Learning Theory and Applications in Model Reduction

at the block seminar

"Angewandte Mathematik"

University of Münster / Kleinwalsertal

Daniel Wirtz, 22.02.2011

Dipl. Math. Dipl. Inf. Daniel Wirtz
Jun.-Prof. Dr. Bernard Haasdonk
Institute of Applied Analysis and Numerical Simulation
University of Stuttgart
Pfaffenwaldring 57, D-70569 Stuttgart
daniel.wirtz@mathematik.uni-stuttgart.de

Contents

Overview

Part I

Kernels and Learning Theory

Introductory examples

ASCII

ASCII Letter recognition

- Assume given images u_i of say 256×256 pixels
- true corresponding ASCII letter a enumerated by $1 \dots 95$ (printable letters)

Figure: Some handwritten ASCII letters and their recognized counterparts

Introductory examples

More..

Math viewpoint

- Inputs $u_i \in \mathbb{R}^{65536}$ and outputs $a_i \in \{1 \ldots 95\}$
- Task: Automatically figure out which is the correct letter for a *new* u.

Introductory examples

More..

Math viewpoint

- Inputs $u_i \in \mathbb{R}^{65536}$ and outputs $a_i \in \{1 \dots 95\}$
- Task: Automatically figure out which is the correct letter for a *new* u.

More scenarios

- Have geographical position, ground size, infrastructure flags and water quality as input, resulting house prices for those areas
- Customer information and the associated probability of payback of loans they ackquired
- Aim: Predict the prices for houses / chances of loan payback given a new customer.

The general setting

Abstract viewpoint

- Input data from some set $X \subset \mathbb{R}^d$
- Output data from some set $Y \subset \mathbb{R}$
- Given data samples

$$D = \{(x_i, y_i) \mid x_i \in X, y_i \in Y, i = 1 \dots N\}$$

- Task: For given D find

$$f : X \to Y$$

  that represents the connection betweeen inputs and outputs and ideally generalizes D.

Which class of functions?

Some function classes

Ideas

- $f \in C^0(X)$ reasonable
- Maybe linear functions: Linear Regression
- Polynomials: Standard Interpolation
- Kernel induced functions! Coming next.

Kernel Definition

Given some input space $\Omega \subset \mathbb{R}^d$ a *symmetric positive definite kernel* (s.p.d.) Φ is a mapping

$$\Phi : \Omega \times \Omega \longrightarrow \mathbb{R}$$

that satifies

$$\Phi(x,y) = \Phi(y,x) \quad \forall\ x,y \in \Omega \tag{1}$$

$$\sum_{i,j}^{N} \alpha_i \alpha_j \Phi(x_i, x_j) \geq 0 \quad \forall\ \alpha \in \mathbb{R}^N, x_i \in \Omega, i = 1 \ldots ..N\ \forall\ N \in \mathbb{N} \tag{2}$$

Kernels
Examples

Some kernel examples

Assume $x, y \in \mathbb{R}^d$.

1. Linear kernel:
$$\Phi(x, y) := \langle x, y \rangle$$
2. Polynomial kernel:
$$\Phi(x, y) := (\langle x, y \rangle + 1)^p$$
for $p \in \mathbb{N}$.
3. Gaussian kernel:
$$\Phi(x, y) := e^{-\frac{||x-y||^2}{\gamma}}$$
for $\gamma > 0$

Induced function space

For every s.p.d. kernel Φ consider the span of functions

$$F_\Phi := \langle \{\Phi(x,\cdot) \mid x \in \Omega\} \rangle = \left\{ \sum_{i=1}^{N} \alpha_i \Phi(x_i,\cdot) \;\middle|\; N \in \mathbb{N}, x_i \in \Omega, \alpha \in \mathbb{R}^N \right\}$$

Equip this space with a scalar product

$$\langle \Phi(x_i,\cdot), \Phi(x_j,\cdot) \rangle_{F_\Phi} := \Phi(x_i, x_j)$$

to get a pre-Hilbert space (sums canonically).

Kernels
Function Spaces

Induced function space

For every s.p.d. kernel Φ consider the span of functions

$$F_\Phi := \langle \{ \Phi(x,\cdot) \mid x \in \Omega \} \rangle = \left\{ \sum_{i=1}^{N} \alpha_i \Phi(x_i,\cdot) \;\middle|\; N \in \mathbb{N}, x_i \in \Omega, \alpha \in \mathbb{R}^N \right\}$$

Equip this space with a scalar product

$$\langle \Phi(x_i,\cdot), \Phi(x_j,\cdot) \rangle_{F_\Phi} := \Phi(x_i, x_j)$$

to get a pre-Hilbert space (sums canonically).

Completion: RKHS

The completion $\mathcal{N}_\Phi(\Omega) := \overline{F_\Phi}$ of F_Φ is a Hilbert space, referred to as

- *Native space* of Φ
- *Reproducing kernel Hilbert space (RKHS)*

Kernels

Approximation with Kernels

Situation

- Recall: Want to find representing/generalizing function $f : X \to Y$
- Recall: Given data samples $D = \{(x_i, y_i) \mid i = 1 \ldots N\}$
- Now choose $f \in F_\Phi$, i.e.

$$f(x) = \sum_{i=1}^{N} \alpha_i \Phi(x, x_i)$$

- Later $f \in \mathcal{N}_\Phi(\Omega)$, details follow
- New Task:
  - Choose/find suitable kernel Φ
  - Choose/compute suitable coefficient (vectors) $\boldsymbol{\alpha} \in \mathbb{R}^d$

Method 1: Kernel Interpolation

Theory

Quick 'n Dirty: Kernel Interpolation

- Have conditions

$$y_k = f(x_k) = \sum_{i=1}^{N} \alpha_i \Phi(x_k, x_i), \quad k = 1 \dots N,$$

- Reformulation leads to

$$\begin{pmatrix} \Phi(x_1, x_1) & \dots & \Phi(x_1, x_N) \\ \vdots & \ddots & \vdots \\ \Phi(x_N, x_1) & \dots & \Phi(x_N, x_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}$$

- Matrix form Defining $K_{i,j} := \Phi(x_i, x_j), Y = (y_i)_i$ we obtain

$$K\boldsymbol{\alpha} = Y.$$

Only uniquely solvable if K is invertible! (Details out of scope..)

Method 1: Kernel Interpolation

Example

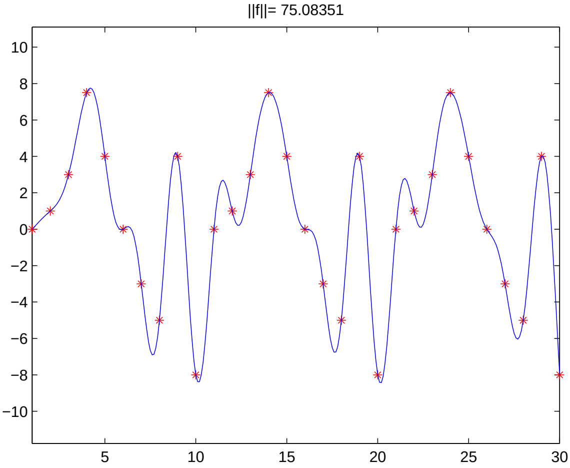

Figure: Kernel Interpolation example with Gaussian kernel and $\gamma = 2.5$

Learning Theory

Now lets consider a different way to select the α coefficients: statistical learning theory.

What is "learning"?

- In real life: Some previously unknown process can be performed better the next time.
- In math context: Figure out how the mapping f works, i.e. increase the quality of a response $f(x)$ to the true data (x, y).

Most generally: Everything is probability!

- Assume D to be generated by probability distribution $P : X \times Y \to \mathbb{R}^+$.
- Task: "Learn" distribution P only using D so that we can predict the future outputs once an input is given.
- Vital assumption: Same process generates future data!

Quality measures

How to measure the quality of a prediction?

Loss functions

Measure quality/discrepancy via *loss functions* (or cost functions)

$$L(x, y, f(x))$$

Examples:

1. Hinge-Loss:
$$L(x, y, f(x)) = \max\{0, 1 - yf(x)\}$$

2. Least-Squares:
$$L(x, y, f(x)) = (y - f(x))^2$$

Expected loss & learning goal

How does f "learn" the probability distribution?

Expected loss / risk

$$R_{L,P}(f) = \int\limits_{X \times Y} L(x, y, f(x)) dP(x, y)$$

for a given mapping $f : X \to Y$.

Define

$$\mathcal{H} := \{g \mid g : X \to Y\}$$

to be all possible mappings betweeen X and Y. Then we set the

Learning goal

Minimize the expected loss / risk:

$$f^* = \arg\min_{f \in \mathcal{H}} R_{L,P}(f)$$

Empirical Risk

We know the distribution at some points: D. This is the discrete version

$$P_D(x,y) = \frac{1}{N}\sum_{i=1}^{N}\delta_{(x_i,y_i)}(\cdot,\cdot)$$

of P. This gives the

Empirical risk

$$R_{L,P_D}(f) = \int_{X\times Y} L(x,y,f(x))dP_D(x,y) = \frac{1}{N}\sum_{i=1}^{N} L(x_i,y_i,f(x_i))$$

Empirical Risk Minimization

Consequently, we obtain the

Empirical Risk Minimization (ERM) problem

$$f_D = \arg\min_{f\in\mathcal{H}} R_{L,P_D}(f) = \arg\min_{f\in\mathcal{H}} \frac{1}{N}\sum_{i=1}^{N} L(x_i, y_i, f(x_i)) \qquad (3)$$

Learning methods

Now we have the discrete learning goal. But by which method?

Types of learning: Learning methods

- Supervised
  L is independent of x, so
  $$L(x, y, f(x)) = \tilde{L}(y, f(x))$$
  . Means labels/output values for an input are known; regression is supervised learning.
- Unsupervised
  L is independent of y, so
  $$L(x, y, f(x)) = \tilde{L}(x, f(x))$$
  . Labels/outputs are not known; clustering algorithms are a famous example.

Trivial Learning

Recall the ERM problem

$$f_D = \arg\min_{f \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} L(x_i, y_i, f(x_i)).$$

- Recall: $\mathcal{H}$ contains all possible mappings
- Obviously $f(x) = \begin{cases} y_k & x = x_k \\ 0 & \text{else} \end{cases}, x \in X$ is a minimizer!
- .. but very bad generalization (*overfitting*).

Trivial Learning

Recall the ERM problem

$$f_D = \arg\min_{f \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} L(x_i, y_i, f(x_i)).$$

- Recall: $\mathcal{H}$ contains all possible mappings
- Obviously $f(x) = \begin{cases} y_k & x = x_k \\ 0 & \text{else} \end{cases}, x \in X$ is a minimizer!
- .. but very bad generalization (*overfitting*).

So we need

Regularized ERM

For some $\lambda > 0$ use (assume $\mathcal{H}$ is normed)

$$f_D = \arg\min_{f \in \mathcal{H}} \lambda \, ||f||_{\mathcal{H}}^2 + R_{L,P_D}(f) \tag{4}$$

Why regularization?

Let us motivate this choice:

- Let f_D be a ERM result, $f \equiv 0 \in \mathcal{H}$ and $L(y, f(x)) = \max\{0, 1 - yf(x)\}$ (Hinge-Loss).

Why regularization?

Let us motivate this choice:

- Let f_D be a ERM result, $f \equiv 0 \in \mathcal{H}$ and $L(y, f(x)) = \max\{0, 1 - yf(x)\}$ (Hinge-Loss).
- Then $R_{L,P_D}(f) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i)) = \frac{1}{N} \sum_{i=1}^{N} 1 = 1,$

Why regularization?

Let us motivate this choice:

- Let f_D be a ERM result, $f \equiv 0 \in \mathcal{H}$ and $L(y, f(x)) = \max\{0, 1 - yf(x)\}$ (Hinge-Loss).
- Then $R_{L,P_D}(f) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) = \frac{1}{N}\sum_{i=1}^{N} 1 = 1$,
- So $\lambda \,||f_D||_{\mathcal{H}}^2 \leq \lambda \,||f_D||_{\mathcal{H}}^2 + R_{L,P_D}(f_D) \leq \underbrace{\lambda \,||0||_{\mathcal{H}}^2}_{=0} + \underbrace{R_{L,P_D}(0)}_{=1} = 1$,

Why regularization?

Let us motivate this choice:

- Let f_D be a ERM result, $f \equiv 0 \in \mathcal{H}$ and $L(y, f(x)) = \max\{0, 1 - yf(x)\}$ (Hinge-Loss).
- Then $R_{L,P_D}(f) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i)) = \frac{1}{N} \sum_{i=1}^{N} 1 = 1$,
- So $\lambda \, ||f_D||_{\mathcal{H}}^2 \leq \lambda \, ||f_D||_{\mathcal{H}}^2 + R_{L,P_D}(f_D) \leq \underbrace{\lambda \, ||0||_{\mathcal{H}}^2}_{=0} + \underbrace{R_{L,P_D}(0)}_{=1} = 1$,
- But this means $||f_D||_{\mathcal{H}} \leq \sqrt{\frac{1}{\lambda}}$.

Why regularization?

Let us motivate this choice:

- Let f_D be a ERM result, $f \equiv 0 \in \mathcal{H}$ and $L(y, f(x)) = \max\{0, 1 - yf(x)\}$ (Hinge-Loss).
- Then $R_{L,P_D}(f) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) = \frac{1}{N}\sum_{i=1}^{N} 1 = 1$,
- So $\lambda\,||f_D||_{\mathcal{H}}^2 \leq \lambda\,||f_D||_{\mathcal{H}}^2 + R_{L,P_D}(f_D) \leq \underbrace{\lambda\,||0||_{\mathcal{H}}^2}_{=0} + \underbrace{R_{L,P_D}(0)}_{=1} = 1$,
- But this means $||f_D||_{\mathcal{H}} \leq \sqrt{\frac{1}{\lambda}}$.
- Let

$$\mathcal{H}_\lambda := \left\{ f \in \mathcal{H} \;\middle|\; ||f||_{\mathcal{H}} \leq \lambda^{-\frac{1}{2}} \right\} \subset \mathcal{H}$$

Why regularization?

Let us motivate this choice:

- Let f_D be a ERM result, $f \equiv 0 \in \mathcal{H}$ and $L(y, f(x)) = \max\{0, 1 - yf(x)\}$ (Hinge-Loss).
- Then $R_{L,P_D}(f) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) = \frac{1}{N}\sum_{i=1}^{N} 1 = 1$,
- So $\lambda\,||f_D||_{\mathcal{H}}^2 \leq \lambda\,||f_D||_{\mathcal{H}}^2 + R_{L,P_D}(f_D) \leq \underbrace{\lambda\,||0||_{\mathcal{H}}^2}_{=0} + \underbrace{R_{L,P_D}(0)}_{=1} = 1$,
- But this means $||f_D||_{\mathcal{H}} \leq \sqrt{\frac{1}{\lambda}}$.
- Let
$$\mathcal{H}_\lambda := \left\{ f \in \mathcal{H} \;\middle|\; ||f||_{\mathcal{H}} \leq \lambda^{-\frac{1}{2}} \right\} \subset \mathcal{H}$$
- So f_D is also a solution of
$$f_D = \arg\min_{f \in \mathcal{H}_\lambda} \lambda\,||f||_{\mathcal{H}}^2 + R_{L,P_D}(f).$$

Why regularization?

Let us motivate this choice:

- Let f_D be a ERM result, $f \equiv 0 \in \mathcal{H}$ and $L(y, f(x)) = \max\{0, 1 - yf(x)\}$ (Hinge-Loss).
- Then $R_{L,P_D}(f) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) = \frac{1}{N}\sum_{i=1}^{N} 1 = 1$,
- So $\lambda\,||f_D||_{\mathcal{H}}^2 \leq \lambda\,||f_D||_{\mathcal{H}}^2 + R_{L,P_D}(f_D) \leq \underbrace{\lambda\,||0||_{\mathcal{H}}^2}_{=0} + \underbrace{R_{L,P_D}(0)}_{=1} = 1$,
- But this means $||f_D||_{\mathcal{H}} \leq \sqrt{\frac{1}{\lambda}}$.
- Let
$$\mathcal{H}_\lambda := \left\{ f \in \mathcal{H} \;\middle|\; ||f||_{\mathcal{H}} \leq \lambda^{-\frac{1}{2}} \right\} \subset \mathcal{H}$$
- So f_D is also a solution of
$$f_D = \arg\min_{f \in \mathcal{H}_\lambda} \lambda\,||f||_{\mathcal{H}}^2 + R_{L,P_D}(f).$$
- Functions f with $||f||_{\mathcal{H}} > \lambda^{-\frac{1}{2}}$ are never solutions to the ERM problem!

Why regularization?

Let us motivate this choice:

- Let f_D be a ERM result, $f \equiv 0 \in \mathcal{H}$ and $L(y, f(x)) = \max\{0, 1 - yf(x)\}$ (Hinge-Loss).
- Then $R_{L,P_D}(f) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) = \frac{1}{N}\sum_{i=1}^{N} 1 = 1$,
- So $\lambda \|f_D\|_{\mathcal{H}}^2 \le \lambda \|f_D\|_{\mathcal{H}}^2 + R_{L,P_D}(f_D) \le \underbrace{\lambda \|0\|_{\mathcal{H}}^2}_{=0} + \underbrace{R_{L,P_D}(0)}_{=1} = 1$,
- But this means $\|f_D\|_{\mathcal{H}} \le \sqrt{\frac{1}{\lambda}}$.
- Let
$$\mathcal{H}_\lambda := \left\{ f \in \mathcal{H} \;\middle|\; \|f\|_{\mathcal{H}} \le \lambda^{-\frac{1}{2}} \right\} \subset \mathcal{H}$$
- So f_D is also a solution of
$$f_D = \arg\min_{f \in \mathcal{H}_\lambda} \lambda \|f\|_{\mathcal{H}}^2 + R_{L,P_D}(f).$$
- Functions f with $\|f\|_{\mathcal{H}} > \lambda^{-\frac{1}{2}}$ are never solutions to the ERM problem!
- f with smaller norm are "smoother", hopefully better generalization!

Method 2: Support Vector Machines

Connection to original setting

Back to the original setting

- Want to compute the coefficients α of our kernel function $f(x) = \sum_{i=1}^{N} \alpha_i \Phi(x, x_i)$.
- Choose $\mathcal{N}_\Phi(\Omega) \subset \mathcal{H}$!
- ERM problem is now

$$f_D = \arg \min_{f \in \mathcal{N}_\Phi(\Omega)} \lambda ||f||_{\mathcal{H}}^2 + R_{L,P_D}(f). \tag{5}$$

Method 2: Support Vector Machines

Connection to original setting

Back to the original setting

- Want to compute the coefficients α of our kernel function $f(x) = \sum_{i=1}^{N} \alpha_i \Phi(x, x_i)$.
- Choose $\mathcal{N}_\Phi(\Omega) \subset \mathcal{H}$!
- ERM problem is now

$$f_D = \arg \min_{f \in \mathcal{N}_\Phi(\Omega)} \lambda ||f||_{\mathcal{H}}^2 + R_{L,P_D}(f). \tag{5}$$

ϵ-insensitive loss function

$$L(x, y, f(x)) = |y - f(x)|_\epsilon := \begin{cases} |y - f(x)| - \epsilon, & |y - f(x)| > \epsilon \\ 0, & \text{else} \end{cases}$$

- Samples approximated better than ϵ are not considered
- Possibility of a sparse representation

Method 2: Support Vector Machines

Quadratic optimization!

Final SVR optimization problem

$$\min_{f \in \mathcal{N}_\Phi(\Omega)} \lambda ||f||^2_{\mathcal{N}_\Phi(\Omega)} + \frac{1}{N} \sum_{i=1}^{N} |y_i - f(x_i)|_\epsilon \tag{6}$$

$$= \min_{\alpha \in \mathbb{R}^N} \lambda \sum_{i,j}^{N} \alpha_i \alpha_j \Phi(x_i, x_j) + \frac{1}{N} \sum_{k}^{N} \left| y_k - \sum_{i=1}^{N} \alpha_i \Phi(x_k, x_i) \right|_\epsilon \tag{7}$$

- Corresponds to a quadratic optimization problem for $\boldsymbol{\alpha} \in \mathbb{R}^N$
- The *Representer Theorem* guarantees $f_D \in F_\Phi$!
- For more information on solving those problems see [2, §9.1ff], for example.

Method 2: Support Vector Machines
An example

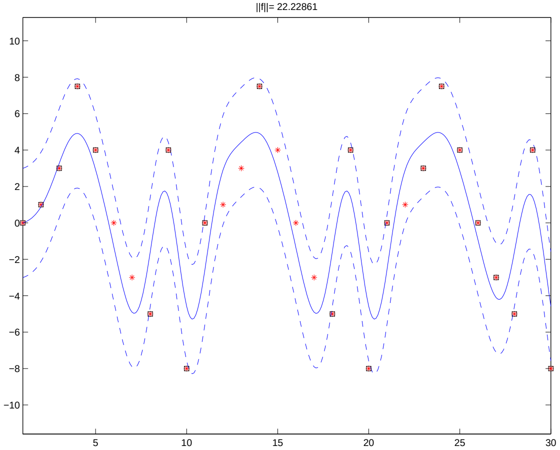

Figure: Kernel SVM example with Gaussian kernel ($\gamma = 2.5, \epsilon = 3$)

Part II

Model Reduction with Kernels

Model Order Reduction of Biochemical Systems

Motivating example

Biochemical systems: cell apoptosis simulation

- Described by PDE
- Spatial discretization yields large scale dynamical system
- Often also parameterized, i.e. TNF receptor inputs

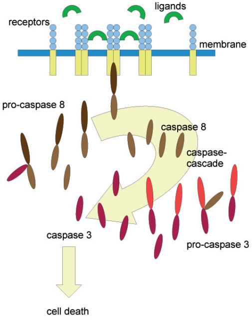

Figure: Cell death model

Current settings

Base dynamical system

Base System

$$
\begin{aligned}
x'(t) &= f(x) + Bu(t) && (8)\\
x(0) &= x_0 && (9)\\
y(t) &= Cx(t) && (10)
\end{aligned}
$$

- System state $x(t) \in \mathbb{R}^d$ at times $t \in [0, T]$
- Nonlinear system function $f : \mathbb{R}^d \to \mathbb{R}^d$
- Varying input function $u : [0, T] \to \mathbb{R}^m$ and $B \in \mathbb{R}^{d \times m}$
- Initial state $x_0 \in \mathbb{R}^d$
- Output conversion $C \in \mathbb{R}^{k \times d}$

Reduction methods

Projection

Projection matrices

Compute matrices

$$V, W \in \mathbb{R}^{d \times r}$$

with $W^t V = I_r$ (biorthogonality) and $r \leq d$ (ideally $r << d$).

Reduction by projection

Projection of the base system (8) into the space spanned by V:

$$z'(t) = W^t f(Vz(t)) + W^t Bu(t) \quad (11)$$

$$z(0) = W^t x_0 =: z_0 \quad (12)$$

$$y^r(t) = CVz(t) \quad (13)$$

- Reduced state variable $z(t) \in \mathbb{R}^r$ at times $t \in [0, T]$
- Reduced in- and output matrices $W^t B \in \mathbb{R}^{r \times m}, CV \in \mathbb{R}^{r \times k}$

Reduction methods

Approximation by kernel expansions

Central idea

Approximate f from system (8) by a kernel expansion $\hat{f}$:

$$\hat{f}(x) = \sum_{i=1}^{N} \boldsymbol{\alpha}\Phi(x, x_i)$$

- Requires N centers $x_i \in \mathbb{R}^d,\ i = 1 \ldots N$
- Coefficient vectors $\boldsymbol{\alpha} \in \mathbb{R}^d,\ i = 1 \ldots N$

Reduction methods

Approximation by kernel expansions

Central idea

Approximate f from system (8) by a kernel expansion $\hat{f}$:

$$\hat{f}(x) = \sum_{i=1}^{N} \boldsymbol{\alpha}\Phi(x, x_i)$$

- Requires N centers $x_i \in \mathbb{R}^d,\ i = 1 \ldots N$
- Coefficient vectors $\boldsymbol{\alpha} \in \mathbb{R}^d,\ i = 1 \ldots N$

Sample a solution

- Choose & fix initial values, inputs, parameters..
- Choose times $0 \leq t_1 \ldots t_N \leq T$
- Compute $x_i := x(t_i)$, $i = 1 \ldots N$ (Expensive!)
- For efficient argument evaluation: Set $x_i := VV^t x_i$

Reduction methods

Combination

Combine both methods

- Projection of kernel expansions [1]
- Combination of form

$$\hat{f}^r(\hat{z}) := W^t \hat{f}(V\hat{z}) = W^t \sum_{i=1}^{N} \boldsymbol{\alpha}\Phi(V\hat{z}, x_i)$$

Reduction methods
Combination

Efficient argument evaluations

- $\Phi(V\hat{z}, x_i)$ high dimensional
- Lossless reduction possible for special class of kernels
- Assume

$$x_i = Vz_i \qquad i = 1 \ldots N$$

Translation- and rotation invariant kernels (Gaussian)

Assume $\Phi(x, y) = \phi(||x - y||_G)$. Then

$$\Phi(V\hat{z}, x_i) = \phi(||V\hat{z} - Vz_i||_G) = \phi(||\hat{z} - z_i||_{V^tGV}) =: \Phi^r(\hat{z}, z_i),$$

with $V^tGV \in \mathbb{R}^{r \times r}$.

Projection of kernel expansions

Coefficient vectors and reduced system

Coefficient projection

Here we have

$$\hat{f}^r(z) = W^t \sum_{i=1}^{N} \boldsymbol{\alpha} \Phi(Vz, x_i) = \sum_{i=1}^{N} \boldsymbol{\beta}_i \Phi(Vz, x_i)$$

with

$$\boldsymbol{\beta}_i := W^t \boldsymbol{\alpha} \in \mathbb{R}^r \text{ instead of } \boldsymbol{\alpha} \in \mathbb{R}^d.$$

Projection of kernel expansions

Coefficient vectors and reduced system

Coefficient projection

Here we have

$$\hat{f}^r(z) = W^t \sum_{i=1}^{N} \boldsymbol{\alpha} \Phi(Vz, x_i) = \sum_{i=1}^{N} \boldsymbol{\beta}_i \Phi(Vz, x_i)$$

with

$$\boldsymbol{\beta}_i := W^t \boldsymbol{\alpha} \in \mathbb{R}^r \text{ instead of } \boldsymbol{\alpha} \in \mathbb{R}^d.$$

Reduced system with all low dimensional components

$$\hat{z}'(t) = W^t \hat{f}(V\hat{z}(t)) + W^t Bu(t) = \sum_{i=1}^{N} \boldsymbol{\beta}_i \Phi^r(\hat{z}(t), z_i) + W^t Bu(t) \quad (14)$$

$$\hat{z}(0) = \hat{z}_0 := z_0 = W^t x_0 \quad (15)$$

$$y^r(t) = CV\hat{z}(t) \quad (16)$$

Finally..

Thank you for your attention!

References I

[1] J. Phillips, J. Afonso, A. Oliveira, and L.M. Silveira.
Analog macromodeling using kernel methods.
In *Proc. of ICCAD-2003, International Conference on Computer Aided Design*, pages 446–453, November 2003.

[2] B. Schölkopf and J. A. Smola.
Learning with Kernels.
Adaptive Computation and Machine Learning. The MIT Press, 2002.