

Chapter 3 - System Modeling

July 9, 2015

Introduction

We can derive the mathematical model of a dynamic system in **two ways** mainly:

① Physical Modeling:

Applying the laws of physics, chemistry, thermodynamics,...
Also called modeling from *First Principles*

- Sometimes these are non-linear. Lots of methods of this course require linear systems. Therefore **linearization** is needed.

e.g. $\sin(\theta) \sim \theta, \theta \rightarrow 0$

② System identification or *Empirical Modeling*:

Developing models from observed or collected data

Main classes of System identification methods

White box modeling: based on first principles.

→ known equations (structure) & parameters (coefficients).

Grey box identification: first principles & experimentation.

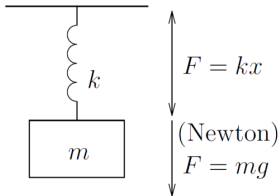
→ known equations, unknown/uncertain parameters.

Black box identification: based on experimentation.

→ unknown equations & unknown parameters.

Most popular approaches are forms of black box identification.

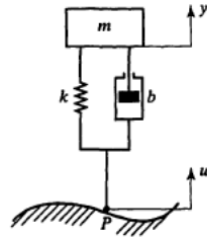
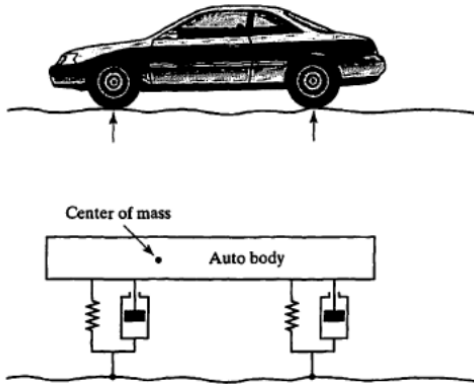
Example 1: Mass-Spring System



If spring is at rest at $x = 0$:

$$m \cdot \frac{d^2x}{dt^2} + k \cdot x = m \cdot g$$

Example 2: Mass-Spring Damped



Force exerted by damper: $F = b\dot{x}$

Differential equation can be found by writing force equilibrium and moment equilibrium around center of mass

Example 3: Pendulum

Dynamic equilibrium:

$$I\ddot{\theta}(t) = -mg\frac{L}{2}\sin(\theta(t)) \text{ with } I = \frac{mL^2}{3}$$

$$\ddot{\theta}(t) = -\frac{3g}{2L}\sin(\theta(t))$$

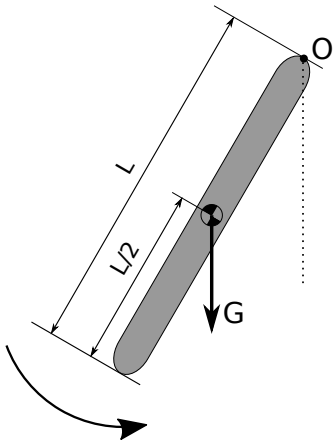
Small deviation of $\theta(t)$:

$$\ddot{\theta}(t) = -\frac{3g}{2L}\theta(t)$$

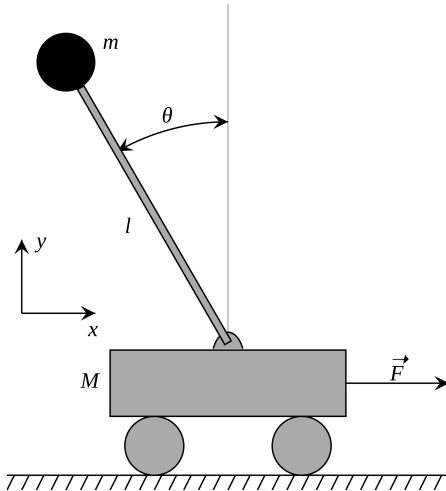
Solving the differential equation yields the general solution:

$$\theta(t) = A\cos(\omega_0 t + \phi) \text{ with } \omega_0 = \sqrt{\frac{3g}{2L}}$$

and ϕ & A to be determined with the initial condition

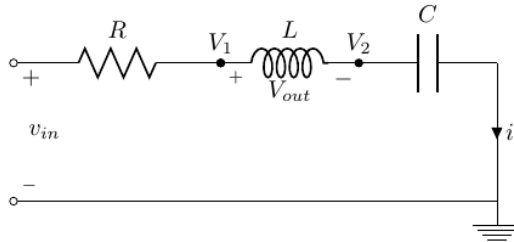


Example 4: Inverted Pendulum



Analysis can be done with Newton like former example, but less tedious is using energy-methods (Lagrange)

Example 5: RLC Circuit



Besides input v_{in} , two internal variables are needed to determine output \Rightarrow Second-order System

Inputs	Outputs	Chosen States
v_{in}	v_{out}	V_2 i

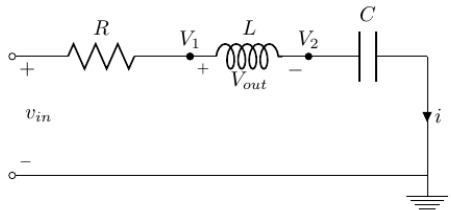
Example 5: RLC Circuit

Equations for each
component:

$$i = \frac{V_{in} - V_1}{R} \quad (\text{Ohm's law})$$

$$V_1 - V_2 = L \cdot \frac{di}{dt} \quad (\text{Coil})$$

$$i = C \cdot \frac{dV_2}{dt} \quad (\text{Capacitor})$$



Example 5: RLC Circuit

- Writing derivatives of state variables in function of state variables and inputs:
$$\begin{cases} \frac{di}{dt} = \frac{V_1 - V_2}{L} = \frac{V_{in} - Ri - V_2}{L} \\ \frac{dV_2}{dt} = \frac{i}{C} \end{cases}$$
- Writing output in function of state variables and inputs:
$$V_{out} = V_1 - V_2 = V_{in} - Ri - V_2$$

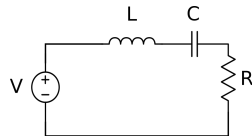
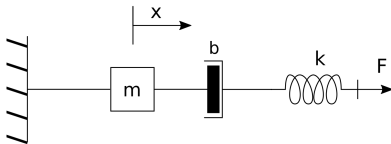
State Space Representation

This yields the **State Space Representation** of the dynamic system. In Matrix form:

$$\begin{bmatrix} \frac{dV_2}{dt} \\ \frac{di}{dt} \end{bmatrix} = \begin{bmatrix} 0 & 1/C \\ -1/L & -R/L \end{bmatrix} \begin{bmatrix} V_2 \\ i \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} V_{in}$$

$$V_{out} = \begin{bmatrix} -1 & -R \end{bmatrix} \begin{bmatrix} V_2 \\ i \end{bmatrix} + V_{in}$$

Force-Voltage Analogy



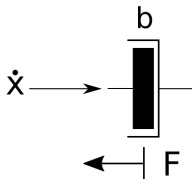
Let:

F	\leftrightarrow	V
\dot{x}	\leftrightarrow	i
x	\leftrightarrow	q

Force-Voltage Analogy

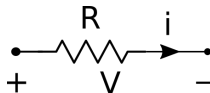
The analogy between the other quantities follows from comparing the physical laws.

Damping:



$$F = b\dot{x}$$

Resistance:

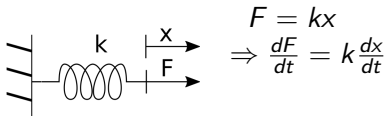


$$V = Ri$$

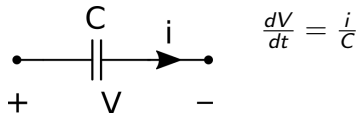
$$b \leftrightarrow R$$

Force-Voltage Analogy

Spring:

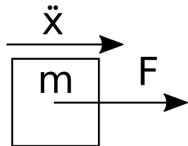


Capacitor:



$$k \leftrightarrow \frac{1}{C}$$

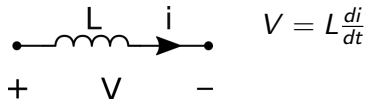
Newton:



$$F = m\ddot{x}$$

$$= m \frac{d\dot{x}}{dt}$$

Coil:



$$m \leftrightarrow L$$

Example 6: Hoover dam

Define:

- Inflow of water: $u(t)$
- Current volume of water: $x(t)$
- Outflow of water: $y(t)$
- Water level: $h(t)$

Assume that $x(t) = c_1 \cdot h(t)$

What will happen when we open the gate?



Example 6: Hoover dam

- Outflow depends on height:

$$y(t) = c_2 \cdot h(t)$$

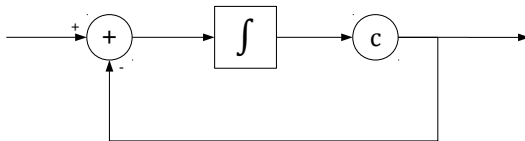
- The state of the system is defined by the contained volume of water:

$$\dot{x}(t) = u(t) - y(t) = u(t) - c_2 \cdot h(t)$$

- Thus a **State Space Representation** is, with $c \triangleq \frac{c_2}{c_1}$:

$$\dot{x}(t) = u(t) - c \cdot x(t)$$

$$y(t) = c \cdot x(t)$$



Nonlinear systems

In this course we focus on the linear state-space representation:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t), \\ y(t) = Cx(t) + Du(t). \end{cases} \quad \begin{cases} x[k+1] = Ax[k] + Bu[k], \\ y[k] = Cx[k] + Du[k]. \end{cases}$$

Most real life systems involve nonlinearity:

$$\begin{cases} \dot{x}(t) = f(x(t), u(t)), \\ y(t) = g(x(t), u(t)), \end{cases}$$

where f and/or g contain some nonlinearity, such as:

- *powers*: e.g. $\dot{x}(t) = Ax(t) + Bu(t) + \gamma u(t)^2$,
- *interactions*: e.g. $\dot{x}(t) = Ax(t) + Bu(t) + \gamma x(t)u(t)$,
- *clipping*: e.g. $\alpha \leq x(t) \leq \beta$.

Linearization around equilibrium point

Nonlinear systems have (several) equilibrium points x_e, u_e, y_e :

$$\begin{cases} \dot{x}_e = f(x_e, u_e) = 0, \\ y_e = g(x_e, u_e). \end{cases}$$

Linearizing in the region of (x_e, u_e, y_e) :

$$x = x_e + \Delta x, \quad u = u_e + \Delta u, \quad y = y_e + \Delta y,$$

with Δx , Δu and Δy *sufficiently* small.

Linearizing is done via **first order Taylor expansions**.

Linearization around equilibrium points

Linearizing is done via **first order Taylor expansions**.

$$\begin{cases} \frac{dx}{dt} = \frac{d(x_e + \Delta x)}{dt} = \frac{d\Delta x}{dt} = f(x, u) = f(x_e + \Delta x, u_e + \Delta u), \\ y_e + \Delta y = g(x, u) = g(x_e + \Delta x, u_e + \Delta u). \end{cases}$$

We write the *vectors* x and u in their individual components to simplify interpretation:

$$\dot{x}_1 = f_1(x_1, \dots, x_n, u_1, \dots, u_l)$$

$$\vdots$$

$$\dot{x}_n = f_n(x_1, \dots, x_n, u_1, \dots, u_l)$$

$$\dot{y}_1 = h_1(x_1, \dots, x_n, u_1, \dots, u_l)$$

$$\vdots$$

$$\dot{y}_l = h_l(x_1, \dots, x_n, u_1, \dots, u_l)$$

Linearization around equilibrium points

The first order Taylor expansion of $f()$ around (x_e, u_e) is described by the **Jacobian Matrix**:

$$\frac{dx}{dt} = f(x_e, u_e) + \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} & \frac{\partial f_1}{\partial u_1} & \cdots & \frac{\partial f_1}{\partial u_l} \\ \vdots & & & & & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} & \frac{\partial f_n}{\partial u_1} & \cdots & \frac{\partial f_n}{\partial u_l} \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \vdots \\ \Delta x_n \\ \Delta u_1 \\ \vdots \\ \Delta u_l \end{bmatrix}$$

With the partial derivatives evaluated in x_e and u_e
 $f(x_e, u_e) = \frac{dx_e}{dt} = 0$ because we *choose* x_e and u_e to be equilibrium points

Linearization around equilibrium points

This can be split up in a contribution by the state x and the input u :

$$\frac{d\Delta x}{dt} = \underbrace{\begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}}_A \begin{bmatrix} \Delta x_1 \\ \vdots \\ \Delta x_n \end{bmatrix} + \underbrace{\begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \cdots & \frac{\partial f_1}{\partial u_l} \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial u_1} & \cdots & \frac{\partial f_n}{\partial u_l} \end{bmatrix}}_B \begin{bmatrix} \Delta u_1 \\ \vdots \\ \Delta u_l \end{bmatrix}$$

Similarly C & D can be constructed from the Jacobian Matrix of $h(x, u)$

Example: decalcification plant

Used to reduce concentration of calcium hydroxide in water:

- chemical reaction: $\text{Ca}(\text{OH})_2 + \text{CO}_2 \rightarrow \text{CaCO}_3 + \text{H}_2\text{O}$
- reaction speed: $r = c[\text{Ca}(\text{OH})_2][\text{CO}_2]$
- rate of change of concentration:

$$\begin{aligned}\frac{d[\text{Ca}(\text{OH})_2]}{dt} &= \frac{k}{V} - \frac{r}{V}, \\ \frac{d[\text{CO}_2]}{dt} &= \frac{u}{V} - \frac{r}{V},\end{aligned}$$

with inflow rates k and u in mol/s and tank volume V in L.

- input u : inflow of CO_2 , output: $[\text{Ca}(\text{OH})_2]$

Nonlinear model and equilibrium point

Nonlinear model for the given reactor:

$$\begin{aligned}\frac{d[Ca(OH)_2]}{dt} &= \frac{k}{V} - \frac{c}{V}[Ca(OH)_2][CO_2], \\ \frac{d[CO_2]}{dt} &= \frac{u}{V} - \frac{c}{V}[Ca(OH)_2][CO_2], \\ y &= [Ca(OH)_2],\end{aligned}$$

with two state variables: $x_1 = [Ca(OH)_2]$ and $x_2 = [CO_2]$.

The equilibrium point $(k_{eq}, u_{eq}, x_{1,eq}, x_{2,eq}, y_{eq})$ of this system is:

$$\begin{aligned}\frac{k_{eq}}{V} - \frac{c}{V}[Ca(OH)_2]_{eq}[CO_2]_{eq} &= 0, \\ \frac{u_{eq}}{V} - \frac{c}{V}[Ca(OH)_2]_{eq}[CO_2]_{eq} &= 0.\end{aligned}$$

Linearization of the decalcification plant

For small deviations near the equilibrium:

$$\begin{aligned}\frac{d\Delta x_1}{dt} &= -\frac{c}{V}[CO_2]_{eq}\Delta x_1 - \frac{c}{V}[Ca(OH)_2]_{eq}\Delta x_2, \\ \frac{d\Delta x_2}{dt} &= -\frac{c}{V}[CO_2]_{eq}\Delta x_1 - \frac{c}{V}[Ca(OH)_2]_{eq}\Delta x_2 + \frac{\Delta u}{V}, \\ \Delta y &= \Delta x_1.\end{aligned}$$

The resulting linear state-space model is $\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases}$.

$$\begin{bmatrix} \frac{d[Ca(OH)_2]}{dt} \\ \frac{d[CO_2]}{dt} \end{bmatrix} = - \begin{bmatrix} \frac{c}{V}[CO_2]_{eq} & \frac{c}{V}[Ca(OH)_2]_{eq} \\ \frac{c}{V}[CO_2]_{eq} & \frac{c}{V}[Ca(OH)_2]_{eq} \end{bmatrix} \begin{bmatrix} [Ca(OH)_2] \\ [CO_2] \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{V} \end{bmatrix} u(t)$$

$$y(t) = [Ca(OH)_2]$$

Grey box identification: conceptual

Grey box identification starts from a known model structure but with unknown/uncertain parameters \leftrightarrow **parametric statistics**.

We assume linear, continuous time state space representation:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t), \\ y(t) = Cx(t) + Du(t). \end{cases}$$

Given: states, inputs, outputs and guesstimates of \tilde{A} , \tilde{B} , \tilde{C} & \tilde{D} .

Task: estimate \hat{A} , \hat{B} , \hat{C} and \hat{D} adequately via experiments.

"All models are wrong, but some are useful." - George E. P. Box

Linear regression

Consider input matrix \mathbf{X} , output vector \mathbf{y} and residuals ϵ :

$$\mathbf{X}\theta = \mathbf{y} + \epsilon.$$

The parameter vector θ must be estimated, given the observations.

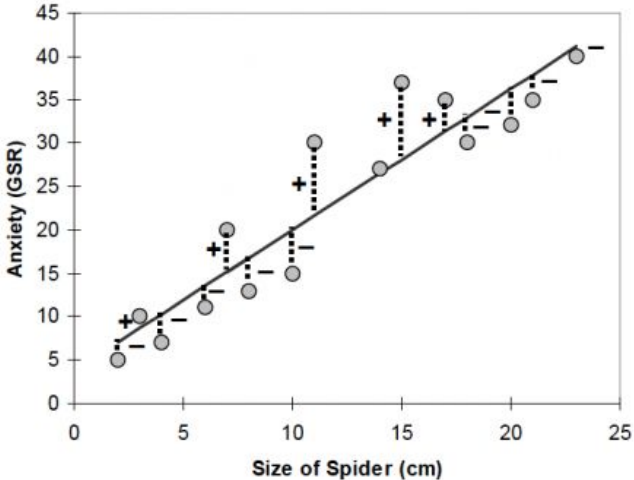
A common estimation approach is ordinary least squares (OLS):

$$\begin{aligned}(\mathbf{X}^T \mathbf{X}) \hat{\theta}_{OLS} &= \mathbf{X}^T \mathbf{y}, \\ \hat{\theta}_{OLS} &= (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}.\end{aligned}$$

The OLS estimate minimizes the sum-of-squares of errors, i.e.:

$$\hat{\theta}_{OLS} = \arg \min_{\theta} \sum_{i=1}^N \left(y(i) - \sum_{j=1}^d X(i,j) \theta(j) \right)^2$$

Linear regression with ordinary least squares



Maximum likelihood estimation

The maximum likelihood estimate $\hat{\theta}_{ML}$ is the parameter vector that maximizes the likelihood $\mathcal{L}(\cdot)$ of observing the (known) outputs \mathbf{y} , given the (known) inputs \mathbf{X} :

$$\hat{\theta}_{ML} = \arg \max_{\theta} \mathcal{L}(\mathbf{y}, \mathbf{X} \mid \theta)$$

For some structures, ML estimate can be obtained in closed form.

Example: least squares estimators are the maximum likelihood estimators if the associated residuals ϵ are normally distributed.

Maximum a posteriori (MAP) estimation

Bayesian: maximum likelihood estimation with a *prior* $p(\theta)$.

→ MAP estimation is a regularization of ML estimation

Bayes' theorem: $P(A | B) = P(B | A) \cdot P(A) / P(B)$.

If a prior distribution $p(\cdot)$ is available for θ , then the posterior distribution for θ becomes:

$$\theta \mapsto \mathcal{L}(\theta | \mathbf{y}, \mathbf{X}) = \frac{\mathcal{L}(\mathbf{y}, \mathbf{X} | \theta)p(\theta)}{\int_{\vartheta} \mathcal{L}(\mathbf{y}, \mathbf{X} | \vartheta)p(\vartheta)d\vartheta}.$$

The MAP estimate is the mode of the posterior distribution of θ :

$$\hat{\theta}_{MAP} = \arg \max_{\theta} \mathcal{L}(\mathbf{y}, \mathbf{X} | \theta)p(\theta).$$

Errors-in-variables approach

Additionally accounts for **measurement errors in inputs**.

↔ standard regression only accounts for **errors in outputs**

Typically described via *latent variables*:

$$\begin{cases} x = x^* + \eta, \\ y = y^* + \epsilon, \\ y^* = g(x^* \mid \theta), \end{cases}$$

with x , y the observed inputs, outputs and latent variables x^* , y^* .

Assumption: latent variables x^* and y^* exist which follow the true functional relationship $g(\cdot)$.

Task: estimate θ .

Black box identification

Start from unknown equations & unknown parameters.

→ related to **machine learning** and **nonparametric statistics**.

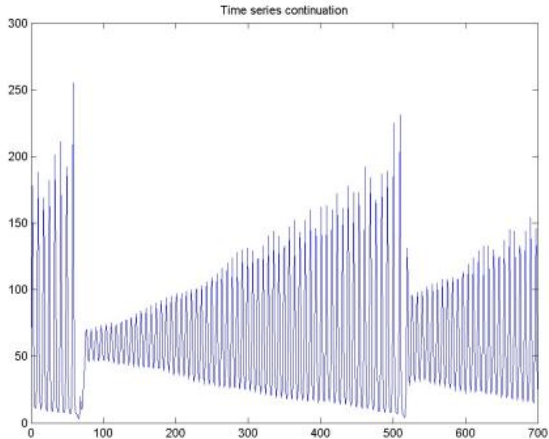
If we assume a linear state space system:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t), \\ y(t) = Cx(t) + Du(t). \end{cases} \quad \begin{cases} x[k+1] = Ax[k] + Bu[k], \\ y[k] = Cx[k] + Du[k]. \end{cases}$$

Black box identification deals with:

- unknown states, both in number & physical interpretation
→ dimensions of A , B & C unknown
- unknown parameters (values in A , B , C , D)

Time series: Santa Fe laser



Modelling the Santa Fe laser

This laser can be treated as an autonomous discrete time system:

$$\begin{cases} x[k+1] = f(x[k-N+1], \dots, x[k]), \\ y[k] = x[k]. \end{cases}$$

The output depends on the past N states & no inputs.

→ how large is N ? → **unknown structure**

Treat it as a regression problem with N inputs: $y = f(X_1, \dots, X_N)$.

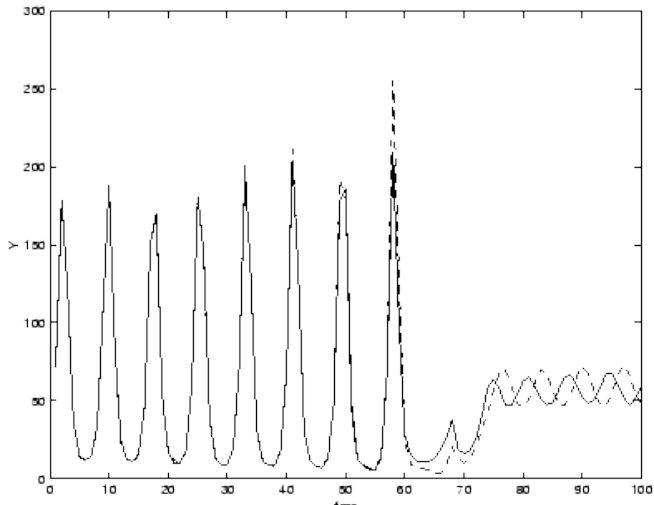
→ lets say linear, i.e. $y = \mathbf{X}\theta$ → **unknown parameters** $\theta \in \mathbb{R}^N$.

→ for given N , we can estimate θ via grey box methods.

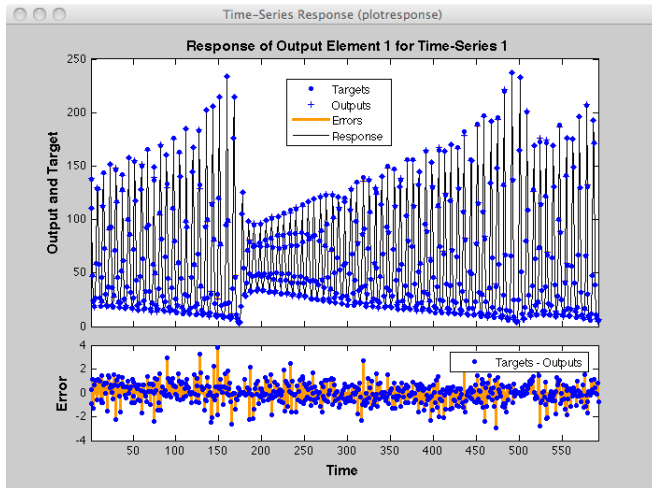
Nonlinear models can be obtained via machine learning methods.

→ neural networks, support vector machine, random forest, ...

Predictions of a least-squares support vector machine



Predictions of an artificial neural network



Neural network: biological

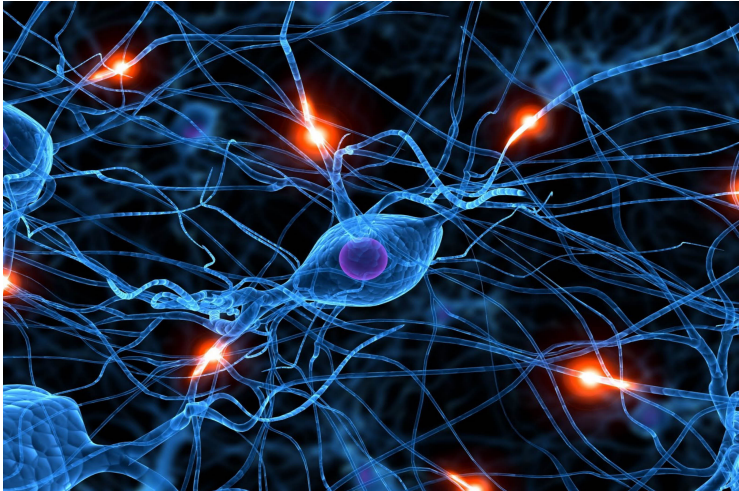


Image taken from <http://www.extremetech.com/wp-content/uploads/2013/09/340.jpg>.

Structure of a single neuron

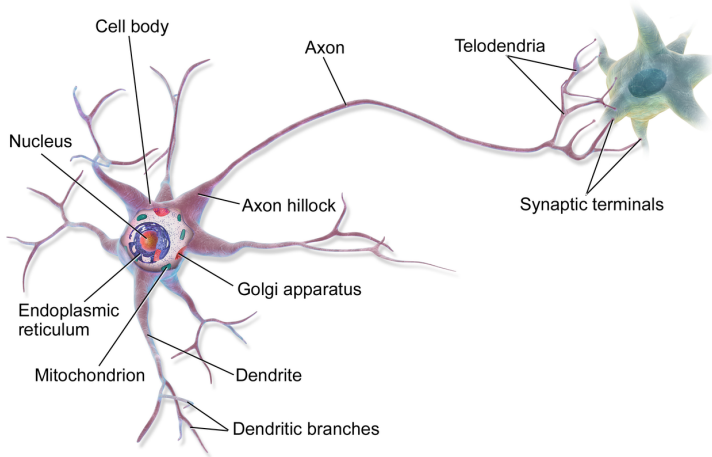


Image taken from http://en.wikipedia.org/wiki/File:Blausen_0657_MultipolarNeuron.png.

Neural network: artificial

