Structured Data

Learning, prediction, dependency, testing
Session 5 (Lecture) - Part I: structured output prediction

Florence d’Alché-Buc

Contact: florence.dalche@telecom-paristech.fr,
M2 Data Science, Télécom ParisTech & Ecole Polytechnique, Université of Paris-Saclay, France
# Table of contents

1. Motivation

2. Multi-task Regression

3. Operator-valued kernel-based Regression

4. Input Output Kernel Regression

5. recap

6. References
Previously, we saw:
End of scoring methods: - with the work about joint kernel maps (Blaschko and Lampert)
- with the use of MLP and deep learning instead of linear models (Chen et al.)
Regression tree methods extended to kernelized outputs
We showed an application to link prediction
Outline

Motivation

- Multi-task Regression
- Operator-valued kernel-based Regression
- Input Output Kernel Regression

recap

References
A very important task in real world machine learning applications:

- **Marketing applications of preference modeling**: where the same choice panel questions (the same $x$’s) are given to many individual consumers, each individual provides his/her own preferences (the $y$’s)

- **Multiple Drug activity prediction**: each drug is described by the same features, each coordinate of the target vector is the activity score of a drug on a specific diseases, some diseases are close, and we want to take into account the relationship between diseases
Multi-task regression

- **INPUT**: an object
- **OUTPUT**: multiple interdependent outputs
• **Structured multiple class classification**: a multiple class classification problem with relationship between classes like a hierarchy etc...

• **Multi-centered multiple class classification in medicine**: a multiple class classification problem with relationship between classes like a hierarchy AND different datasets coming from different hospitals
Reverse-engineering of a biological dynamical system

Network Inference

Infer the (causal) relationships between state variables from the observation of a noisy multivariate time series

Example: from time course of gene expression, extract a gene regulatory network
Outline

Motivation

Multi-task Regression

Operator-valued kernel-based Regression

Input Output Kernel Regression

recap

References
Multi-task prediction: formulation 1

Same attributes/ same input space, **same target variable** but observed data are not the same.

- $T$ prediction tasks to solve jointly from $T$ datasets
- fixed unknown probability distribution $\mathbb{P}_t(X, Y)$
- $(x_{ti}, y_{ti})$ i.i.d. from $\mathbb{P}_t$.
- Let $\mathcal{S}(t) = \{(x_{ti}, y_{ti}), i = 1, \ldots, n; t = 1, \ldots, T\}$
- How to learn simultaneously $T$ functions $h_t, t = 1, \ldots T$ to respectively predict $y_t$?

In the loss function, we incorporate a penalty that encourages the functions $h_t$ to be close to some.
Assume
\[ w_t = v_t + w_0 \] (1)
and each \( v_t \) is close to \( w_0 \)

**Multitask SVM (linear case)**

\[
\begin{align*}
\min_{w_0, v_t, \xi} & \quad \sum_{t=1}^{T} \sum_{i=1}^{n} \xi_{it} + \frac{\lambda_1}{T} \sum_{t=1}^{T} \|v_t\|^2 + \lambda_2 \|w_0\|^2 \\
\text{s.c.} \quad & \quad \forall i = 1, \ldots, n, \forall t = 1, \ldots, T \\
& \quad y_{it}(w_0 + v_t)^T x_{it} \geq 1 - \xi_{it} \\
& \quad \xi_{it} \geq 0
\end{align*}
\]
Optimal solution

\[ w_0^* = \frac{\lambda_1}{\lambda_1 + \lambda_2} \sum_t w_t^* \]

Proof:
\[ \mathcal{L}(w_0, v_t, \xi_t, \alpha_{it}, \gamma_{it}) = \]
\[ j(w_0, v_t, \xi_t) - \sum_{i,t} \alpha_{it} (y_{it}(w_0 + v_t)^T x_{it} - 1 + \xi_{it}) - \sum_{i,t} \gamma_{it} \xi_{it} \]
Setting the derivative of $\mathcal{L}$ to zero

\[
\begin{align*}
\mathbf{w}_0^* &= \frac{1}{2\lambda_2} \sum_{it} \alpha_{it} y_{it} x_{it} \\
\mathbf{v}_t^* &= \frac{T}{2\lambda_1} \sum_i \alpha_{it} y_{it} x_{it}
\end{align*}
\]
Multi-task classification: formulation 1’

Multitask SVM (linear), new formulation

\[
\min_{w_t, \xi} \left\{ \sum_{t=1}^{T} \sum_{i=1}^{n} \xi_{it} + \rho_1 \sum_{t=1}^{T} \|w_t\|^2 + \rho_2 \sum_{t=1}^{T} \|w_t - \frac{1}{T} \sum_{s=1}^{T} w_s\|^2 \right\}
\]
s.t. \( \forall i = 1, \ldots, n, \forall t = 1, \ldots, T \)

\[ y_{it} w_t^T x_{it} \geq 1 - \xi_{it} \]

\( \xi_{it} \geq 0 \)

where we have:

\[
\rho_1 = \frac{1}{T} \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}
\]

\[
\rho_2 = \frac{1}{T} \frac{\lambda_2^2}{\lambda_1 + \lambda_2}
\]

Therefore, \( w_0^* = \frac{\lambda_1}{T \lambda_2} \sum_t w_t^* \). This suggests a new formulation...

Ref: Evgeniou and Pontil, 2005.
Multi-task classification: formulation 1’

Multitask SVM (linear), new formulation

$$\min_{\mathbf{w}_t, \xi} \left\{ \sum_{t=1}^{T} \sum_{i=1}^{n} \xi_{it} + \rho_1 \sum_{t=1}^{T} \|\mathbf{w}_t\|^2 + \rho_2 \sum_{t=1}^{T} \|\mathbf{w}_t - \frac{1}{T} \sum_{s=1}^{T} \mathbf{w}_s\|^2 \right\}$$

s.c. $\forall i = 1, \ldots, n, \forall t = 1, \ldots, T$

$$y_{it}\mathbf{w}_t^T\mathbf{x}_{it} \geq 1 - \xi_{it}$$

$$\xi_{it} \geq 0$$

where we have:

$$\rho_1 = \frac{1}{T} \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}$$

$$\rho_2 = \frac{1}{T} \frac{\lambda_2^2}{\lambda_1 + \lambda_2}$$
Solving the new problem

• Apply classic theory of duality and find $T$ vectors $w_t$

• Or, use the following trick:
  • Define a function $H(x, t) = h_t(x)$
  • $H(t, x) = w^T \phi(x, t)$
  • $\phi((x, t)) = (\frac{x}{\sqrt{\mu}}, 0, \ldots, 0, x, 0 \ldots, 0)$
  • each $0$ is of dimension $d$: repeated $t - 1$ times, first and then, repeated, $T - t$ times.
  • $\mu$ reflects how much all the tasks are similar
  • $w = (\sqrt{\mu}w_0, v_1, \ldots, v_T)$
A single multi-task classifier

We have then:

$$\Phi((x, t)) = \left(\frac{x}{\sqrt{\mu}} \cdot \underbrace{0, \ldots, 0}_{t-1}, x, \underbrace{0, \ldots, 0}_{T-t}\right)$$

Now the problem boils down to learning a classic SVM with its classic dual formulation

$$w^T \phi((x, t)) = (\mu w_0 + v_t)^T x$$

$$\|w\|^2 = \mu \|w_0\|^2 + \sum_t \|v_t\|^2$$
A single multi-task classifier and its dual formulation

A convenient notation:

\[
\max_{\alpha} - \frac{1}{2} \sum_{ij,s,t} \alpha_{is} y_{is} \alpha_{jt} y_{jt} K_{st}(x_i, x_j) + \sum_{it} \alpha_{it}
\]

s.t. \( \forall i, t, 0 \leq \alpha_{it} \leq C. \)

**THEOREM 2.1.** Let \( C := \frac{T}{2\lambda_1}, \mu = \frac{T\lambda_2}{\lambda_1}, \) and define the kernel

\[
K_{st}(x, z) := \left( \frac{1}{\mu} + \delta_{st} \right) x \cdot z, \quad s, t = 1, \ldots, T.
\]
Nonlinear multi-task classification

\[ S = \{(x_i, t_i; y_i), i = 1, \ldots, n\} \]

**Dual formulation**

\[
\max_{\alpha} -\frac{1}{2} \sum_{ij} \beta_i y_i \beta_j y_j G((x_i, t_i), (x_j, t_j)) + \sum_i \alpha_i \\
\text{s.c. } \forall i, 0 \leq \alpha_i \leq C.
\]

**Link with pb in page 17**

Equivalent if: \( n = mT \) and

\[
\begin{align*}
x_i &= x(i \mod T)(i \mod m) \\
y_j &= y(i \mod T)(i \mod m)
\end{align*}
\]

and \( G(x_i, t_i), (x_j, t_j) = K_{t_i t_j}(x_i, x_j) \)
Let us define $k$, a PDS kernel and $\mathcal{H}_k$, the RKHS associated to that kernel $k$. $\| \cdot \|_k$ is the associated norm. NB: here all the functions work in the same input space

**Multi-task regression in RKHS**

Minimize for $h_1, \ldots, h_T \in \mathcal{H}_k$

$$
\sum_{t=1}^T \sum_{i=1}^n \ell(x_{it}, y_{it}, h_t(x_{it})) + \rho_1 \sum_{t=1}^T \|h_t\|_k^2 + \rho_2 \sum_{t=1}^T \|h_t - \frac{1}{T} \sum_{s=1}^T h_s\|_k^2
$$
Structured Multiple class classification

Again: let us look at the linear case. Now $M$ is a $T \times T$ matrix encoding the relationship between the $T$ tasks.

- Training set: $\mathcal{S} = \{(x_i, y_i = (y_{i1}, \ldots, y_{iT}), i = 1, \ldots n\}$

$$\min_{w_t, \xi_{it}} \left\{ \sum_{t=1}^{T} \sum_{i=1}^{n} \xi_{it} + \rho_1 \sum_{t=1}^{T} \|w_t\|^2 + \rho_2 \sum_{i,j=1}^{T} m_{uv} \|w_i - w_j\|^2 \right\} \text{s.c.} \forall i = 1, \ldots, n, \forall t = 1, \ldots, T, y_{it} w_{it}^T x_i \geq 1 - \xi_{it} \text{ and } \xi_{it} \geq 0$$
Structured Multi-task Regression in RKHS

Let us define $k$ a PDS kernel and $\mathcal{H}_k$ the RKHS associated to that kernel $k$. NB: here all the functions work in the same input space

**Multi-task regression in RKHS**

Minimize for $h_1, \ldots, h_T \in \mathcal{H}_k^T$

$$\sum_{t=1}^{T} \left\{ \sum_{i=1}^{n} \ell(x_i, y_{it}, h_t(x_i)) + \lambda_1 \|h_t\|^2_{k} \right\} + \sum_{i,j=1}^{T} m_{ij} \|h_i - h_j\|^2$$

where $M$ is a $T \times T$ matrix encoding the relationship between the $T$ tasks.
Outline

Motivation

Multi-task Regression

Operator-valued kernel-based Regression
  Reminder: scalar-valued case
  Operator-valued kernels
  Multi-task regression
  Other applications of operator-valued kernels

Input Output Kernel Regression

recap

References
Outline

Motivation

Multi-task Regression

Operator-valued kernel-based Regression
  Reminder: scalar-valued case
  Operator-valued kernels
  Multi-task regression
  Other applications of operator-valued kernels

Input Output Kernel Regression

recap

References
• Similarly to MLP or vector-valued regression trees, we would like to solve the problem 2 with a unique function $h$ with vectorial values.

• The RKHS theory extends to vector-valued functions: it uses operator-valued kernels instead of scalar-valued kernels.

• Micchelli and Pontil introduced this theory to the Machine Learning Community.
Outline

Motivation

Multi-task Regression

Operator-valued kernel-based Regression

Reminder: scalar-valued case

Operator-valued kernels

Multi-task regression

Other applications of operator-valued kernels

Input Output Kernel Regression

recap

References
Reminder: Kernel trick and feature map 1/2

Input Space

Feature Space

ϕ
Reminder: Kernel trick and feature map 2/2

\[ f(x) = \sum_{i=1}^{n} \alpha_i y_i \phi(x), \phi(x_i) > \mathcal{F} = \sum_{i=1}^{n} \alpha_i y_i k(x, x_i), \]
\[ g(z) = (\sum_i \alpha_i \phi(x_i))^T z \]
\[ f(x) = g \circ \phi(x) \]
\[ SVM: h(x) = \text{sign}(f(x) + b) \]
Definition (Reproducing Kernel Hilbert space - RKHS)

Let $\mathcal{H}$ be a Hilbert space of $\mathbb{R}$-valued functions on non-empty set $\mathcal{X}$. A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a **reproducing kernel** of $\mathcal{H}$, and $\mathcal{H}$ is a reproducing kernel Hilbert space if:

- $\forall x \in \mathcal{X}, k(\cdot, x) \in \mathcal{X}$,
- $\forall x \in \mathcal{X}, \forall f \in \mathcal{H}, \langle f(\cdot), k(\cdot, x) \rangle_\mathcal{H} = f(x)$ (**reproducing property**).

In particular, for any $x,y \in \mathcal{X}$,

$$k(x, y) = \langle k(\cdot, x), k(\cdot, y) \rangle_\mathcal{H}$$
Theorem (Reproducing Kernel Hilbert space induced by a kernel (Aronszajn, 1950))

Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a positive definite symmetric kernel. Then, there exists a Hilbert space $\mathcal{H}$ and a function $\phi : \mathcal{X} \to \mathcal{H}$ such that:

$$\forall (x, x') \in \mathcal{X} \times \mathcal{X}, \; k(x, x') = <\phi(x), \phi(x')>_{\mathcal{H}}$$

Furthermore, there exists a unique $\mathcal{H}$ that has the reproducing property:

$$\forall f \in \mathcal{H}, \forall x \in \mathcal{X}, f(x) = <f(\cdot), k(\cdot, x)>$$
Let us define $\mathcal{H}_0 = \text{span}\{\sum_{i \in I} \alpha_i k(\cdot, x_i), x_i \in \mathcal{X}, ||l|| < \infty\}$. $\mathcal{H}_0$ is the set of finite linear combinations of functions $x \rightarrow k(\cdot, x_i)$.

Introduce the operation $<\cdot, \cdot>_{\mathcal{H}_0}$:

$$\forall f, g, \in \mathcal{H}_0^2, f(\cdot) = \sum_{i \in I} \alpha_i k(\cdot, x_i)$$

$$g(\cdot) = \sum_{j \in J} \beta_j k(\cdot, z_j)$$

by

$$<f, g>_{\mathcal{H}_0} = \sum_{i \in I, j \in J} \alpha_i \beta_j k(x_i, z_j)$$

We notice that:

$$<f, g> = \sum_{j \in J} \beta_j f(z_j) = \sum_{i \in I} \alpha_i g(x_i)$$

meaning that this product between $f$ and $g$ does not depend on the expansions of $f$ or $g$. This last equation also shows that this product is bilinear. It is also trivially symmetric. $<\cdot, \cdot>_{\mathcal{H}_0}$ is a dot product on functions of $\mathcal{H}_0$. 
We define a norm from this dot product:

\[ \|f\|^2 = \langle f, f \rangle = \sum_{ij} \alpha_i \alpha_j k(x_i, x_j) = \alpha^T K \alpha \]

where \( K \) is the Gram matrix associated to \( k \).

Remark: we have a Cauchy-Schwartz inequality for PDS kernels (that we will use).

**Proposition:** Cauchy-Schwartz inequality

Let \( k \) be a PDS kernel then \( \forall (x, z) \in \mathcal{X}^2 \), we have:

\[ k(x, z)^2 \leq k(x, x) k(z, z) \]
We need to prove that we have the reproducing property:

\[
<f, k(\cdot, x)>_{\mathcal{H}_0} = <\sum_i \alpha_i k(\cdot, x_i), k(\cdot, x_i)>
\]

\[
= \sum_i \alpha_i k(x, x_i)
\]

\[
= f(x)
\]

Now $\mathcal{H}_0$ is named a pre-Hilbert space and we need to complete it with the limits of Cauchy sequences to get a **Hilbert space**.

Let $(f_n)_n$, a Cauchy sequence of functions of $\mathcal{H}_0$.

\[
\forall \epsilon > 0, \exists N \in \mathbb{N}, \forall p, q > N, \|f_p - f_q\|^2 < \epsilon
\]

Let us consider $\mathcal{H} = \mathcal{H}_0 \cup \{\text{lim of Cauchy sequences from } \mathcal{H}_0\}$.

Let us call $f = \lim_{n \to \infty} f_n$.

To ensure the reproducing property for these new functions, we need to have the pointwise convergence of $(f_n(x))_n$ for $x \in \mathcal{X}$. 
Proof of pointwise convergence of \((f_n(x))_n\) for \(x \in \mathcal{X}\)

\[\forall x \in \mathcal{X}, \forall (p, q) \in \mathbb{N}^2,\]

\[|f_p(x) - f_q(x)| = |< f_p, k(\cdot, x) - f_q, k(\cdot, x)>|\]
\[= |< f_p - f_q, k(\cdot, x)>|\]
\[\leq \sqrt{< f_p - f_q, f_p - f_q> \sqrt{k(x, x)}}\]
\[\leq \|f_p - f_q\| \sqrt{k(x, x)}\]

Then it comes that \((f_n(x))_n\) is a Cauchy Sequence in \(\mathbb{R}\) and thus has a limit.

Now \(f(x) = \lim_{n \to \infty} f_n(x)\).

We want to compute \(< \lim f_n, k(\cdot, x)>\). Let us first compute:

\[\lim_{n \to \infty} < f_n, k(\cdot, x)> = \lim f_n(x) = f(x).\]

We now define the dot product between a limit of Cauchy Sequence and the function \(k(\cdot, x)\) from \(\mathcal{H}_0\) as:\n\(< \lim f_n, k(\cdot, x)> := \lim f_n(x) = f(x)\). The dot product can be also defined between two limits of Cauchy sequences and also benefit from the reproducing property.
Theorem
Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a positive definite symmetric kernel and $\mathcal{H}_k$ be a Hilbert space built from $k$ and $\mathcal{X}$, then $\mathcal{H}_k$ is unique.
Any Hilbert space $\mathcal{H}$ such that there exists $\phi : \mathcal{X} \rightarrow \mathcal{H}$ with:

$$\forall (x, x') \in \mathcal{X} \times \mathcal{X}, \ k(x, x') = <\phi(x), \phi(x')>_{\mathcal{H}}$$

is called a feature space associated with $k$ and $\phi$ is called a feature map.
Representer theorem

Theorem
Let $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ be a positive definite symmetric kernel and $\mathcal{H}_k$, its corresponding RKHS, then, for any non-decreasing function $\Omega : \mathbb{R} \rightarrow \mathbb{R}$ and any loss function $L : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$, any minimizer of:

$$J(f) = L(f(x_1), \ldots, f(x_n)) + \lambda \Omega(\|f\|_H^2) \quad (2)$$

admits an expansion of the form:

$$f^*(\cdot) = \sum_{i=1}^{n} \alpha_i k(x_i, \cdot).$$

Moreover if $\Omega$ is strictly increasing, then any minimizer of 2 has exactly this form.
Proof of the Representer theorem

Let us define: $\mathcal{H}_1 = \text{span} \{k(x_i, \cdot), i = 1, \ldots, n\}$

Any $f \in \mathcal{H}$ writes as: $f = f_1 + f_{\perp}$, with $f_1 \in \mathcal{H}_1$ and $f_{\perp} \in \mathcal{H}_{1}^\perp$

where $\mathcal{H} = \text{direct sum of } \mathcal{H}_1$ and $\mathcal{H}_{1}^\perp$.

By orthogonality, $\|f\|^2 = \|f_1\|^2 + \|f_{\perp}\|^2$

Hence, by property of $\Omega$, $\Omega(\|f\|^2) = \Omega(\|f_1\|^2) + \Omega(\|f_{\perp}\|^2) \geq \Omega(\|f_1\|^2)$

By the reproducing property, we get:

$f(x_i) = \langle f_1(\cdot) + f_{1}^{\perp}(\cdot), k(x_i, \cdot) \rangle = \langle f_1(\cdot), k(x_i, \cdot) \rangle = f_1(x_i)$

Hence, $L(f(x_1), \ldots, f(x_n)) = L(f_1(x_1), \ldots, f_1(x_n))$ and $J(f_1) \leq J(f)$

To recap, if $f$ is a minimizer of $J(f)$, then $f_1$ is also a minimizer of $J$.

Moreover if $\Omega$ is strictly increasing, $J(f_1) < J(f)$, then any $f = f_1 + f_{1}^{\perp}$

exactly equals to $f_1$. 


1. Define a PDS kernel: $k(\cdot, \cdot)$
2. Define a RKHS, $\mathcal{H}$ from $k$ with an appropriate norm $\| \cdot \|_{\mathcal{H}}$
3. Define a loss functional with two terms: a local loss function $\ell$ and a penalty function $\Omega$
4. Prove/use a representer theorem to get the form of the minimizer of this functional: $\sum_i \alpha_i k(\cdot, x_i)$
5. Solve the optimization problem with this minimizer
Outline

Motivation

Multi-task Regression

Operator-valued kernel-based Regression

Reminder: scalar-valued case

Operator-valued kernels

Multi-task regression

Other applications of operator-valued kernels

Input Output Kernel Regression

recap

References
Outline

Motivation

Multi-task Regression

Operator-valued kernel-based Regression
  Reminder: scalar-valued case
  Operator-valued kernels
  Multi-task regression
  Other applications of operator-valued kernels

Input Output Kernel Regression

recap

References
We now consider the following case:

- We search a function $h: \mathcal{X} \rightarrow \mathcal{F}_y$
- $\mathcal{F}_y$ is an Hilbert space (NB: can be $\mathbb{R}^p$): this theory does not need to assume a kernel $k_y$ (it is more general)
- We will define functions of the following form:

$$h(x) = \sum_i K(x, x_i)c_i$$
Development of new learning tasks:

• Multi-task learning [Micchelli & Pontil, 2005, Evgeniou et al., 2005, Caponnetto et al. 2008]
• Functional regression [Kadri et al., 2010]
• Structured output prediction, link prediction [Brouard et al., Dinuzzo et al. 2011]
• Semi-supervised learning [Brouard et al. 2011, Quang 2011, 2014]
Definition of an operator-valued kernel

Let \( \mathcal{X} \) be a non-empty set and \( \mathcal{F}_y \), a Hilbert space \( \mathcal{F}_y \); \( \mathcal{L}(\mathcal{F}_y) \) is the set of all bounded linear operators from \( \mathcal{F}_y \) to itself.

**Operator-valued kernel:**

(Senkene & Tempel’man, 1973 ; Caponnetto et al., 2008)

\( K : \mathcal{X} \times \mathcal{X} \rightarrow \mathcal{L}(\mathcal{F}_y) \) is an operator-valued kernel if:

- \( \forall (x, x') \in \mathcal{X} \times \mathcal{X}, K(x, x') = K(x', x)^* \)
- \( \forall m \in \mathbb{N}, \) such that \( \forall i \in \{1, \ldots, m\}, (x_i, y_i) \in \mathcal{X} \times \mathcal{F}_y, \)

\[
\sum_{i,j=1}^{m} \langle K(x_i, x_j)y_j, y_i \rangle_{\mathcal{F}_y} \geq 0.
\]
Examples of operator-valued kernel \( \mathcal{F}_y = \mathbb{R}^p \)

Refs: Caponnetto et al. 2008; Carmeli et al. 2012; Lim et al. 2013

Trivial kernel (diagonal kernels):

\[
K(x, z) = k(x, z)l
\]

with \( l \): identity matrix, and \( k \) a scalar kernel.

Let \( k_1, \ldots, k_p \) be \( p \) scalar kernels

\[
K(x, x') = \begin{pmatrix}
  k_1(x, x') & 0 & \cdots & 0 \\
  0 & k_2(x, x') & \cdots & 0 \\
  0 & 0 & \cdots & k_p(x, x') \\
\end{pmatrix}
\]
Examples of operator-valued kernel

Other examples

• Decomposable kernel:
  • $\forall (x, z) \in \mathbb{R}^p \times \mathbb{R}^p, K(x, z) = k(x, z)B$ and $B \in S_p^+$
  • This kernel is used for multi-task regression when the matrix coefficient $B_{ij}$ codes for the dependency relationship between task $i$ and task $j$

• Transformable kernel
  • $\forall (i, j) \in \{1, \ldots, d\}^2, K(x, z)_{ij} = k(T_i(x), T_j(z))$
  • where $T_i$ is a transformation from $\mathcal{X}$ to some set $\mathcal{Z}$, and $k: \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$ is a scalar kernel

• Hadamard product of two kernels:
  • $K(x, z) = K_1(x, z) \circ K_2(x, z)$
Building a RKHS from an operator-valued kernel

Theorem (Senkene & Tempel’man, 1973 ; Micchelli & Pontil, 2005)

Let $\mathcal{X}$ be a set and $\mathcal{Y}$ be an Hilbert space.
If $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathcal{L}(\mathcal{F}_y)$ is an operator-valued kernel, then there exists a unique RKHS $\mathcal{H}_K$ which admits $K$ as the reproducing kernel, that is

$$\forall x \in \mathcal{X}, \forall y \in \mathcal{F}_y, \langle K(\cdot, x)y, h \rangle_{\mathcal{H}} = \langle y, h(x) \rangle_{\mathcal{F}_y}. \quad (3)$$

• $\mathcal{H}_K$ is built from functions of the form: $f(\cdot) = \sum_i K(\cdot, x_i)a_i$, 
  defining a inner product of functions $f$ and $g(\cdot) = \sum_j K(\cdot, z_j)b_j$ as $\langle f, g \rangle_{\mathcal{H}} = \sum_{i,j} \langle a_i, K(x_i, z_j)b_j \rangle_{\mathcal{F}_y}$ and completing this space by limits of Cauchy sequences.

• For sake of simplicity we omit $K$ and use $\mathcal{H} = \mathcal{H}_K$
Theorem (Micchelli & Pontil, 2005)

Let $\mathcal{X}$ be a set and $\mathcal{Y}$ be an Hilbert space. Given the RKHS $\mathcal{H}$ with reproducing kernel $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{F}_y)$, a set of labeled examples $S_n = \{(x_i, y_i)\}_{i=1}^n \subseteq \mathcal{X} \times \mathcal{F}_y$, a positive $\lambda \in \mathbb{R}^+$, let

$$J(h) = \sum_{i=1}^n \|y_i - h(x_i)\|_{\mathcal{F}_y}^2 + \lambda\|h\|_{\mathcal{H}}^2$$

be the functional loss, then the minimizer $\hat{h}$ of $J(h)$

$$\arg \min_{h \in \mathcal{H}} J(h) = \sum_{i=1}^n \|y_i - h(x_i)\|_{\mathcal{F}_y}^2 + \lambda\|h\|_{\mathcal{H}}^2,$$  \hspace{1cm} (4)

is unique and admits an expansion:

$$\hat{h}(\cdot) = \sum_{i=1}^n K(\cdot, x_i)c_i,$$
As the loss function $J(h)$ is strictly convex, there exists a unique minimizer.

**Closed-form solution**

The minimizer of $J(h) = \sum_{i=1}^{n} \|y_i - h(x_i)\|_F^2 + \lambda\|h\|_{\mathcal{H}}^2$, with $h : h(\cdot) = \sum_{i=1}^{n} K(\cdot, x_i)c_i$, has the following form:

$$\hat{h}(\cdot) = \phi_x(\cdot)(K_x + \lambda I_n)^{-1}Y_n$$

$K_x$ is the $n \times n$ block matrix, with each block of the form $K(x_i, x_j)$. $Y_n$ is the vector of all stacked vectors $y_1, \ldots, y_n$, and $\phi_x$ is the matrix composed of $[K(\cdot, x_1) \ldots K(\cdot, x_n)]$. 
Outline

Motivation

Multi-task Regression

Operator-valued kernel-based Regression
  Reminder: scalar-valued case
  Operator-valued kernels
  Multi-task regression
  Other applications of operator-valued kernels

Input Output Kernel Regression

recap

References
Multi-task regression: first formulation using Operator-Valued Kernels

**Mixed Effect Regularizer**

We set:

\[
A_\omega = \frac{1}{2(1-\omega)(1-\omega + \omega D)} \\
C_\omega = (2 - 2\omega + \omega D)
\]

Regularization:

\[
\Omega(h) = A_\omega \left( C_\omega \sum_{t=1}^{T} \|h_t\|_K^2 + \omega D \sum_{t} \|h_t - \bar{h}\|^2 \right)
\]

Choose a kernel \(K_\omega(x, x) = k(x, x)(\omega K + (1 - \omega)I_T)\)
Let $k_1, \ldots, k_p$ be $p$ scalar-valued kernels, 
$\mathcal{H}_1, \ldots, \mathcal{H}_p$ be the $p$ RKHS defined from these kernels. 
Let $M$ be a $p \times p$ positive weight matrix encoding the similarity between regression tasks.

Let us define the following regularizer:

$$\Omega(f) = \frac{1}{2} \sum_{ij} \|f_i - f_j\|^2_k m_{ij} + \sum_{i=1}^p \|f_i\|^2_k m_{ii}$$

This regularizer enforces the similarity between component functions that are close according matrix $M$. 
Multi-task regression: a second formulation

\[
\Omega(f) = \frac{1}{2} \sum_{ij} \|f_i - f_j\|^2_{m_{ij}} + \sum_{i=1}^{p} \|f_i\|^2_{m_{ii}}
\]

This regularizer enforces the similarity between component functions that are close according matrix \(M\).

\[
\Omega(f) = \sum_{i,j} \|f_i\|^2_{m_{ij}} - \langle f_i, f_j \rangle_{m_{ij}} + \sum_{i=1}^{p} \|f_i\|^2_{m_{ii}}
\]

\[
= \sum_{i=1}^{p} \|f_i\|^2_{m_{ii}} \sum_{j}(1 + \delta_{ij})m_{ij} - \sum_{ij} \langle f_i, f_j \rangle_{m_{ij}}
\]

\[
= \sum_{ij} \langle f_i, f_j \rangle_{L_{ij}}
\]

with \(L = D - M\), \(D = (d_{ij})_{ij}\), with the convention: \(d_{ij} = \delta_{ij}(\sum_k m_{ik} + m_{ij})\)
Let us define a decomposable kernel:

\[ K(x, x') = Bk(x, x'), \]

with \( B \) a sdp matrix and \( k \), a scalar-valued kernel. Then the \( \ell_2 \) norm in the RKHS \( \mathcal{H}_K \) of \( f(\cdot) = \sum_\ell K(\cdot, x_\ell)c_\ell \) can be written as:

\[ \|f\|_K^2 = \sum_{i,j=1}^{p} B_{ij}^+ \langle f_i, f_j \rangle_k, \]

where \( B^+ \) is the pseudo-inverse of \( B \) and \( f = (f_1, \ldots, f_p) \). Then the resulting kernel for Multi-task regression is the decomposable kernel: \( K(x, x') = L^+ k(x, x') \), with \( L^+ \), the pseudo inverse of \( L \).
If you want to solve a multi-task regression problem, you define the proper decomposable kernel to get a graph regularizer term
\[
\Omega(f) = \frac{1}{2} \sum_{i,j}^{p} ||f_i - f_j||_k^2 m_{ij} + \sum_{i=1}^{p} ||f_i||_k^2 m_{ii}
\]
**Working in RKHS of vector-valued functions**
As in the scalar-valued case, to define a regularizer of your choice, it is very often the case that you just need to define the proper kernel! You have now a new tool to take into account structure in the output space: define the proper operator-valued kernel and thus the proper norm \([f]_K^2\). NB: \([\cdot]_K := \| \cdot \|_{\mathcal{H}_K}.\)
Su et al. 2010 Dataset: contains the 2303 molecules that are all active against at least one cell line. Each molecule is represented by a graph, where nodes correspond to atoms and edges to bonds between atoms. The Tanimoto kernel (Ralaivola et al. 2005) is used for the scalar input kernel:

\[ K(x, x') = \frac{k_m(x, x')}{k_m(x, x) + k_m(x', x') - k_m(x, x')} \]

\( k_m \) is chosen as a linear path kernel. Input feature vectors \( \varphi_{x_m}(x) \) are binary vectors that indicate the presences and absences in the molecules of all existing paths containing a maximum of \( m \) bonds. \( m = 6 \).
Results on NCI-dataset: Drug activity prediction
Outline

Motivation

Multi-task Regression

Operator-valued kernel-based Regression

Reminder: scalar-valued case

Operator-valued kernels

Multi-task regression

Other applications of operator-valued kernels

Input Output Kernel Regression

recap

References
Other applications

- Vector Field learning
- Mismatch Removal
- Image Colorization
- Structured Classification
Mismatch Vector Field Learning

Establishing correspondences between two images of the same scene

Ref: Ma et al. Pattern Recognition, 2013.

Fig. 1. Schematic illustration of motion field introduced by image pairs. Left: an image pair and its putative matches; right: motion field samples introduced by matches in the left figure. ◆ and × indicate feature points in the first and second images, respectively.
Model and observation assumptions

The temporal evolution of the system is ruled by a \textbf{first-order autoregressive} model \( h : \mathbb{R}^p \rightarrow \mathbb{R}^p \):

\[
x_{t+1} = h(x_t) + \epsilon_{t+1}
\]

where

\begin{itemize}
  \item \( x_1, \ldots, x_{N+1} \in \mathbb{R}^p \) : observed time series of a dynamical system comprising of \( p \) variables at time \( t = 1, \ldots, N + 1 \)
  \item \( \epsilon_t \) : a noise term (chosen Gaussian) with zero-mean
\end{itemize}
Network inference by thresholding the Jacobian

<table>
<thead>
<tr>
<th></th>
<th>Linear</th>
<th>Nonlinear</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Model</td>
<td>$x_{t+1} = Bx_t + \epsilon_{t+1}$</td>
<td></td>
</tr>
<tr>
<td>2. Learn</td>
<td>$B$</td>
<td></td>
</tr>
<tr>
<td>4. Adjacency matrix</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$\hat{A}_{ij} = 1?$
Network inference by thresholding the Jacobian

<table>
<thead>
<tr>
<th>Linear</th>
<th>Nonlinear</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Model</td>
<td>$x_{t+1} = Bx_t + \epsilon_{t+1}$</td>
</tr>
<tr>
<td>2. Learn</td>
<td>$B$</td>
</tr>
<tr>
<td>4. Adjacency matrix</td>
<td>Threshold $\hat{B}$</td>
</tr>
</tbody>
</table>

$|\hat{B}_{ij}| \geq \epsilon$?
Network inference by thresholding the Jacobian

<table>
<thead>
<tr>
<th></th>
<th>Linear</th>
<th>Nonlinear</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Model</td>
<td>$x_{t+1} = Bx_t + \epsilon_{t+1}$</td>
<td>$h : \mathbb{R}^p \rightarrow \mathbb{R}^p$</td>
</tr>
<tr>
<td></td>
<td>$x_{t+1} = h(x_t) + \epsilon_{t+1}$</td>
<td></td>
</tr>
<tr>
<td>2. Learn</td>
<td>$B$</td>
<td>$h$</td>
</tr>
<tr>
<td>4. Adjacency matrix</td>
<td>Threshold $\hat{B}$</td>
<td></td>
</tr>
</tbody>
</table>

![Diagram](image_url)
Network inference by thresholding the Jacobian

<table>
<thead>
<tr>
<th>1. Model</th>
<th>Linear</th>
<th>Nonlinear</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_{t+1} = Bx_t + \epsilon_{t+1}$</td>
<td>$h : \mathbb{R}^p \rightarrow \mathbb{R}^p$</td>
<td>$x_{t+1} = h(x_t) + \epsilon_{t+1}$</td>
</tr>
<tr>
<td>2. Learn</td>
<td>$B$</td>
<td>$h$</td>
</tr>
<tr>
<td>$\hat{B}_{ij} = \frac{\partial (\hat{B}x_t)^i}{\partial x_t^j}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. Adjacency matrix</td>
<td>Threshold $\hat{B}$</td>
<td></td>
</tr>
</tbody>
</table>

$|\hat{B}_{ij}| \geq \epsilon$?
Network inference by thresholding the Jacobian

<table>
<thead>
<tr>
<th></th>
<th>Linear</th>
<th>Nonlinear</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Model</td>
<td>$x_{t+1} = Bx_t + \epsilon_{t+1}$</td>
<td>$h : \mathbb{R}^p \to \mathbb{R}^p$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$x_{t+1} = h(x_t) + \epsilon_{t+1}$</td>
</tr>
<tr>
<td>2. Learn</td>
<td>$B$</td>
<td>$h$</td>
</tr>
<tr>
<td>3. Jacobian $J(\hat{h})_{ij}$</td>
<td>$\hat{B}_{ij} = \frac{\partial (\hat{B}x_t)^i}{\partial x_t^j}$</td>
<td>$T \left( \frac{\partial \hat{h}(x_1)^i}{\partial x_1^j}, \ldots, \frac{\partial \hat{h}(x_N)^i}{\partial x_N^j} \right)$</td>
</tr>
<tr>
<td>4. Adjacency matrix</td>
<td>Threshold $\hat{B}$</td>
<td>Threshold $J(\hat{h})$</td>
</tr>
</tbody>
</table>

Examples for $T(z_1, \ldots, z_N)$:

$\frac{1}{N} \sum_{t=1}^N z_t$, median$(z_1, \ldots, z_N)$, max$(|z_1|, \ldots, |z_N|)$, ...
Learning a sparse OVKR

**Case 1:** Kernel $K$ is given

Learning $h$ boils down to the following optimization problem:

$$\min_{\mathbf{C}} \mathcal{L}(\mathbf{C}) = \sum_{i=1}^{n} ||\mathbf{y}_i - h_{\mathbf{C}}(\mathbf{x}_i)||^2_2$$

(6)
Case 1: Kernel $K$ is given

Learning $h$ boils down to the following optimization problem:

$$
\underset{C}{\text{minimize}} \quad \mathcal{L}(C) = \sum_{i=1}^{n} \|y_i - h_C(x_i)\|_2^2 + \lambda_h \|h_C\|_{\mathcal{H}_K}^2
$$

(6)
Learning a sparse OVKR

**Case 1:** Kernel $K$ is given

Learning $h$ boils down to the following optimization problem:

$$
\text{minimize } \mathcal{L}(C) = \sum_{i=1}^{n} ||y_i - h_C(x_i)||^2_2 + \lambda_h ||h_C||^2_{\mathcal{H}_K} + \Omega(C)
$$

- **Penalty on $c_\ell$'s:**
  - $\Omega_1(C) = \lambda_c ||C||_{\ell_1} = \lambda_c \sum_{\ell=1}^{n} \sum_{p=1}^{d} |c_p^\ell|$
  - $\Omega_{\text{struct}}(C) = \lambda_c ||C||_{\ell_1/\ell_2} = \lambda_c \sum_{\ell=1}^{n} ||c_\ell||_2$
Learning a sparse OVKR

Case 2: \( K(x, x') = B k(x, x') \) Parameter \( B \) of Kernel \( K \) has to be learned

Learning \( h \) boils down to the following optimization problem:

\[
\minimize_{B, C} \mathcal{L}(B, C) = \sum_{i=1}^{n} \| y_i - h_{B, C}(x_i) \|^2_2 + \lambda_h \| h_{B, C} \|^2_{H_K} + \Omega(C) + \Omega(B) \quad (6)
\]

- Penalty on \( c_\ell \)'s:
  - \( \Omega_1(C) = \lambda_c \| C \|_{\ell_1} = \lambda_c \sum_{\ell=1}^{n} \sum_{p=1}^{d} |c^p_\ell| \)
  - \( \Omega_{\text{struct}}(C) = \lambda_c \| C \|_{\ell_1/\ell_2} = \lambda_c \sum_{\ell=1}^{n} \| c_\ell \|_2 \)

- Penalty on \( B \):
  - \( \Omega_1(B) = \lambda_B \| B \|_{\ell_1} + 1_{S_+^d}(B) = \lambda_B \sum_{p, q=1}^{d} |B_{pq}| + 1_{S_+^d}(B) \)
Learning OVKR

• For fixed $\hat{B}$, the loss function to be minimized becomes:

$$L(\hat{B}, \hat{C}) = \sum_{i=1}^{n} \|y_i - h_{\hat{B},\hat{C}}(x_i)\|_2^2 + \lambda_h \|h_{\hat{B},\hat{C}}\|_{\mathcal{H}_K}^2 + \Omega(\hat{C}) \quad (7)$$

• For given $\hat{C}$, the loss function to be minimized is the following:

$$L(B, \hat{C}) = \sum_{i=1}^{n} \|y_i - h_{B,\hat{C}}(x_i)\|_2^2 + \lambda_h \|h_{B,\hat{C}}\|_{\mathcal{H}_K}^2 + \Omega(B) \quad (8)$$

We employ proximal gradient algorithms to minimize (7) and (8)

Ref: introduction to proximal algorithms, see Vandenberghe 'slides http://www.seas.ucla.edu/~vandenbe/236C/lectures/proxgrad.pdf

Ref: Lim et al. 2015.
• For fixed $\hat{B}$, the loss function to be minimized becomes:

$$
\mathcal{L}(\hat{B}, C) = \sum_{i=1}^{n} \|y_i - h_{\hat{B}, C}(x_i)\|_2^2 + \lambda_h \|h_{\hat{B}, C}\|_{\mathcal{H}_K}^2 + \Omega(C) \quad (7)
$$

• For given $\hat{C}$, the loss function to be minimized is the following:

$$
\mathcal{L}(B, \hat{C}) = \sum_{i=1}^{n} \|y_i - h_{B, \hat{C}}(x_i)\|_2^2 + \lambda_h \|h_{B, \hat{C}}\|_{\mathcal{H}_K}^2 + \Omega(B) \quad (8)
$$

We employ proximal gradient algorithms to minimize (7) and (8).

Ref: introduction to proximal algorithms, see Vandenberghe 'slides http://www.seas.ucla.edu/~vandenbe/236C/lectures/proxgrad.pdf + Ref: Lim et al. 2015.
What are proximal algorithms about?

• Class of optimization algorithms
• Tools for convex, **nonsmooth**, constrained and **large-scale** problems
• Recent interest in machine learning, image and signal processing communities
• Proximal operators are **not new** [Moreau, 1962]
Definition

Proximal operator

Let \( f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\} \) be a closed proper convex function, the proximal operator \( \text{prox}_{\lambda f} : \mathbb{R}^n \to \mathbb{R}^n \) of the scaled function \( \lambda f \), where \( \lambda > 0 \) is defined by

\[
\text{prox}_{\lambda f}(v) = \arg \min_x \left\{ f(x) + \frac{1}{2\lambda} \|x - v\|^2 \right\}
\]  

(9)
Proximal gradient algorithms

- Type 1 problem

\[
\text{minimize } f(x) + g(x) \tag{10}
\]

- \( f : \mathbb{R}^n \to \mathbb{R} \) and \( g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\} \) are closed proper convex functions
- \( f \) is differentiable and \( g \) possibly nonsmooth
Proximal gradient algorithms

- **Type 1 problem**

  \[
  \text{minimize } f(x) + g(x) \tag{10}
  \]

  - \( f : \mathbb{R}^n \rightarrow \mathbb{R} \) and \( g : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\} \) are closed proper convex functions
  - \( f \) is **differentiable** and \( g \) possibly **nonsmooth**
  - **Forward-Backward Splitting** [Bruck 1975], [Lions & Mercier, 1979], [Beck & Teboulle, 2009, 2010]

- **Type 2 problem**

  \[
  \text{minimize } f(x) + \sum_{i=1}^{N} g_i(x) \tag{11}
  \]

  - \( f : \mathbb{R}^n \rightarrow \mathbb{R} \) and \( g_i : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\} \) are closed proper convex functions for all \( i = 1 \ldots N \)
  - \( f \) is **differentiable** and \( g_i \)'s are possibly **nonsmooth**
  - **Generalized Forward-Backward Splitting** [Raguet et al., 2011]
Alternate scheme to learn $C$ and $B$

Inputs: $B_0 \in S_p^+; M; \epsilon_B; \epsilon_C$

Initialize: $m = 0; STOP = false$

while $m < M$ and STOP=false do
    Step 1: Given $B_m$, minimize the loss function (7) and obtain $C_m$
    Step 2: Given $C_m$, minimize the loss function (8) and obtain $B_{m+1}$
    if $m > 0$ then
        STOP:=||$B_m - B_{m-1}|| \leq \epsilon_B$ and $||C_m - C_{m-1}|| \leq \epsilon_C$
    end if
    Step 3: $m \leftarrow m + 1$
end while
• Dinuzzo et al. 2011, Sylvester equations for C and block-coordinate descent for B but only smooth penalties
• Ciliberto et al. 2015, variable change and recent algorithm (barrier method)
DREAM3 data set

• DREAM stands for Dialogue for Reverse Engineering Assessments and Methods

• Why DREAM3? Two data sets including time-series without perturbation data

• An example of gene regulatory network: *E. coli* subnetwork

  ![Gene Regulatory Network](image)

  → activation → inhibition
Simulated data to have ground truth to build AUROC and AUPR.

- 5 networks whose structure is taken from E. coli or S. Cerevisiae: simulation of Michaelis-Menten equations
- Datasets: 4 time series of 21 time-points (4 initial conditions)
- Comparison with the best team in DREAM3 challenge that only uses time-series and other methods
- Assessment of performance: area under the ROC curve (TP rate vs FP rate), area under the Precision Recall curve (Pr = TP/P, Rec = TP)
Comparison with state-of-the-art methods

<table>
<thead>
<tr>
<th></th>
<th>AUROC</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>OKVAR Prox* + True B</td>
<td>96.2</td>
<td>97.1</td>
<td>95.8</td>
<td>90.6</td>
<td>89.7</td>
</tr>
<tr>
<td>OKVAR Prox</td>
<td>81.5</td>
<td>78.7</td>
<td>76.5</td>
<td>70.3</td>
<td>75.1</td>
</tr>
<tr>
<td>LASSO</td>
<td>69.5</td>
<td>57.2</td>
<td>46.6</td>
<td>62.0</td>
<td>54.5</td>
</tr>
<tr>
<td>GPODE</td>
<td>60.7</td>
<td>51.6</td>
<td>49.4</td>
<td>61.3</td>
<td>57.1</td>
</tr>
<tr>
<td>G1DBN</td>
<td>63.4</td>
<td>77.4</td>
<td>60.9</td>
<td>50.3</td>
<td>62.4</td>
</tr>
<tr>
<td>Team 236</td>
<td>62.1</td>
<td>65.0</td>
<td>64.6</td>
<td>43.8</td>
<td>48.8</td>
</tr>
<tr>
<td>Team 190</td>
<td>57.3</td>
<td>51.5</td>
<td>63.1</td>
<td>57.7</td>
<td>60.3</td>
</tr>
</tbody>
</table>

## Comparison with state-of-the-art methods 2

<table>
<thead>
<tr>
<th>Method</th>
<th>AUPR E1</th>
<th>AUPR E2</th>
<th>AUPR Y1</th>
<th>AUPR Y2</th>
<th>AUPR Y3</th>
</tr>
</thead>
<tbody>
<tr>
<td>OKVAR Prox* + True B</td>
<td>43.2</td>
<td>51.6</td>
<td>27.9</td>
<td>40.7</td>
<td>36.4</td>
</tr>
<tr>
<td>OKVAR Prox</td>
<td>32.1</td>
<td>50.1</td>
<td>35.4</td>
<td>37.4</td>
<td>39.7</td>
</tr>
<tr>
<td>LASSO</td>
<td>17.0</td>
<td>16.9</td>
<td>8.5</td>
<td>32.9</td>
<td>23.2</td>
</tr>
<tr>
<td>GPODE</td>
<td>18.0</td>
<td>14.6</td>
<td>8.9</td>
<td>37.7</td>
<td>34.1</td>
</tr>
<tr>
<td>G1DBN</td>
<td>16.5</td>
<td>36.4</td>
<td>11.6</td>
<td>23.2</td>
<td>26.3</td>
</tr>
<tr>
<td>Team 236</td>
<td>19.7</td>
<td>37.8</td>
<td>19.4</td>
<td>23.6</td>
<td>23.9</td>
</tr>
<tr>
<td>Team 190</td>
<td>15.2</td>
<td>18.1</td>
<td>16.7</td>
<td>37.1</td>
<td>37.3</td>
</tr>
</tbody>
</table>
When no ground truth is provided: climate data

- 125 equally spaced meteorological stations in USA
- Up to 12 climate variables
- Model selection using one station with BIC, learning on the others

We used OKVAR to infer a consensus network from multiple runs of OKVAR on climate time series of each of the remaining stations. We clustered the inferred networks and visualized the clustering on the US map.
Outline

Motivation

Multi-task Regression

Operator-valued kernel-based Regression

Input Output Kernel Regression

recap

References
Learning with complex outputs with output kernel

1. Define $k_y$ for the output space, $\phi_y$ is a feature map associated to $k$ and $\mathcal{F}_y$ the feature space corresponding to $\phi_y$

2. learn $h : \mathcal{X} \rightarrow \mathcal{F}_y$ from training data $S$

3. Once $h$ is learned, to make a prediction : solve a pre-image problem
To solve the link prediction task:

1. We learn $h_{\text{tree}} : \mathcal{X} \rightarrow \mathcal{F}_y$

2. For prediction:
   - we compute for a new pair:
     
     $$f(x, x') = \text{sign}(< h_{\text{tree}}(x), h_{\text{tree}}(x') >_{\mathcal{F}_y} - \theta)$$
Limitations of tree-based methods

- Not appropriate when huge input dimension (although Random Forest is a possible key)
- Not appropriate for structured data
- Not appropriate for semi-supervised learning

Idea: learn with functions in vector-valued RKHS with kernelized outputs
Motivation: deal with structured outputs as well as structured inputs

1. Define $k_y$ for the output space, $\phi_y$ is a feature map associated to $k$ and $\mathcal{F}_y$ the feature space corresponding to $\phi_y$
2. NB : special case : $\mathcal{F}_y = \mathbb{R}^p$
3. Define $K_x : \mathcal{X} \times \mathcal{X} \rightarrow \mathcal{L}(\mathcal{F}_y)$ for the input space ($\phi_x$: feature map associated with $K$)
4. Learn $h : \mathcal{X} \rightarrow \mathcal{F}_y$ from training data $S$
5. Once $h$ is learned, to make a prediction : solve a pre-image problem
Link prediction with output kernel

In this task, we assume that for each object $u \in \mathcal{U}$

- $\phi_y(u) \in \mathcal{F}_y$, using a feature map $\phi_y(\cdot)$ and a feature space $\mathcal{G}_y$ associated to a given kernel $k_y : \mathcal{U} \times \mathcal{U} \to \mathbb{R}$
- $k_y$ codes for the proximity of two nodes in the graph

Training data:
Description of the input data by $K_x = (K_{ij})_{ij}$ the block matrix composed of matrices $K(u_i, u_j)$ and $K_y$
Input and output kernel regression for link prediction

We define $f$ as follows:

$$f(u, u') = ((h(u), h(u')) - t),$$

where $t$ is a threshold and $h: U \to \mathcal{F}_y$ approximates the relationship between $u$ and $\phi_y(u)$.

We define $K: U \times U \to \mathcal{L}(\mathcal{F}_y)$, $K$ has values in the set of linear bounded operators on $\mathcal{F}_y$.

$$h(u) = \sum_{\ell} K(u, u') c_\ell \in \mathcal{F}_y$$
Similarly to output kernel tree (OK3), we have: \( \forall (u, u'), \langle h(u), h(u') \rangle \mathcal{F}_y \) is an approximation of \( k_y(u, u') \)

How to choose \( k_y \) for link prediction?

- Use the \textbf{diffusion kernel} [Kondor & Lafferty, 2002] :
  \[
  K_y = \exp(-\beta L),
  \]

- where the graph Laplacian \( L \) is defined by \( L = D - A \)
- with \( D \) the degree matrix and \( A \) the adjacency matrix of the training graph

Training data:
Description of the input data by \( K_x = (K_{ij})_{ij} \) the block matrix composed of matrices \( K(u_i, u_j) \) and \( K_y \)
The link prediction problem can be solved by minimizing:

\[
L(f) = \sum_{\ell} ||\phi_y(u_\ell) - f(u_\ell)||^2_K + \lambda_f ||f||^2_K + \lambda_s \sum_{\ell,m} w_{\ell m} ||f(u_\ell) - f(u_m)||^2,
\]

with $W$ a matrix such that $w_{\ell m}$ encodes proximity between $u_\ell$ and $u_m$. NB: as for kernel ridge regression, there is a closed form solution. (Brouard et al. 2011)
### a) AUC-ROC:

<table>
<thead>
<tr>
<th>Method</th>
<th>GO-BP</th>
<th>GO-CC</th>
<th>GO-MF</th>
<th>int</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naive</td>
<td>60.8 ± 0.8</td>
<td>64.4 ± 2.5</td>
<td>64.2 ± 0.8</td>
<td>67.7 ± 1.5</td>
</tr>
<tr>
<td>kCCA</td>
<td>82.4 ± 3.6</td>
<td>77.0 ± 1.7</td>
<td>75.0 ± 0.6</td>
<td>85.7 ± 1.6</td>
</tr>
<tr>
<td>kML</td>
<td>83.2 ± 2.4</td>
<td>77.8 ± 1.1</td>
<td>76.6 ± 1.9</td>
<td>84.5 ± 1.5</td>
</tr>
<tr>
<td>Local</td>
<td>79.5 ± 1.6</td>
<td>73.1 ± 1.3</td>
<td>66.8 ± 1.2</td>
<td>83.0 ± 0.5</td>
</tr>
<tr>
<td>OK3+ET</td>
<td>84.3 ± 2.4</td>
<td>81.5 ± 1.6</td>
<td>79.3 ± 1.8</td>
<td>86.9 ± 1.6</td>
</tr>
<tr>
<td>IOKR-ridge</td>
<td>88.8 ± 1.9</td>
<td>87.1 ± 1.3</td>
<td>84.0 ± 0.6</td>
<td>91.2 ± 1.2</td>
</tr>
</tbody>
</table>

### b) AUC-PR:

<table>
<thead>
<tr>
<th>Method</th>
<th>GO-BP</th>
<th>GO-CC</th>
<th>GO-MF</th>
<th>int</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naive</td>
<td>4.8 ± 1.0</td>
<td>2.1 ± 0.6</td>
<td>2.4 ± 0.4</td>
<td>8.0 ± 1.7</td>
</tr>
<tr>
<td>kCCA</td>
<td>7.1 ± 1.5</td>
<td>7.7 ± 1.4</td>
<td>4.2 ± 0.5</td>
<td>9.9 ± 0.4</td>
</tr>
<tr>
<td>kML</td>
<td>7.1 ± 1.3</td>
<td>3.1 ± 0.6</td>
<td>3.5 ± 0.4</td>
<td>7.8 ± 1.6</td>
</tr>
<tr>
<td>Local</td>
<td>6.0 ± 1.1</td>
<td>1.1 ± 0.3</td>
<td>0.7 ± 0.0</td>
<td>22.6 ± 6.6</td>
</tr>
<tr>
<td>OK3+ET</td>
<td>19.0 ± 1.8</td>
<td>21.8 ± 2.5</td>
<td>10.5 ± 2.0</td>
<td>26.8 ± 2.4</td>
</tr>
<tr>
<td>IOKR-ridge</td>
<td>15.3 ± 1.2</td>
<td>20.9 ± 2.1</td>
<td>8.6 ± 0.3</td>
<td>22.2 ± 1.6</td>
</tr>
</tbody>
</table>

Table 4: AUC-ROC and AUC-PR estimated by 5-CV for the yeast PPI network reconstruction in the supervised setting with different input kernels (GO-BP: GO biological processes; GO-CC: GO cellular components; GO-MF: GO molecular functions; int : average of the different kernels).

- Results of Céline Brouard: various input kernel, output kernel: diffusion kernel.
<table>
<thead>
<tr>
<th></th>
<th>AUC-ROC</th>
<th>AUC-PR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5%</td>
<td>10%</td>
</tr>
<tr>
<td>Transductive setting</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EM</td>
<td>87.3 ± 2.4</td>
<td>92.9 ± 1.7</td>
</tr>
<tr>
<td>PKMR</td>
<td>85.7 ± 4.1</td>
<td>92.4 ± 1.6</td>
</tr>
<tr>
<td>IOKR</td>
<td>83.6 ± 5.9</td>
<td>93.6 ± 1.0</td>
</tr>
<tr>
<td>Semi-supervised setting</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IOKR</td>
<td>86.0 ± 2.7</td>
<td>93.3 ± 0.7</td>
</tr>
</tbody>
</table>

Table 3: AUC-ROC and AUC-PR obtained for the NIPS co-authorship network inference with EM, PKMR, IOKR in the transductive setting, and with IOKR in the semi-supervised setting. $p$ indicates the percentage of labeled examples.

- Results of Céline Brouard, JMLR 2016.
Motivation

Multi-task Regression

Operator-valued kernel-based Regression

Input Output Kernel Regression

recap

References
Part II : structured output prediction

- Large Margin approaches, Logistic Regression model (CRF), Joint Kernel Map
- Output Kernel Regression, Operator-valued Kernel Regression, Input Output Kernel Regression
- Open challenges : large scale approaches
• Evgeniou, Micchelli, Pontil, Regularizing multi-tasks, 2005.
• Francesco Dinuzzo, Cheng Soon Ong, Peter V. Gehler, Gianluigi Pillonetto: Learning Output Kernels with Block Coordinate Descent. ICML 2011: 49-56 (Eligible for the project)
• Micchelli and Pontil, Learning vector-valued functions, JMLR 2005.(seminal paper)
• C. Brouard, F. d’Alché-Buc, M. Szafranski, Semi-supervised link prediction with penalized output kernel regression, ICML 2011
• J. Ma, J. Zhao, J. Tuian, X. Bai, Z. Tu. Regularized Vector Field with sparse approximation, Pattern Recognition (2013) - (Eligible for the project)