A walk through the intersection between machine learning and mechanistic modeling

Data Science Seminar, EURECOM, France. May 3, 2018
Juan Pablo Carbajal juanpablo.carbajal@eawag.ch
Eawag: Swiss Federal Institute of Aquatic Science and Technology

This work is licensed under a Creative Commons “Attribution-ShareAlike 4.0 International” license.
Background
Machine Learning and Mechanistic modeling

Machine Learning
- Discover patterns from data (prediction)
- Minimal processes knowledge

Emulation
- Discover "useful" approximation of numerical model from data
- Partial/Fuzzy (sub)processes knowledge

Mechanistic modeling
- Interpret data/model using model/data
- Detailed (sub)processes knowledge
Environmental science and engineering
Model calibration, Fast (extreme) event prediction, & Design

source: https://www.dailyrecord.co.uk/news/scottish-news/storm-frank-rescuers-brave-storm-7096067
Engineered complex systems
Optimization, Control, & Maintenance

source: Leaflet, CC BY-SA 3.0, commons.wikimedia.org/w/index.php?curid=5704247
Data fusion/interpretation

source: Azzimonti et al. 10.1080/01621459.2014.946036
What is emulation?

How is emulation even possible?
Alien researcher

What is emulation?

source: Svjo, CC BY-SA 3.0, commons.wikimedia.org/w/index.php?curid=25795521
What is emulation?
From molecules to fluids

source: doi:10.1088/0965-0393/12/6/R01

What is emulation?
Emulation
many names & similarities ...

Metamodelling

Surrogate modeling

Coarse graining

Model order reduction

Interpolation

Phenom. modeling

Empiric. modeling

Extrapolation

Response surface
Types of simulators
Partial Differential Equations (PDE)

- Fluid dynamics
- Convection-Diffusion-Reaction systems
- Heat transfer
- Multi-physics, e.g. fluid-structure interaction
- ...

Ordinary Differential Equations (ODE)

- Biochemical reactions
- Interconnected tanks
- Population dynamics
- Mechanics (Lagrangian/Hamiltonian)
- ...
Algebraic equations

- Equilibrium solutions of PDE/ODE
- Conservation laws (e.g. mass (Stoichiometry), energy, ...)
- Empirical formulas
- Data-driven models
- ...

source: doi:10.1039/C4CY00409D
Agent based systems
Description of individual (groups of) agents and their interaction

- Sociology
- Biology
- Epidemiology
- Network science
- Molecular dynamics
- ...

source: www.youtube.com/watch?v=GUKjC-69vaw
Cellular automata
agents in a graph, discrete states

bullet Agents correspond to nodes in a grid (graph)
bullet The agents contain one or more discrete internal states
bullet Each agent interacts with set of nodes: neighborhood
bullet Rule-set to change states according to neighborhood

source: www.youtube.com/watch?v=FaqC4h5Ftg
Gas lattices (Boltzmann)
"continuous" state cellular automata

Time $t$  

Post-collision $f_{out}$  

source: www.coursera.org/learn/
modeling-simulation-natural-processes

Time $t + \delta t$  

Pre-collision $f_{in}$  

- Probabilistic: collisions redistribute pdf
- Observables = moments of pdf

- Incompresible Navier-Stokes
- Natural multiphysics (fluid-structure)
Combination of models
Dimensional heterogeneous models

- Some details are less relevant
- "Simplify" those
  - 1D phenomenological
  - 1D Emulator of detailed model
  - Reduced model
- Coupling is challenging
Technical background
Intra-, extra-, inter-polation

interpolation $\neq$ smoothing/approximation

intrapolation $\neq$ extrapolation
The Regression problem

Dataset: \( \{x, y\} \)
Hypotheses set: \( \mathcal{H} \)

\[
\hat{f} = \arg\min_{f \in \mathcal{H}} \text{Loss}(f[x], y; \theta)
\]
The **Constrained** Regression problem

Dataset: \( \{x, y\} \)

Hypotheses set: \( \mathcal{H} \)

(nonlinear) Operator: \( D \)

\[
\hat{f} = \arg\min_{f \in \mathcal{H}} \text{Loss}(f[x], y; \theta)
\]

s. t.

\[
D(f; \mu) = 0
\]

\[
g(f; \mu) = 0
\]

\[
h(f; \mu) \leq 0
\]
The **Regularized** Regression problem

Dataset: \( \{x, y\} \)

Hypotheses set: \( \mathcal{H} \)

(nonlinear) Operator: \( \mathcal{D} \)

\[
\hat{f} = \arg \min_{f \in \mathcal{H}} \text{Loss}(f[x], y; \theta) + \kappa_D \|\mathcal{D}(f; \mu)\| + \kappa_g \|g(f; \mu)\| + \kappa_h \|m(h(f; \mu))\|
\]
The **Regularized and Constrained** Regression problem

Dataset: \( \{ x, y \} \)

Hypotheses set: \( \mathcal{H} \)

(nonlinear) Operator: \( \mathcal{D} \)

\[
\hat{f} = \arg\min_{f \in \mathcal{H}} \text{Loss}(f[x], y; \theta) \\
+ \kappa_{\mathcal{D}} \| \mathcal{D}(f; \mu) \| \\
\text{s. t.} \\
g(f; \mu) = 0 \\
h(f; \mu) \leq 0
\]
Regularized Regression
Linear Differential Operators
Gaussian Processes
Kalman Filters
Regularized Regression (RR) with quadratic (convex) Loss function

Given design data set with input-output values \( \{t_i, y_i\} = (T, y(T)) \), find \( f(t) \) that approximates the data and provides good predictions for unseen \( t \):

\[
\min_f \sum_{j=1}^{N} (y_j - f(t_j))^2 + \kappa ||Rf||^2
\]

\( f(t) \) should be close to the design data, but it should be regular according to \( R \).
Regularized Regression (RR) with quadratic (convex) Loss function

$N$-degree polynomial regression on $N$ points with

$$\| Rf \|^2 = \int \left( \frac{d^2 f}{dt^2} \right)^2$$
Regularization operator\(^1\)

\[
\begin{align*}
\min_f \sum_{j=1}^{N} (y_j - f(t_j))^2 + \kappa \|Rf\|^2 \\
J[f] = \sum_{j=1}^{N} \left( y_j - \int f(t)\delta(t - t_j)dt \right)^2 + \kappa \langle Rf(t), Rf(t) \rangle
\end{align*}
\]

Searching for the critical point of that functional leads to

\[
R^\dagger R f(t) = \sum_{j=1}^{N} \frac{y_j - f_j}{\kappa} \delta(t - t_j)
\]

\[
R^\dagger R G(t, t') = \delta(t - t'),
\]

\(^1\)Tomaso Poggio and F. Girosi (1990). “Networks for approximation and learning.” In: Proceedings of the IEEE 78.9, pp. 1481–1497. ISSN: 0018-9219. DOI: 10.1109/5.58326.
Regularization operator\textsuperscript{1}

\[
\min_f \sum_{j=1}^{N} (y_j - f(t_j))^2 + \kappa \|Rf\|^2
\]

Solution

\[
f(t) = G(t, T) \left( G(T, T) + \kappa I \right)^{-1} (y(T) - n(T)) + n(t).
\]

Where \(G(t, t')\) is the Green’s function of the operator \(R^\dagger R\). If \(R\) has a Green’s function \(g(t, t')\) (and \(R^\dagger, g^\dagger(t, t')\)), then

\[
G(t, t') = \int g(t, u)g^\dagger(u, t')du
\]

\textsuperscript{1}Tomaso Poggio and F. Girosi (1990). “Networks for approximation and learning.” In: Proceedings of the IEEE 78.9, pp. 1481–1497. ISSN: 00189219. DOI: 10.1109/5.58326.
Gaussian processes (GP): characterization

GP is a distribution of functions defined by a mean function $m(t)$ and a covariance function $k(t, t')$.

Prior GP conditioned on a design data set $\{t_i, y_i\} = (T, y(T))$, gives posterior GP.

**mean of posterior GP**

$$f(t) = k(t, T) \left(k(T, T) + \kappa I\right)^{-1}(y(T) - m(T)) + m(t).$$

The evaluated covariance function $k(T, T)$ is the covariance matrix.
Comparing GP and RR

RR solution
\[ f(t) = G(t, T) \left( G(T, T) + \kappa I \right)^{-1} \left( y(T) - n(T) \right) + n(t). \]

GP predictive mean
\[ f(t) = k(t, T) \left( k(T, T) + \kappa I \right)^{-1} \left( y(T) - m(T) \right) + m(t). \]
Example: ODE

1st order linear ODE with piece-wise constant input and random input, giving distribution of functions

\[ L f(t) - u(t) = \eta(t) \]

\[ \eta(t) \sim \mathcal{N} \left( 0, \Sigma \delta(t - t') \right) \]
Example: ODE
(non-stationary) Covariance function
Example: ODE

\[ f(t) = k(t, t_i) \left( k(t_i, t_i) \right)^{-1} \hat{f}(t_i) + L^{-1}u(t) \]
Relations between GP and RR


\begin{itemize}
  \item Gaussian process $p_K(f)$
  \item zero mean
  \item kernel function $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$
  \item pos. def.
  \item covariance operator $K : \mathcal{H} \rightarrow \mathcal{H}$
  \item sym. pos. def.
  \item RKHS $\| \cdot \|_K$
  \item inner prod.
  \item regularization operator $R : \mathcal{H} \rightarrow \mathcal{G}$
  \item one-to-one
\end{itemize}

\footnote{Florian Steinke and Bernhard Schölkopf (Nov. 2008). “Kernels, regularization and differential equations.” In:}
Relations between GP and RR

Kalman filters (KF): characterization

An iterative method ($O(Tn^3)$) to predict (hidden) states of a $n$-state space dynamic model

\[
\dot{x}(t) = A x(t) + B u(t) + \nu(t)
\]
\[
y(t_k) = H x(t_k) + D u(t_k) + \epsilon(t_k)
\]

$x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, $u \in \mathbb{R}^l$

$\nu \sim \mathcal{GP}(0, Q)$, $\epsilon \sim \mathcal{N}(0, P)$

This is the continuous-time model discrete-time measurements flavor of KF. It can be extended to nonlinear (and non-Gaussian) systems: the Extended KF (model linearization), or the Unscented KF (posterior approximation).
**KF: algorithm structure**

1. **State update**
   1. Simulate forward: predict state.
   2. Propagate state error covariance.

2. **Measurement update**
   1. Compute Kalman gain.
   2. Update state estimate with data.
   3. Update state error covariance.

Initial values

observation
"... all [reviewers] mention the affinity between my results ... and the Kalman filter, ... If only the work in these fields were more readily accessible to the statistician who (like me) is not a specialist pure mathematician in terms familiar to him, much duplication could be avoided. In fact I explicitly denied any originality for these result."

Relations between GP and KF

• Evidence of GP and KF equivalence based on optimal estimation of solution (O’Hagan, Steinke & Schölkopf, ...)

• Stationary covariance matrices can be converted to state-space models, where KF can be applied\(^3\), via Wiener-Khinchin theorem

• KF is extended to non-Gaussian likelihoods\(^4\)

---


Summary: Linear operators

- KF, GP and RR are unified (linear, stationary, convex loss-function): interpretability, algorithmic versatility
- GP $\rightarrow$ KF: GP inference linear in the number of time samples
- RR $\rightarrow$ GP: non-parametric curve fitting

**Conclusion:** information provided as linear operators can be merged with ML!

Technical challenges: optimal algorithms, fast implementations, (sample) convergence rates, language abstractions, optimized hardware
Nonlinear operators
Nonlinear operators

Here it gets sketchy!
Differential operators that are "linear on the differential part"

\[ \mathcal{D}(x) = \mathcal{L}x - f(x, \theta) = 0 \]

We will use a surrogate for \( x \)

\[ \mathcal{D}(\phi) \simeq 0 \]

How to find the surrogate?
One alternative

\[ \phi_{n+1} = L^{-1}f(\phi_n, \theta) \]

\[ \phi_n = x \cdot \varphi \]
Thank you!

Q&A
Stationary covariance matrices can be converted to state-space models, where KF can be applied\(^5\), via Wiener-Khinchin theorem.

\[
\kappa(t - t') \propto \int S(\omega)e^{i\omega(t-t')}d\omega
\]

Concretely

Spatio-temporal GP representation

\[ f(r, t) \sim \mathcal{GP}(0, \kappa(r, t; r', t')) \]

\[ y_k = \mathcal{H}_k f(r, t_k) + \epsilon_k \]

\[ O((Tn)^3) \text{ general matrix inversion} \]

Linear stochastic partial differential equation

\[ \frac{\partial f(r, t)}{\partial t} = \mathcal{F} f(r, t) + Lw(r, t) \]

\[ y_k = \mathcal{H}_k f(r, t_k) + \epsilon_k \]

\[ O(Tn^3) \text{ infinite-dimensional Kalman filtering and smoothing} \]

1. Compute spectral density (SD) via Fourier transform of covariance
2. Approximate SD with a rational function (if necessary)
3. Find stable transfer function (poles in upper half complex plane)
4. Transform to SS using control theory methods
Example: isotropic Matérn covariance function

Temporal process

\[ \kappa(t - t') = \sigma^2 \lambda(t - t') K_1[\lambda(t - t')] \]

\[ \frac{\partial f(t)}{\partial t} = \begin{bmatrix} 0 & 1 \\ -\lambda & -2\sqrt{\lambda} \end{bmatrix} f(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} w(x, t) \]

Spatio-temporal process

\[ r = \| (x, t) - (x', t') \| \]

\[ \kappa(r) = \sigma^2 \lambda r K_1(\lambda r) \]

\[ \frac{\partial f(x, t)}{\partial t} = \begin{bmatrix} 0 & 1 \\ \frac{\partial^2}{\partial x} - \lambda & -2\sqrt{\lambda} - \frac{\partial^2}{\partial x} \end{bmatrix} f(x, t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} w(x, t) \]
Surrogate trajectory

Adomain polynomials RKHS Regularization
Other methods

Volterra-Wiener kernels