Machine Learning

Lecture 5 – Kernel methods

UPPSALA UNIVERSITET

Thomas Schön
Division of Systems and Control
Department of Information Technology
Uppsala University.

Email: thomas.schon@it.uu.se,
www: `user.it.uu.se/~thosc112`

Contents – lecture 5

Chapter 6.4 – 7.2 (Chapter 12 in HTF, GP not covered in HTF)

Summary of lecture 4 (I/II)

A **neural network** is a nonlinear function (as a function expansion) from a set of input variables to a set of output variables controlled by adjustable parameters w.

This function expansion is found by formulating the problem as usual, which results in a (non-convex) optimization problem. This problem is solved using numerical methods.

Backpropagation refers to a way of computing the gradients by making use of the chain rule, combined with clever reuse of information that is needed for more than one gradient.

Summary of lecture 4 (II/II)

A kernel function $k(x,z)$ is defined as an inner product

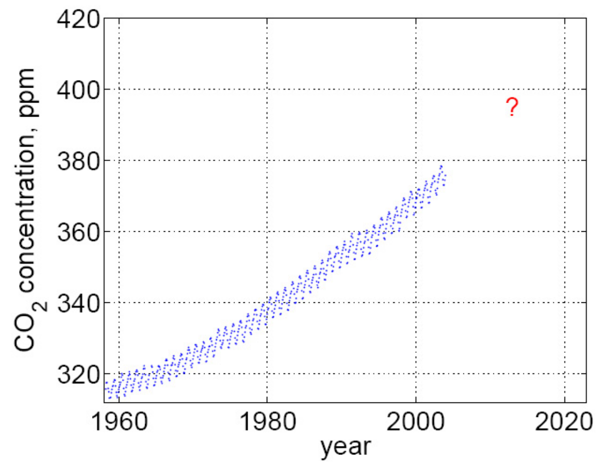
$$k(x,z) = \phi(x)^T\phi(z),$$

where $\phi(x)$ is a fixed mapping.

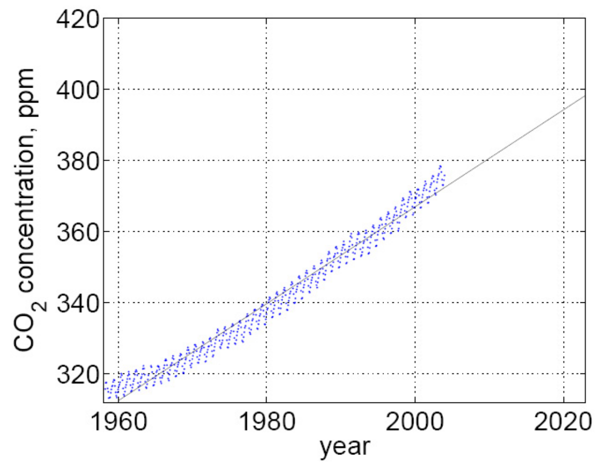
Introduced the **kernel trick** (a.k.a. kernel substitution). In an algorithm where the input data x enters only in the form of scalar products we can replace this scalar product with another choice of kernel.

The use of kernels allows us to implicitly use basis functions of high, even infinite, dimensions ($M \to \infty$).

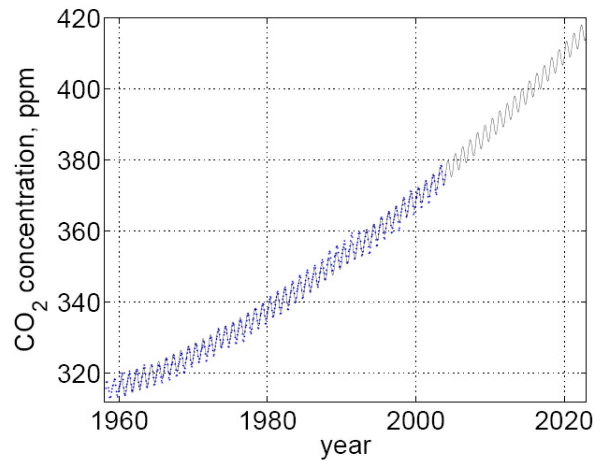
Introductory example (I/IV)



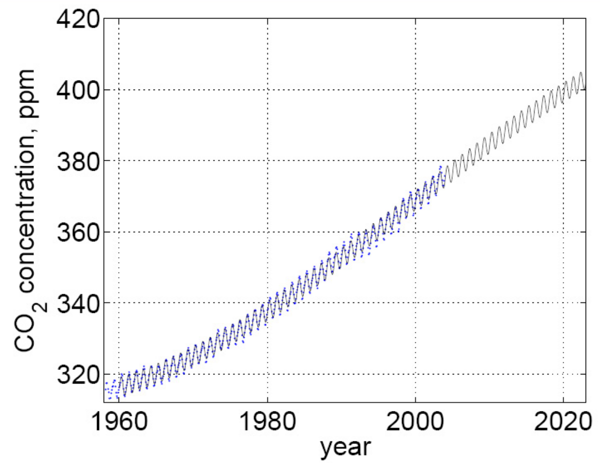
Introductory example (II/IV)



Introductory example (III/IV)



Introductory example (IV/IV)



Stochastic processes

Definition (Stochastic process): A stochastic process can be defined as a family of random variables $\{y(x), x \in \mathcal{X}\}$.

Property: For a fixed $x \in \mathcal{X}$, $y(x)$ is a random variable.

Examples: Wiener process, Chinese restaurant process, Dirichlet processes, Poisson process, Gaussian process, Markov process.

Åström K. J. (2006). **Introduction to Stochastic Control Theory**. Dover Publications, Inc., NY, USA.

Linear regression model on matrix form

Write the linear regression model (without noise) as

$$y_n = w^T \phi(x_n), \qquad n = 1, \dots, N,$$

where $w = \begin{pmatrix} w_0 & w_1 & \dots & w_{M-1} \end{pmatrix}^T$ and $\phi = \begin{pmatrix} 1 & \phi_1(x_n) & \dots & \phi_{M-1}(x_n) \end{pmatrix}^T$ on matrix form

$$Y = \Phi w,$$

where

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix} \qquad \Phi = \begin{pmatrix} \phi_0(x_1) & \phi_1(x_1) & \dots & \phi_{M-1}(x_1) \\ \phi_0(x_2) & \phi_1(x_2) & \dots & \phi_{M-1}(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(x_N) & \phi_1(x_N) & \dots & \phi_{M-1}(x_N) \end{pmatrix}$$

Gram matrix made up of kernels

The matrix K is formed from covariance functions (kernels) $k(x_n, x_m)$

$$K_{n,m} = k(x_n, x_m)$$

and it is referred to as the **Gram matrix**.

Definition (covariance function (kernel)): Given any collection of points $x_1, \dots, x_N$, a covariance function $k(x_n, x_m)$ defines the elements of an $N \times N$ matrix

$$K_{n,m} = k(x_n, x_m),$$

such that K is positive semidefinite.

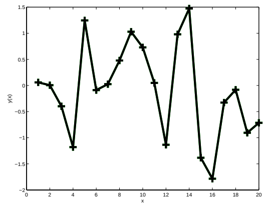
A Gaussian Process (GP)

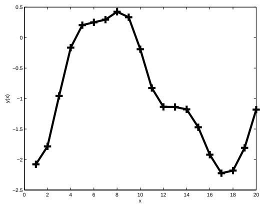
Definition (Gaussian process): A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution.

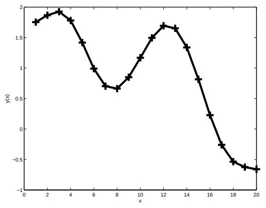
What does this mean?

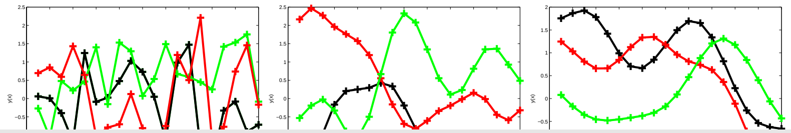
Samples from a GP

Let $y(x)$ be a Gaussian process with mean function $m = 0$ and a covariance function $\text{Cov}(y(x), y(x')) = k(x, x') = e^{-(x-x')^2/l}$. Let $x = 1 : 20$. Samples from this GP are shown below.


$l = 1$


$l = 10$


$l = 20$

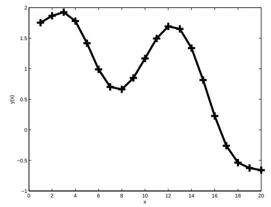


GP as a distribution over functions

It is commonly said that the GP defined by

$$p(Y \mid X) = \mathcal{N}(Y \mid 0, K),$$

specifies a *distribution over functions*. The term “function” is potentially confusing, since it merely referes to a set of outputs values $y_1, \ldots, y_N$ that corresponds to a set of input variables $x_1, \ldots, x_N$.



Hence, there is no explicit functional form for the input-output map.

Gaussian Process Regression (GPR)

So how do we use this for regression? Assume that we are given the training data $\{(y_n, x_n)\}_{n=1}^{N}$ and that we seek an estimate for $y(x^*)$. If we then assume that $y(x)$ can be modeled by a GP, we have

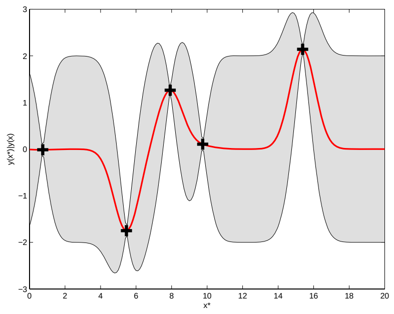
$$\begin{bmatrix} \mathbf{y} \\ y(x^*) \end{bmatrix} \sim N\left(\begin{bmatrix} m(\mathbf{x}) \\ m(x^*) \end{bmatrix}, \begin{bmatrix} k(\mathbf{x}, \mathbf{x}) & k(\mathbf{x}, x^*) \\ k(x^*, \mathbf{x}) & k(x^*, x^*) \end{bmatrix} \right)$$

and using standard Gaussian identities (lecture 1) we obtain the predictive (or conditional) density

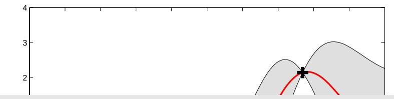
$$\begin{aligned} y(x^*) | \mathbf{y} \sim N\Big(& k(x^*, \mathbf{x}) k(\mathbf{x}, \mathbf{x})^{-1} (\mathbf{y} - m(\mathbf{x})) + m(x^*), \\ & k(x^*, x^*) - k(x^*, \mathbf{x}) k(\mathbf{x}, \mathbf{x})^{-1} k(\mathbf{x}, x^*) \Big) \end{aligned}$$

Let us try this.

GP predictive distribution



$l = 1$



Finding the hyperparameters

The parameters of the kernels (e.g. l) are often referred to as hyper-parameters and they are typically found using **empirical Bayes** (lecture 2).

Recall that empirical Bayes amounts to maximizing the log marginal likelihood

$$\begin{aligned}\max_{\text{hyperpar}} \log p(\mathbf{t}) &= \max_{\text{hyperpar}} \log \int p(\mathbf{t}|\mathbf{y})p(\mathbf{y})d\mathbf{y} \\ &= \max_{\text{hyperpar}} \log \int N(\mathbf{t};\mathbf{y},\sigma^2)N(\mathbf{y};0,k(\mathbf{x},\mathbf{x}))d\mathbf{y} \\ &= \max_{\text{hyperpar}} -\frac{1}{2}\mathbf{t}^T\left(\sigma^2 I + k(\mathbf{x},\mathbf{x})\right)^{-1}\mathbf{t} - \frac{1}{2}\log|\sigma^2 I + k(\mathbf{x},\mathbf{x})| - \frac{N}{2}\log 2\pi\end{aligned}$$

Some kernels

The kernel k (i.e. the covariance function) should ideally be $k(x_t, x_s) = \text{Cov}(y_t, y_s)$. Some common choices are:

- The γ-exponential kernel

$$k(x_t, x_s) = me^{-\|x_t - x_s\|^\gamma / l}, \qquad 0 < \gamma \leq 2,$$

  where m and l have to be chosen. Squared exponential ($\gamma = 2$), Ornstein-Uhlenbeck ($\gamma = 1$).
- The Matérn kernel

$$k(x_t, x_s) = \|x_t - x_s\|^\nu K_\nu(\|x_t - x_s\|)$$

  where K_ν is a modified Bessel function, $\nu > 0$.
- ...

Some GP properties

- Probabilistic
- Discriminative
- Nonparametric (a member of the model class referred to as Bayesian nonparametric (BNP) models (lecture 11)).
- Classification can also be done.
- Known under many names, e.g. Kriging (Daniel Krige, 1951).
- Can only handle Gaussian measurement noise.
- Multidimensional output
- Have to invert an $N \times N$ matrix, $\mathcal{O}(N^3)$ computational complexity. There are techniques to reduce this.
- Strong relations to neural networks.

Example – CO_2

What covariance functions to chose?

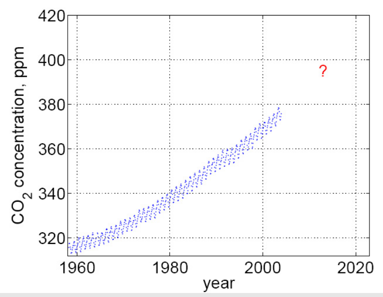
- There is a long-term smooth trend

$$k_1(x, x') = \theta_1^2 e^{-(x-x')^2/\theta_2^2}$$

- There is a periodic component

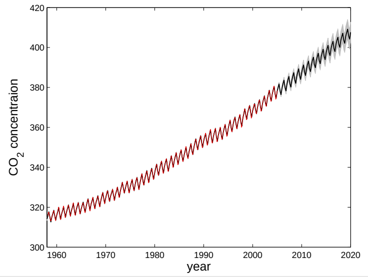
$$k_2(x, x') = \theta_3^2 e^{-\left(\sin\left(\pi(x-x')\right)\right)^2/\theta_4^2} e^{-(x-x')^2/\theta_5^2}$$

- ...

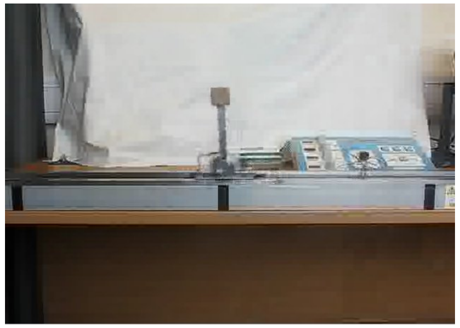


Example – CO_2

```
k1 = 'covSEiso';
k2 = {'covProd',{'covSEiso','covPeriodic'}};
k3 = 'covRQiso';
k4 = {'covSum',{'covSEiso', 'covNoise'}};
covfunc = {'covSum',{k1,k2,k3,k4}};
init=[1 1 1 1 0.3 -0.1 0.12 -0.4 -0.06 -2 -1.69 -1.6]';
loghypers = minimize(init,'gpr',-100,covfunc,x,y);
xstar=(1958:0.2:2020)';
[mu S2] = gpr(loghypers, covfunc, x, y, xstar);
```



Example – inverted pendulum



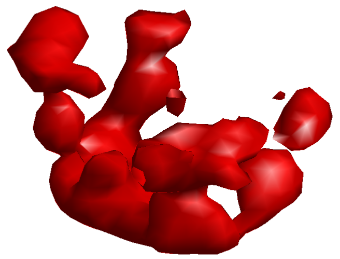
Research conducted by Marc Deisenroth and Carl Rasmussen.
Movie: `http://www.youtube.com/watch?v=XiigTGKZfks`

Deisenroth, M. **Efficient Reinforcement Learning using Gaussian Processes**, PhD thesis, Karlsruhe Institute of Technology. 2011.

Example – GPs obeying Maxwell's equations

Idea: Make use of GPs (obeying Maxwell's equations) in modeling the magnetic field and the magnetic sources in complex environments. The result is a map of magnetized items.

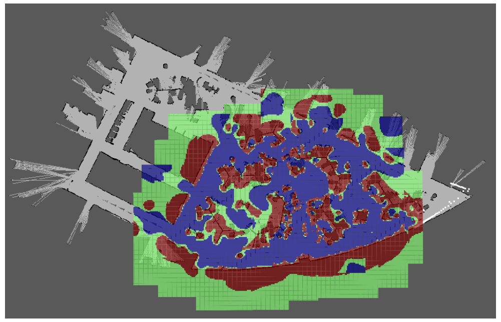
Preliminary results:



Niklas Wahlström, Manon Kok, Thomas B. Schön and Fredrik Gustafsson. **Modeling magnetic fields using Gaussian processes**. In *Proceedings of the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vancouver, Canada, May 2013.

Example – Occupancy grid maps

Idea: Build continuous occupancy maps using GPs. Allows for spatial correlation to be accounted for in a natural way, and a priori discretization of the area is not necessary as within most standard methods. Downside: computationally complex.



Wågberg, J. and Walldén Viklund, E. **Continuous occupancy mapping using Gaussian processes**. Master's thesis. Department of Electrical Engineering, Linköping University, Sweden.

Example – The Wiener model

$$\begin{tikzcd} & v_t \arrow[d] & & & e_t \arrow[d] & \\ u_t \arrow[r] & \boxed{\mathcal{L}} \arrow[r, "z_t"] & \boxed{h(\cdot)} \arrow[rr] & & \Sigma \arrow[r] & y_t \end{tikzcd}$$

A Wiener model is a linear dynamical model ($\mathcal{L}$) followed by a static nonlinearity $(h(\cdot))$.

Learning problem: Find $\mathcal{L}$ and $h(\cdot)$ based on $\{u_{1:T}, y_{1:T}\}$.

$$x_{t+1} = \underbrace{\begin{pmatrix} A & B \end{pmatrix}}_{\Gamma} \begin{pmatrix} x_t \\ u_t \end{pmatrix} + v_t, \qquad v_t \sim \mathcal{N}(0, Q),$$

$$z_t = Cx_t.$$

$$y_t = h(z_t) + e_t, \qquad e_t \sim \mathcal{N}(0, R).$$

Example – semiparametric Wiener models (I/II)

Idea: Use a Gaussian process to model the static nonlinearity $h(\cdot)$,

$$h(\cdot) \sim \mathcal{GP}(z, k(z, z')).$$

Solve the resulting problem using a Gibbs sampler (lecture 10). More specifically this is a Particle Gibbs sampler with ancestor sampling (PGAS).

PGAS is one member of the family of particle MCMC (PMCMC) algorithms recently introduced in

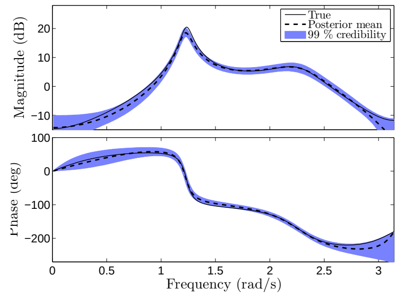
Christophe Andrieu, Arnaud Doucet and Roman Holenstein, **Particle Markov chain Monte Carlo methods**, *Journal of the Royal Statistical Society: Series B*, 72:269-342, 2010.

PGAS was introduced in

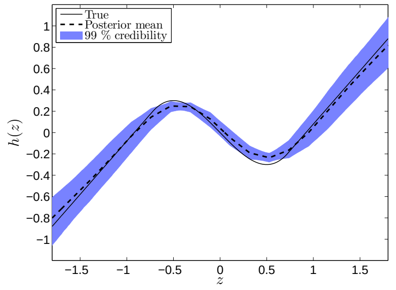
Fredrik Lindsten, Michael I. Jordan and Thomas B. Schön. **Ancestor sampling for particle Gibbs**. in *Advances in Neural Information Processing Systems (NIPS) 25*, Lake Tahoe, NV, USA, 2012.

Example – semiparametric Wiener models (II/II)

Show movie



Bode diagram of the 4th-order linear system. Estimated mean (dashed black), true (solid black) and 99% credibility intervals (blue).



Static nonlinearity (non-monotonic), estimated mean (dashed black), true (black) and the 99% credibility intervals (blue).

Fredrik Lindsten, Thomas B. Schön and Michael I. Jordan. **Bayesian semiparametric Wiener system identification**. *Automatica*, 49(7): 2053-2063, July 2013.

Gaussian process regression – illustration

Illustration of the use of GPs in the previous example.

GP state space models for nonlinear systems

We have been able to construct and learn a Gaussian process (GP) state space model

$$\begin{aligned} f(x_t) &\sim \mathcal{GP}(m_{\theta_x}(x_t), k_{\theta_x}(x_t, x_t')), \\ x_{t+1} \mid f_t &\sim \mathcal{N}(x_{t+1} \mid f_t, Q), \\ y_t \mid x_t &\sim p(y_t \mid x_t, \theta_y). \end{aligned}$$

Key idea: Marginalize out the entire function f.

Problem: Renders the model non-Markovian. **Solution:** PGAS

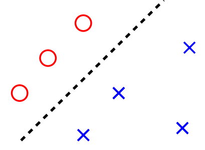
For details, see

Roger Frigola, Fredrik Lindsten, Thomas B. Schön and Carl E. Rasmussen, **Bayesian inference and learning in Gaussian process state-space models with particle MCMC**. In *Advances in Neural Information Processing Systems (NIPS) 26*, Lake Tahoe, NV, USA, December 2013.

Support Vector Machines (SVM)

Very popular classifier.

- Non-probabilistic
- Discriminative
- Can also be used for regression (then called *support vector regression*, SVR).
- Convex optimization
- Sparse
- SMV are often used to illustrate the interplay between optimization and machine learning.



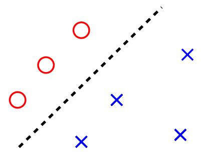
SVM for classification (I/IV)

Assume: $\{(t_n, x_n)\}_{n=1}^{N}$, $x_n \in \mathbb{R}^{n_x}$ and $t_n \in \{-1, 1\}$, is a given training data set (linearly separable).

Task: Given x^*, what is the corresponding label?

SVM is a discriminative classifier, i.e. it provides a decision boundary. The decision boundary is given by $\{x | w^T\phi(x) + b = 0\}$.

Goal: Find the decision boundary that maximizes the margin! The *margin* is the distance to the closest point on the decision boundary.



SVM for classification (II/IV)

The decision boundary that maximizes the margin is given as the solution to the **quadratic program (QP)**

$$\begin{aligned} &\min_{w,b} \frac{1}{2}\|w\|^2 \\ &\text{s.t.} \quad t_n(w^T\phi(x_n) + b) - 1 \geq 0, \quad n = 1, \dots, N. \end{aligned}$$

To make it possible to let the dimension of the feature space (dim of $\phi(x_n)$) go to infinity, we have to work with the **dual problem**.

SVM for classification (III/IV)

First, the Lagrangian is

$$L(w, b, \mathbf{a}) = \frac{1}{2}\|w\|^2 - \sum_{n=1}^{N} a_n \Big(t_n (w^T \phi(x_n) + b) - 1 \Big)$$

and minimizing w.r.t. w, b we obtain the dual objective $g(\mathbf{a})$. Taking the derivative w.r.t. w, b and set them to zero,

$$\frac{dL(w, b, \mathbf{a})}{db} = \sum_{n=1}^{N} a_n t_n = 0, \quad \frac{dL(w, b, \mathbf{a})}{dw} = w - \sum_{n=1}^{N} a_n t_n \phi(x_n) = 0.$$

This gives

$$g(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{m=1}^{N} \sum_{n=1}^{N} a_n a_m t_n t_m \phi(x_m)^T \phi(x_n).$$

SVM for classification (IV/IV)

Let $k(x_i, x_j) = \phi(x_i)^T \phi(x_j)$. The **dual objective** then becomes

$$g(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{m=1}^{N} \sum_{n=1}^{N} a_n a_m t_n t_m k(x_m, x_n)$$

which we can maximize w.r.t. $\mathbf{a}$ and subject to

$$a_n \geq 0, \qquad \sum_{n=1}^{N} a_n t_n = 0.$$

The maximizing $\mathbf{a}$ (let us call it $\hat{\mathbf{a}}$) gives using $w^T \phi(x^*) = (\sum_{n=1}^{N} a_n t_n \phi(x_n))^T \phi(x^*)$ that

$$y(x^*) = \sum_{n=1}^{N} \hat{a}_n t_n k(x^*, x_n) + b.$$

Many $\hat{a}$'s will be zero (**sparseness**) $\Rightarrow$ computational remedy.

Support vectors – sparse version of training data

It can be shown that the KKT conditions for this optimization problem satisfies

$$\begin{aligned} a_n &\geq 0, \\ t_n y(x_n) - 1 &\geq 0, \\ a_n (t_n y(x_n) - 1) &= 0. \end{aligned}$$

The result is that for each training data the following is true

1. Either $a_n = 0$ or
2. $t_n y(x_n) = 1$.

Training data with $a_n = 0$ do not appear in the solution. The remaining training data (i.e., where $t_n y_n = 1$) are referred to as **support vectors** (training data that lie on the maximum margin decision boundary).

SVM for classification – non-separable classes

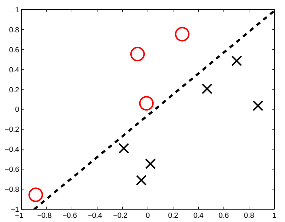
If points are on the right side of the decision boundary, then $t_n(w^T \phi(x_n) + b) \geq 1$. To allow for some violations, we introduce slack variables ζ_n, $n = 1, \dots, N$. The modified optimization problem becomes

$$\begin{aligned} &\min_{w,b,\zeta} \frac{1}{2}\|w\|^2 + C \sum_{n}^{N} \zeta_n \\ \text{s.t.} \quad & t_n(w^T \phi(x_n) + b) + \zeta_n - 1 \geq 0, \quad n = 1, \dots, N, \\ & \zeta_n \geq 0, \quad n = 1, \dots, N. \end{aligned}$$

Example – CVX to compute SVM (I/II)

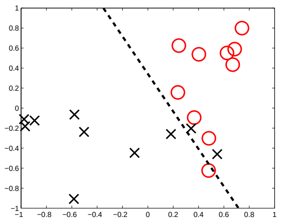
Linearly separable data:

```
cvx_begin
variables w(nx,1) b
minimize (0.5∗w'∗w)
  subject to
   y.∗(w'∗x+b∗ones(1,N))−ones(1,N) >= 0
cvx_end
```



Non-separable data:

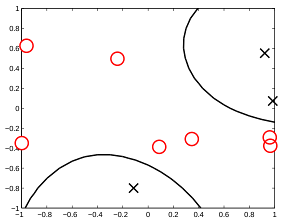
```
cvx_begin
variables w(nx,1) b zeta(1,N)
minimize (0.5∗w'∗w + C∗ones(1,N)∗zeta')
  subject to
   y.∗(w'∗x+b∗ones(1,N))−ones(1,N)+zeta >= 0
   zeta >= 0
cvx_end
```



Example – CVX to compute SVM (II/II)

SVM – Solving the dual:

```
k=@(x1,x2) exp(−sum((x1∗ones(1,size(x2,2))−x2).^2)/0.5)'
 for t=1:N;for s=t:N
K(t,s)=k(x(:,t),x(:,s));K(s,t)=K(t,s);
end;end
cvx_begin
 variables a(N,1)
 minimize( 1/2∗(a.∗y')'∗K∗(a.∗y') − ones(1,N)∗a)
   subject to
    ones(1,N)∗(a.∗y') == 0
    a >= 0
cvx_end
ind=find(a>0.01);
wphi = @(xstar) ones(1,N)∗(a.∗y'.∗k(xstar,x))
b=0;
for i=1:length(ind)
b=b+1/y(ind(i))−wphi(x(:,ind(i)));
end
b=b/length(ind);
 ystar = @(xstar) wphi(xstar)+b
```



A few concepts to summarize lecture 5

Kernel: Kernel is another name for covariance function, i.e., a function k that depends in input data in the following way $k(x_m, x_n) = \phi(x_m)^T\phi(x_n)$.

Gaussian processes (GP): A GP is a collection of random variables, any finite number of which have a joint Gaussian distribution.

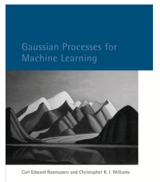
Gaussian process regression: Assume that the underlying process can be modeled using a GP. Use Gaussian identities to compute the conditional distribution.

Support vector machines: A discriminative classifier that gives the maximum margin decision boundary.

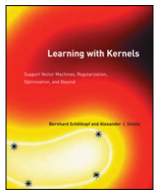
Support vector: A training data that lies on the maximum margin decision boundary.

Further reading and code

- Rasmussen & Williams, 2006 (electronic version: `www.gaussianprocess.org/gpml/`).
- Bernhard Schölkopf and Alex Smola. Learning with Kernels. MIT Press, Cambridge, MA, 2002.
- GP site `www.gaussianprocess.org/`
- Yalmip can be downloaded from `users.isy.liu.se/johanl/yalmip/`
- CVX can be downloaded from `cvxr.com/cvx/`
- GPR MATLAB toolbox can be downloaded from: `www.gaussianprocess.org/gpml/`
- Video lecture on GPR: `videolectures.net/mlss09uk_rasmussen_gp/`



Gaussian Processes for Machine Learning
Carl Edward Rasmussen and Christopher K. I. Williams



Learning with Kernels
Support Vector Machines, Regularization, Optimization, and Beyond
Bernhard Schölkopf and Alexander J. Smola