Parameter estimation from partial observations with neural networks

Manu Kalia

SIAM MDS 2020

June 25, 2020

Collaborators

- Hil Meijer, Christoph Brune (University of Twente)
- J. Nathan Kutz, Steven Brunton (University of Washington)

Time-evolution models from data

Physical modeling

- Biophysics
- Neuroscience
- Fluid dynamics
- etc.

Model optimisation

- Parameter estimation
- UDEs [Rackauckas et al, 2020]

Model discovery

- SINDy [Brunton et al., 2016]
- DMD
- Koopman
- HAVOK [Brunton et al., 2017]

Time-evolution models from data

Physical modeling

- Biophysics
- Neuroscience
- Fluid dynamics
- etc.

Model optimisation

- **Parameter estimation**
- UDEs [Rackauckas et al, 2020]

Model discovery

- SINDy [Brunton et al., 2016]
- DMD
- Koopman
- HAVOK [Brunton et al., 2017]

The problem

Consider an ODE,

$$\dot{x} = f(x, \alpha), \quad x \in \mathbb{R}^n, \alpha \in \mathbb{R}^p. \qquad \textbf{(Model)}$$

We observe the state **partially** via some $H(x)$

$$y = H(x),\ y \in \mathbb{R}^m,\ m < n. \qquad \textbf{(Observation)}$$

The problem

Consider an ODE,

$$\dot{x} = f(x, \alpha), \quad x \in \mathbb{R}^n, \alpha \in \mathbb{R}^p. \qquad \textbf{(Model)}$$

We observe the state **partially** via some $H(x)$

$$y = H(x),\ y \in \mathbb{R}^m,\ m < n. \qquad \textbf{(Observation)}$$

Problem: Given noisy y, f and H also, estimate α.

The problem

Consider an ODE,

$$\dot{x} = f(x, \alpha), \quad x \in \mathbb{R}^n, \alpha \in \mathbb{R}^p. \qquad \textbf{(Model)}$$

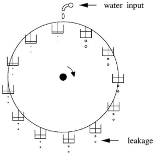
We observe the state **partially** via some $H(x)$

$$y = H(x), \; y \in \mathbb{R}^m, \; m < n. \qquad \textbf{(Observation)}$$

Problem: Given noisy y, f and H also, estimate α.

Example

Malkus-waterwheel equations:



$$\dot{\mathbf{x}} = \begin{cases} \dot{x} = \sigma(y - x) \\ \dot{y} = -y + ax + zx \\ \dot{z} = -bz + xy. \end{cases}$$ **(Model)**

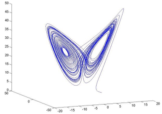
$$H(\mathbf{x}) = x.$$ **(Observation)**

$$C(\sigma) = \|x^{data} - x\|_2^2.$$ **(Loss)**

Compute $\sigma_{opt} = \arg\min C(\sigma)$. **(Optimization)**

Example

Malkus-waterwheel equations:

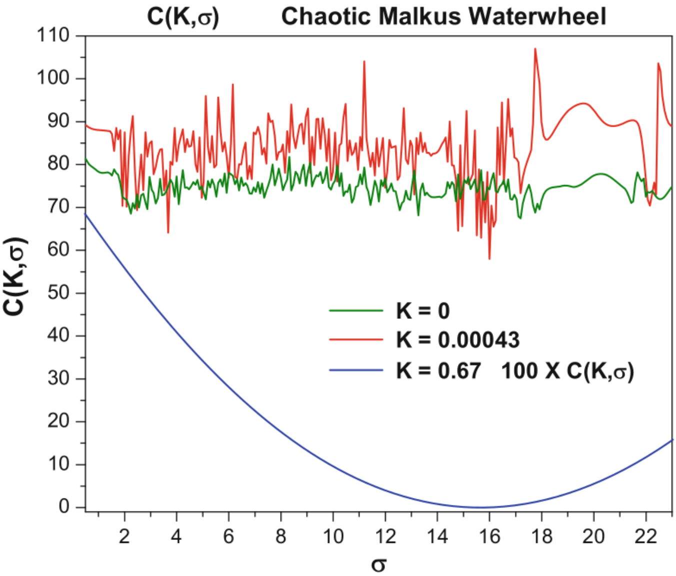


$$\dot{\mathbf{x}} = \begin{cases} \dot{x} = \sigma(y - x) \\ \dot{y} = -y + ax + zx \\ \dot{z} = -bz + xy. \end{cases} \qquad \textbf{(Model)}$$

$$H(\mathbf{x}) = x. \qquad \textbf{(Observation)}$$

$$C(\sigma) = \|x^{data} - x\|_2^2. \qquad \textbf{(Loss)}$$

Compute $\sigma_{opt} = \arg\min C(\sigma)$. **(Optimization)**



From H. Abarbanel, Predicting the Future - Completing models of observed complex systems, *Springer* (2013)

Synchronization

We now consider the following problem,

$$\dot{\mathbf{x}} = \begin{cases} \dot{x} = \sigma(y - x) + K(x^{data} - x) \\ \dot{y} = -y + ax + zx \\ \dot{z} = -bz + xy. \end{cases}$$

(Model)

$$H(\mathbf{x}) = x.$$

(Observation)

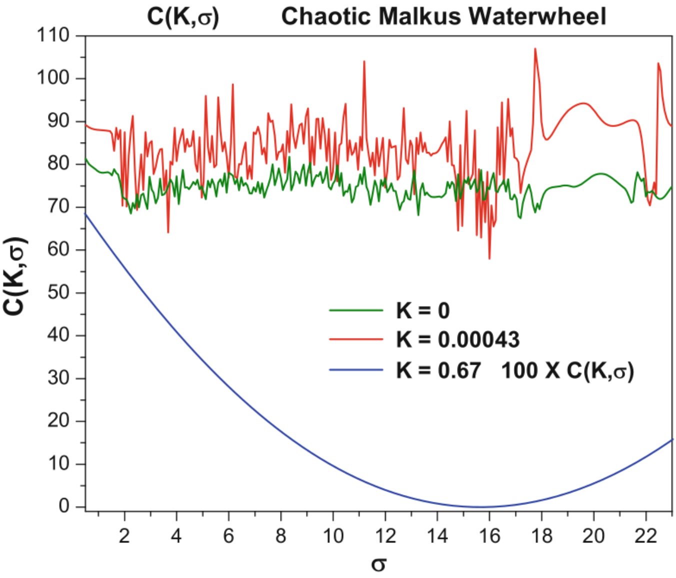
$$C(K, \sigma) = \|x^{data} - x\|_2^2.$$

(Loss)

Fix K. Compute $\sigma_{opt} = \arg\min C(\sigma)$.

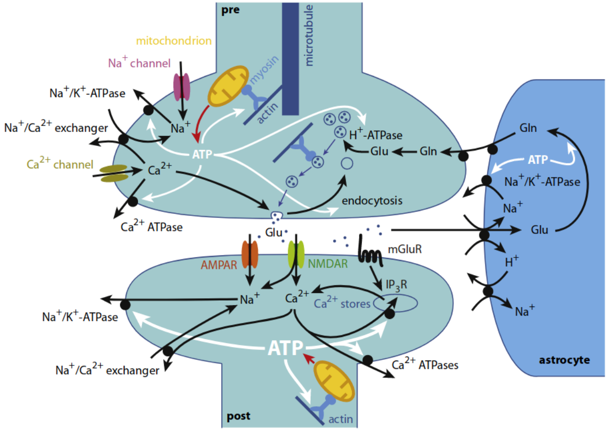
(Optimization)

From H. Abarbanel, Predicting the Future - Completing models of observed complex systems, *Springer* (2013)



From H. Abarbanel, Predicting the Future - Completing models of observed complex systems, *Springer* (2013)

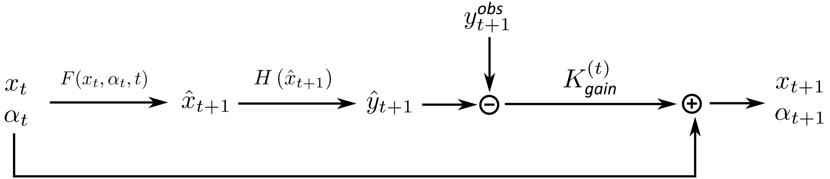
Problems in neuroscience



- Highly nonlinear
- Measurements of *few* ions and voltages available.

Gerkau et al. (2018), Kalia et al. (In prep.)

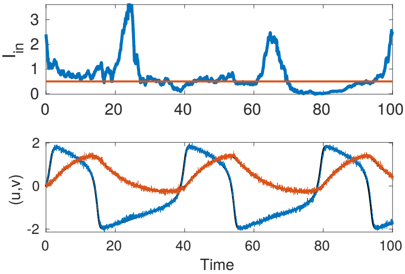
Augmented Ensemble Kalman filter (AEnKF)



- Augmented filtering $\rightarrow$ Append parameter α to state x.
- α updated by propagating cross-covariances.

Arnold, Calvetti, Somersalo. *Inverse Problems* (2014); Evensen. *Ocean Dynamics* (2003).

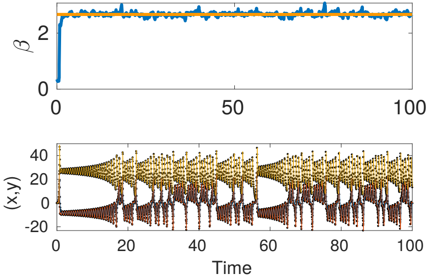
AEnKF: examples



FitzHugh Nagumo

$$\begin{cases} \dot{v} = v - v^3/3 - r + I_{in} \\ \dot{r} = 1/\tau(v + a - br) \end{cases}$$

$$H(\mathbf{x}) = (v, w)$$



Lorenz63

$$\begin{cases} \dot{x} = \sigma(y - x) \\ \dot{y} = x(\rho - z) - y \\ \dot{z} = xy - \beta z \end{cases}$$

(Model)

$$H(\mathbf{x}) = (x, y, z)$$

(Observation)

Augmented Ensemble Kalman filter (AEnKF)

Efficient and robust over higher dimensions, but

(1) Requires explicit numerical method, linear observation operator

(2) State estimation $\not\Longrightarrow$ parameter estimation.

Arnold, Calvetti, Somersalo. *Inverse Problems* (2014); Evensen. *Ocean Dynamics* (2003).

Augmented Ensemble Kalman filter (AEnKF)

Efficient and robust over higher dimensions, but

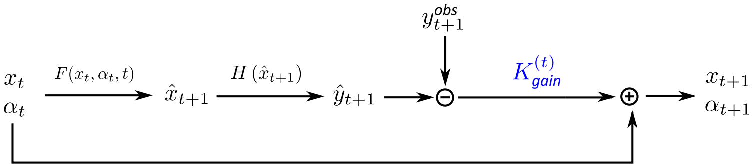
(1) Requires explicit numerical method, linear observation operator

(2) State estimation $\not\Rightarrow$ parameter estimation.

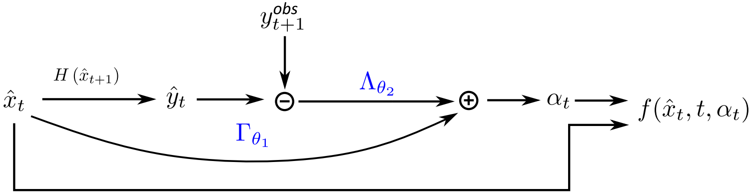
Goal: Better parameter estimation by adapting above approaches with neural networks.

Arnold, Calvetti, Somersalo. *Inverse Problems* (2014); Evensen. *Ocean Dynamics* (2003).

Neural networks and Kalman filters

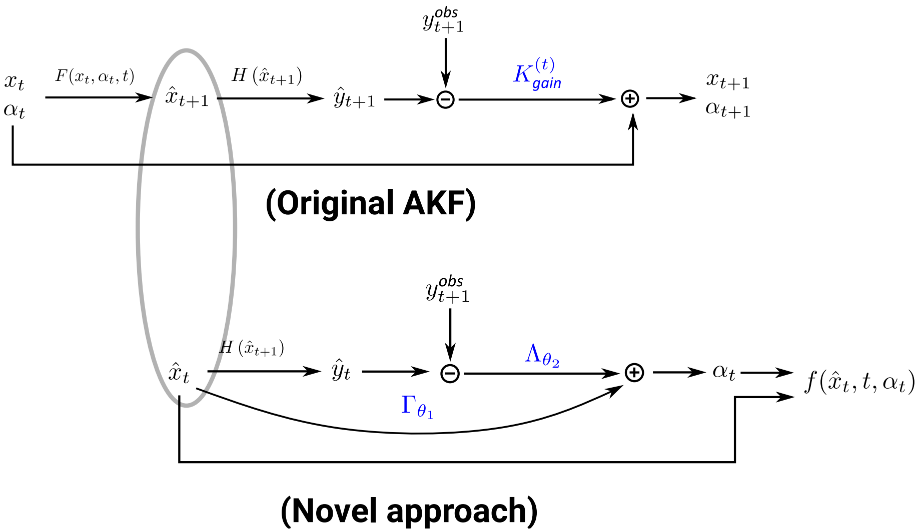


(Original AKF)

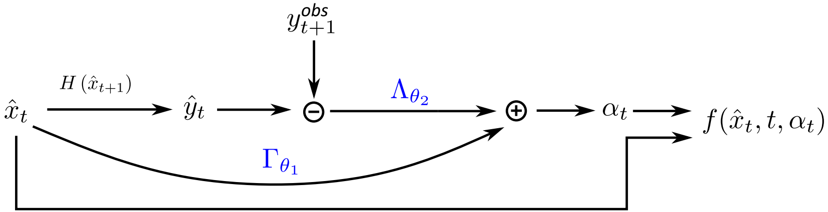


(Novel approach)

Neural networks and Kalman filters



Neural networks and Kalman filters



(Novel approach)

$$\min_{\theta_1,\theta_2} \sum_t \left\| H(\hat{x}_t) - y_t^{obs} \right\|_2^2 + R(\hat{x}, y^{obs}, \theta_1, \theta_2)$$

subject to (**Optimisation**)

$$\dot{\hat{x}} = f(\hat{x}, \alpha_t, t).$$

Numerical implementation

(1) Choose random initial θ and solve ODE.
(2) Compute loss.
(3) Collect gradients w.r.t. θ: **adjoint sensitivity** or **automatic differentiation**
(4) Update weights. Repeat (1-3) till parameters stabilize.

Chen et al. Neural ordinary differential equations. *Advances in neural information processing systems* (2018)

Numerical implementation

(1) Choose random initial θ and solve ODE.

(2) Compute loss.

(3) Collect gradients w.r.t. θ: **adjoint sensitivity** or **automatic differentiation**

(4) Update weights. Repeat (1-3) till parameters stabilize.

Efficiently implemented in **DiffEqFlux.jl** (adjoint sensitivity) and **Zygote.jl** (automatic differentiation) in Julia.

Chen et al. Neural ordinary differential equations. *Advances in neural information processing systems* (2018)

Example: FitzHugh-Nagumo

$$\dot{v} = c\left(v - \frac{v^3}{3} + cr\right)$$

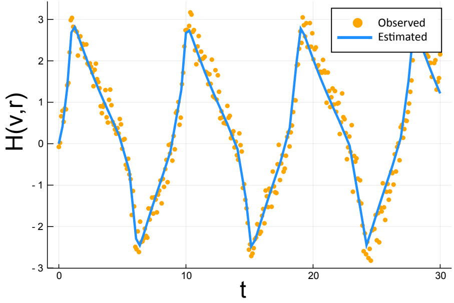
$$\dot{r} = \frac{-1}{c}(v - a + br)$$

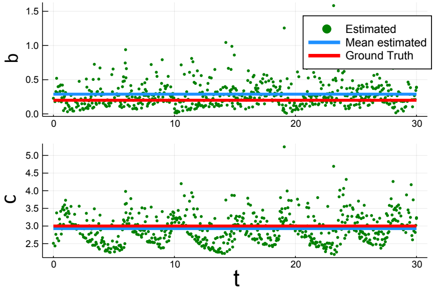
$\Gamma_{\theta_1} \longrightarrow$ 1×10 ReLU — 10×10 ReLU — 10×2

$\Lambda_{\theta_2} \longrightarrow$ 1×10 ReLU — 10×10 ReLU — 10×2

$$H(v, r) = v + r. \qquad \textbf{(Observation)}$$

$$R(\hat{x}, \alpha) = \left\|\dot{\hat{x}} - f(\hat{x}, \bar{\alpha})\right\|_2^2 \qquad \textbf{(Regularization)}$$



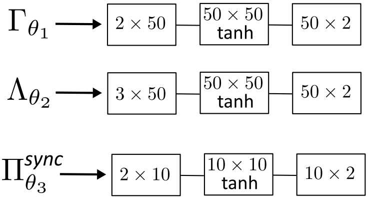


Example: Lorenz63

$$\dot{x} = \sigma(y - x),$$
$$\dot{y} = x(\rho - z) - y,$$
$$\dot{z} = xy - \beta z.$$

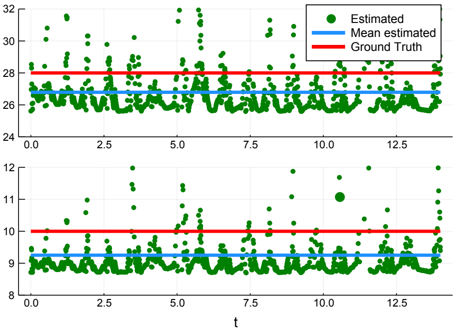
$$H(x, y, z) = (x, y).$$

(Observation)



Avoid using large NN to cover state space by adding synchronisation,

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} \mapsto \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} + \Pi^{sync}_{\theta_3}(\mathbf{x} - \mathbf{x}_{obs}).$$



Summary

Results:

- A neural-network based parameter estimation scheme in continuous models
- Generalizable and computationally efficient
- Extends well to partial observations and nonlinear systems

Limitations

- Existence and stability
- Observability of states and parameters
- Model tuning

Thanks! Open to questions.