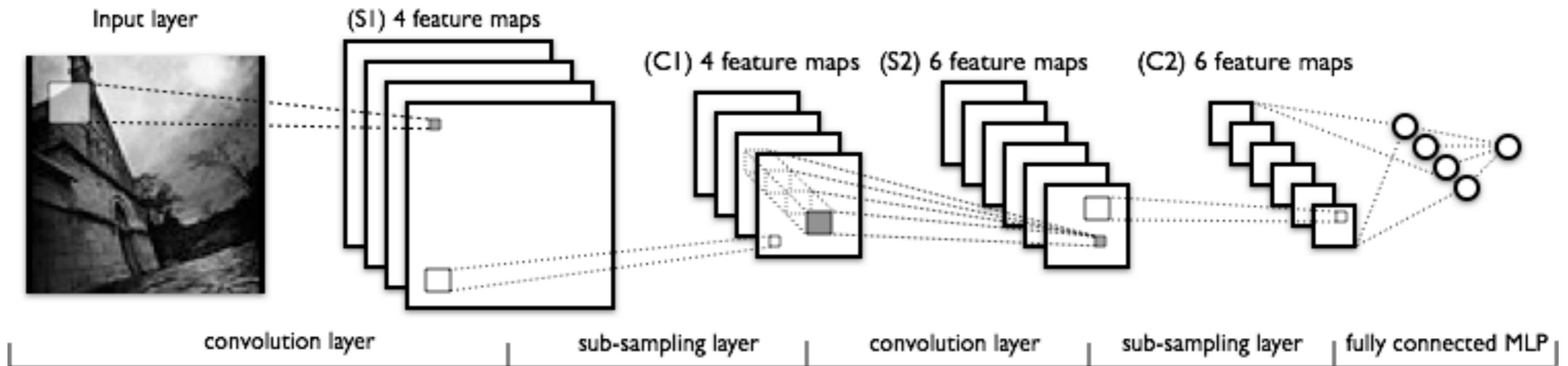


- Left : Example of filtered signal (red) vs unfiltered data (black)
- Right: actual sampling of the current taken at a similar transformer (La Trabiella substation, Avilés) with a properly shielded transmission line, and therefore without noise (i.e. no need to filter)

Deep learning para reconocimiento de patrones en imágenes

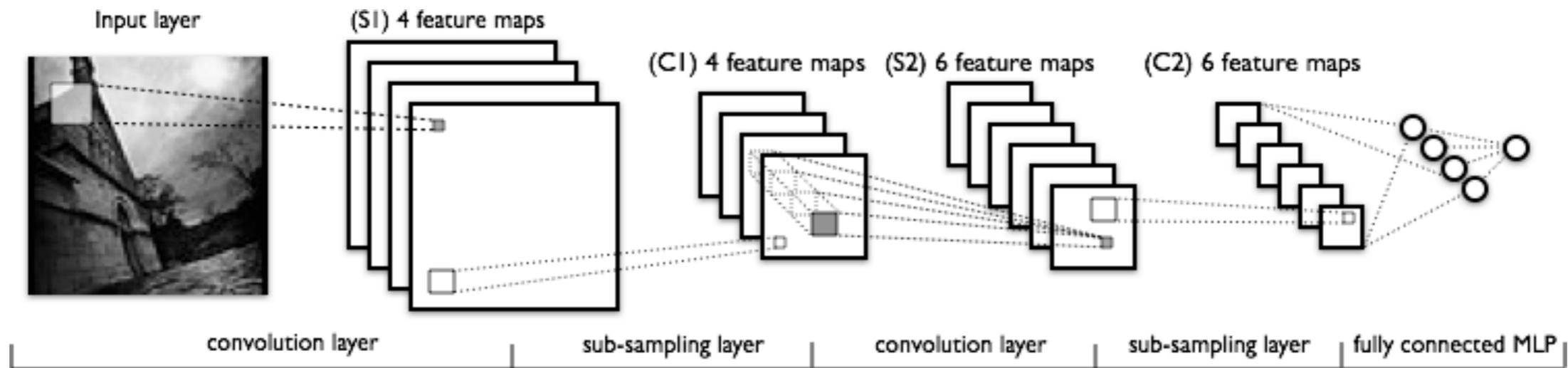


Deep learning para reconocimiento de patrones en imágenes

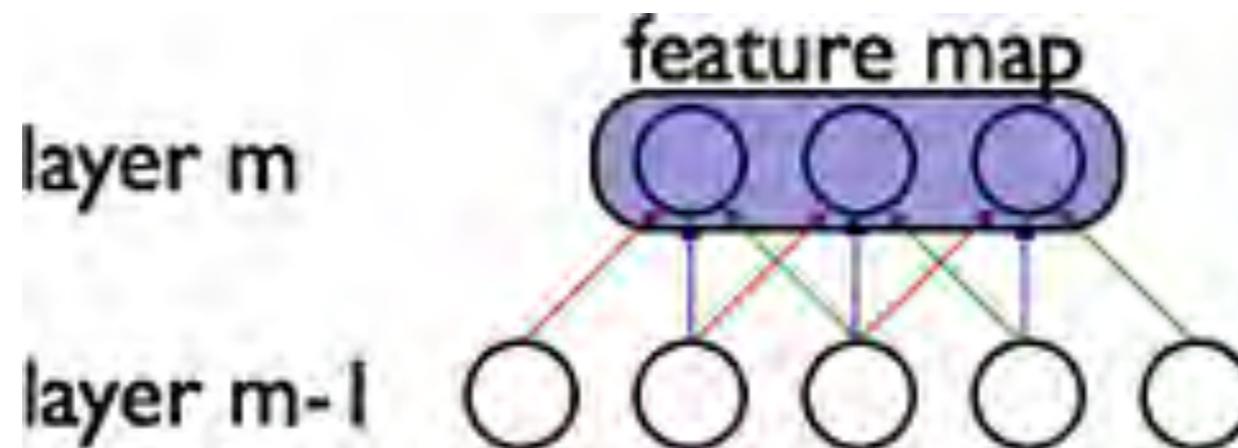


- Una red tiene al menos una capa de convolución, que transforma la imagen en varios mapas (imágenes de contornos)
- Cada convolución puede afectar a un canal de la imagen (RGB) o a varios
- Los mapas se remuestrean para reducir su dimensión y se combinan mediante una capa de red convencional o bien se toman como entrada de una segunda capa de convolución

Deep learning para reconocimiento de patrones en imágenes

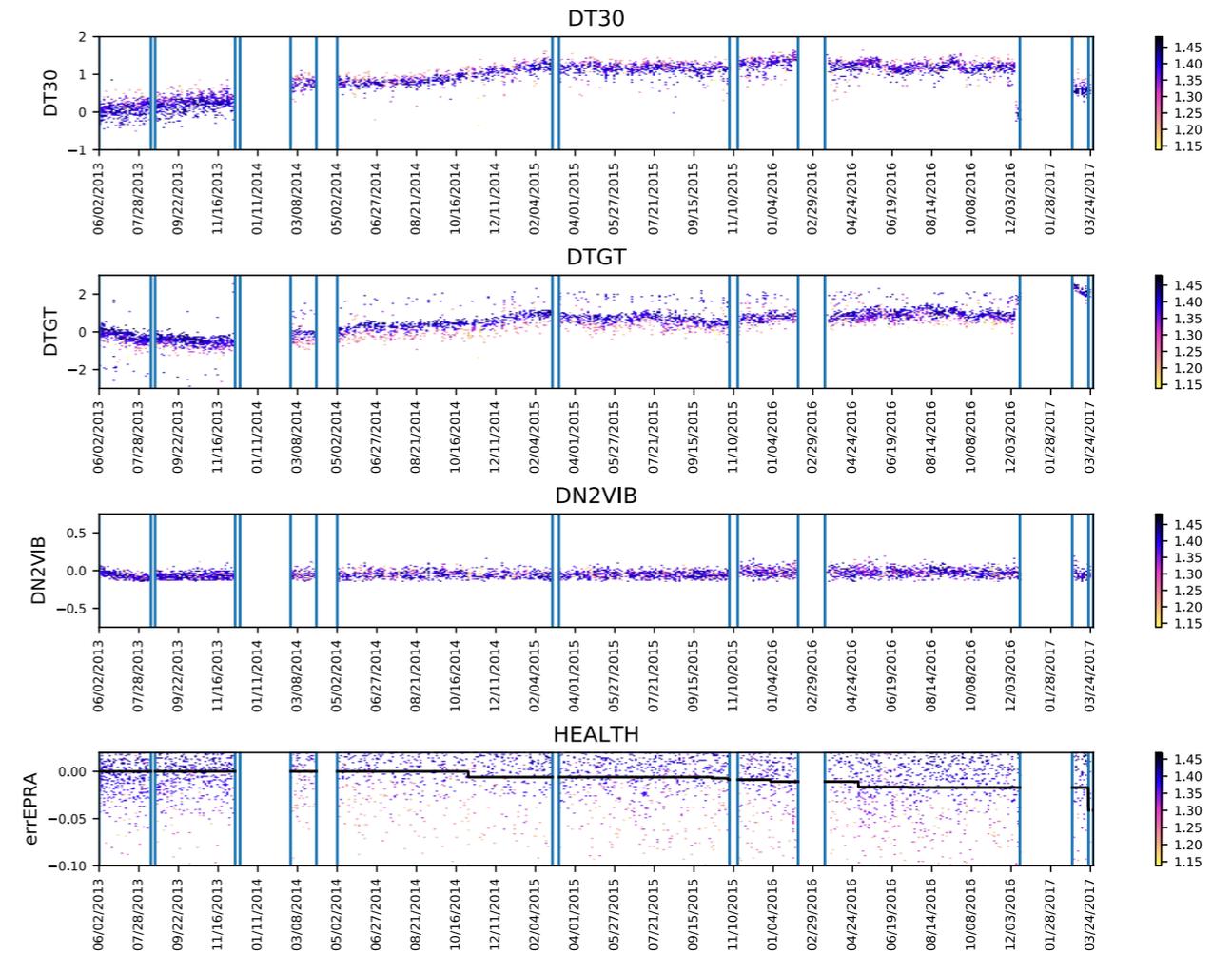
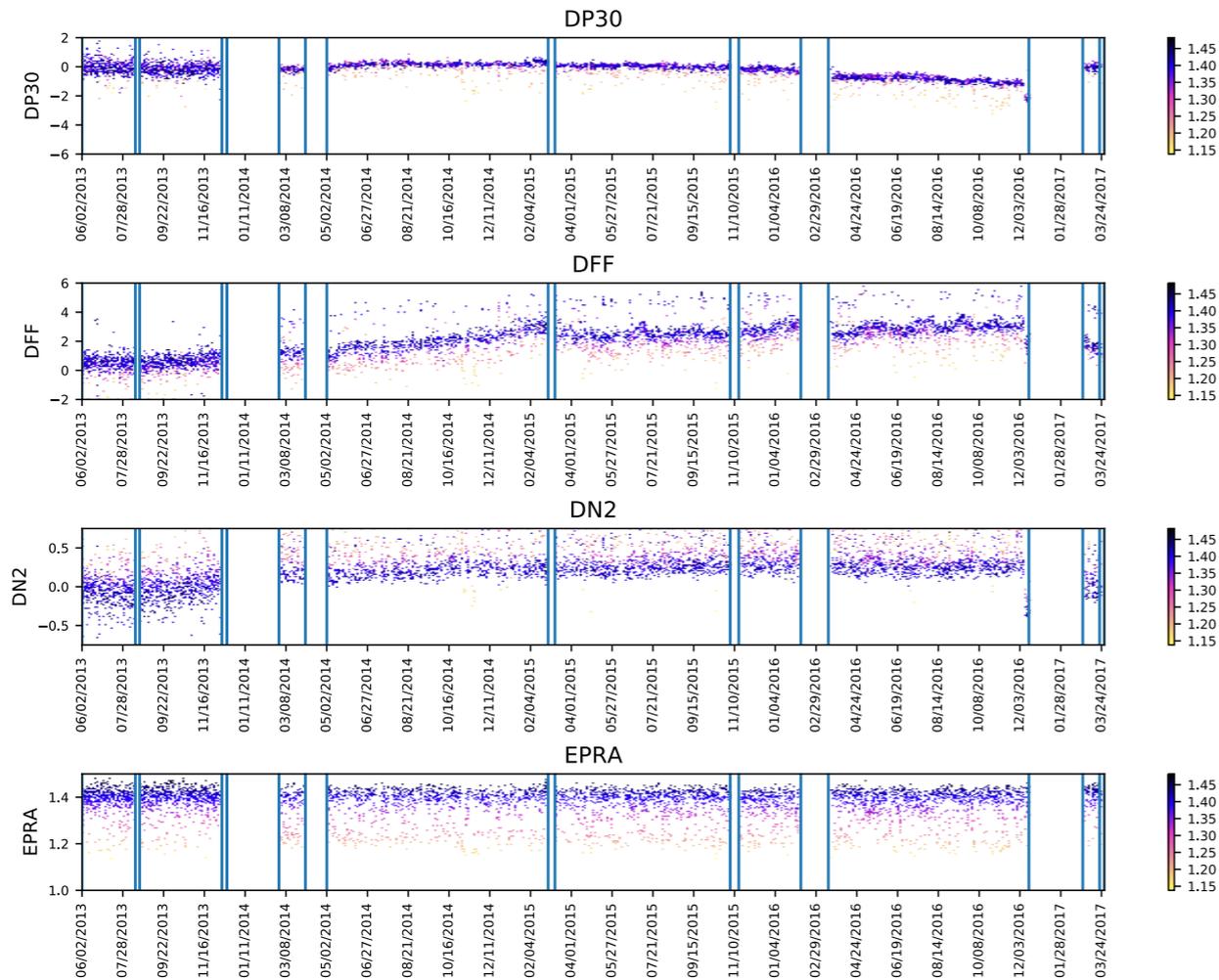


- Las convoluciones se implementan mediante pesos compartidos: las conexiones del mismo color en la figura inferior tienen el mismo peso
- El número de pesos necesario para calcular cada mapa depende del área del campo visual y no de la dimensión de la imagen



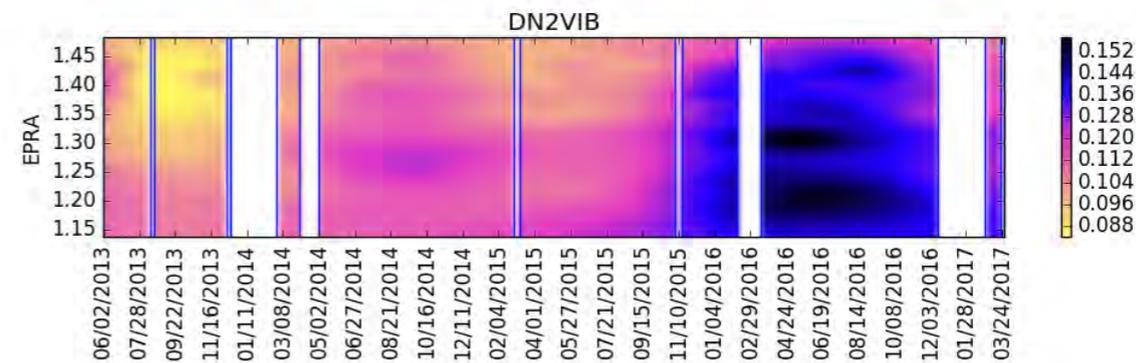
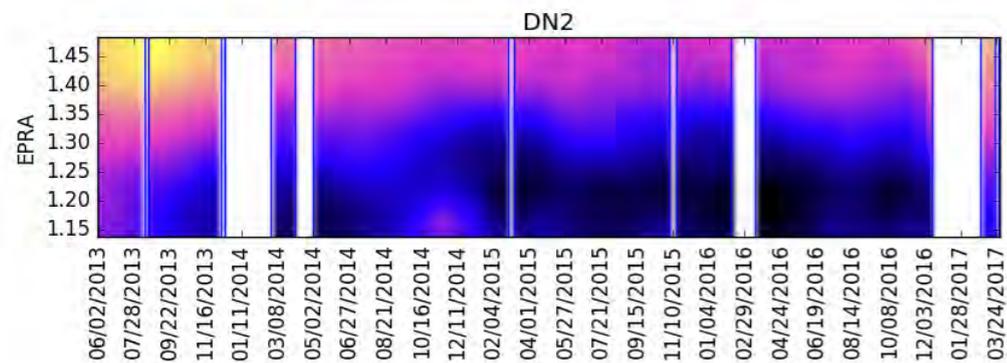
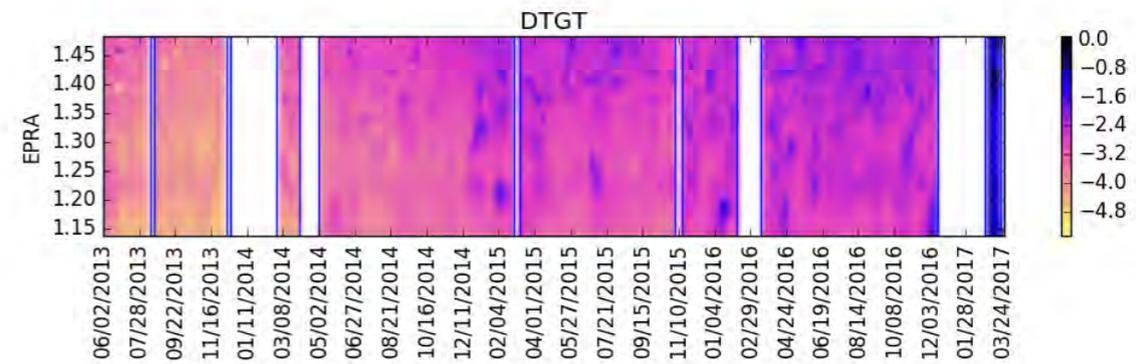
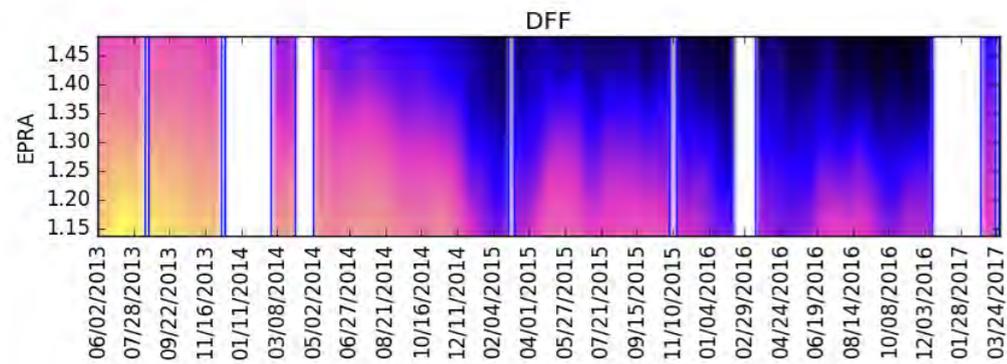
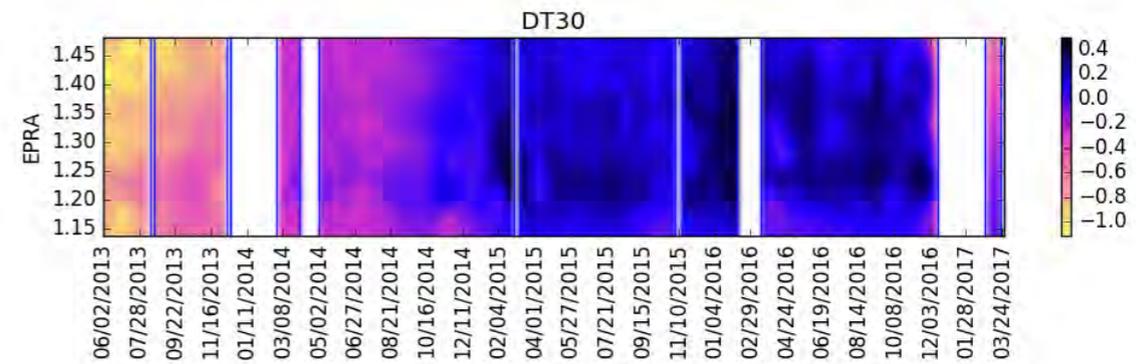
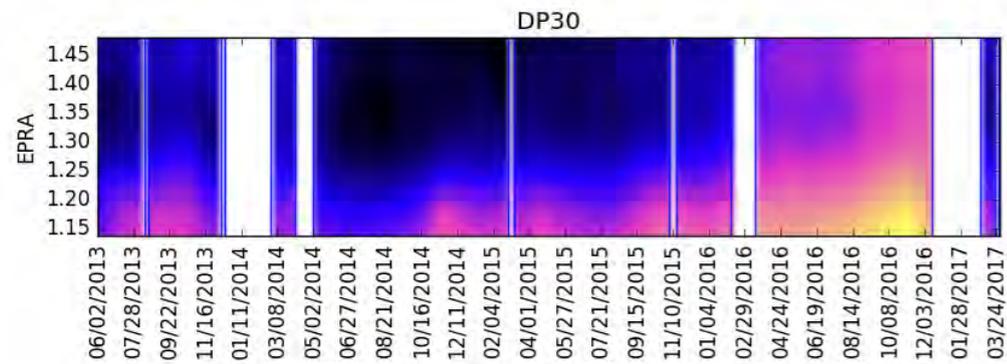
Deep learning para reconocimiento de patrones en imágenes

CR - 13177_CR.csv.dat



Deep learning para reconocimiento de patrones en imágenes

CR - 13177_CR.csv.dat



Algunas conclusiones

- El resumen es muy incompleto, sólo hemos mostrado algunas técnicas en las que hemos trabajado
- La mayoría de las técnicas son muy costosas en CPU o en memoria
- ESN (reservoir computing) son adecuadas para modelar sistemas dinámicos pero muy costosas en memoria. Hemos hecho una implementación en Tensorflow
- LSTM es el modelo usado más frecuentemente para modelos recurrentes. Está incluido en Tensorflow
- La tecnología más usada para buscar patrones en imágenes son las redes convolucionales. El mayor problema es que necesitan una gran cantidad de patrones. Una vez entrenadas son muy eficientes.