Apprentissage Statistique
Partie 2

Sorbonne Université- Master informatique DAC et Master mathématiques et applications M2A-
P. Gallinari, patrick.gallinari@lip6.fr, http://www-connex.lip6.fr/~gallinar/
Nicolas Baskiotis, Benjamin Piwowarski, Laure Soulier, nom.prenom@lip6.fr
Année 2020-2021

Apprentissage supervisé

Machines à noyaux

Machines à vecteurs support

Processus Gaussiens

Machines à noyaux

Introduction

- Familles de machines d'apprentissage générales qui exploitent l'idée suivante :
  - Projeter les données dans un espace de grande dimension - éventuellement infini -où le problème sera facile à traiter
  - Utiliser des "projections" non linéaires permettant des calculs "efficaces"
- Exemples traités en cours :
  - Machines à Vecteurs Support - classification
  - Processus Gaussien - regression

Perceptron : formulation duale pour la classification

hyp: 2 classes linéairement séparables, sorties désirées -1, 1

Perceptron primal

Initialiser $\boldsymbol{w}(0) = 0$

Répeter (t)

choisir un exemple, $(\boldsymbol{x}(t), d(t))$

si $d(t)\boldsymbol{w}(t).\boldsymbol{x}(t) \leq 0$

alors $\boldsymbol{w}(t+1) = \boldsymbol{w}(t) + d(t)\boldsymbol{x}(t)$

Jusqu'à convergence

Fonction de décision

$$F(x) = sgn(\sum_{j=0}^{n} w_j x_j),$$

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i \boldsymbol{d^i x^i}$$

α_i : nombre de fois où l'algorithme a rencontré une erreur de classification sur $\boldsymbol{x}^i$

Perceptron dual

Initialiser $\boldsymbol{\alpha} = \boldsymbol{0}, \boldsymbol{\alpha} \in \boldsymbol{R^N}$

Répeter (t)

choisir un exemple, $(\boldsymbol{x}(t), d(t))$

soit k: $\boldsymbol{x}(t) = \boldsymbol{x}^k$

si $d(t)\sum_{i=1}^{N} \alpha_i \boldsymbol{d^i x^i}.\boldsymbol{x}(t) \leq 0$

alors $\alpha_k = \alpha_k + 1$

Jusqu'à convergence

Fonction de décision

$$F(x) = sgn(\sum_{i=1}^{N} \alpha_i \boldsymbol{d^i x^i}.\boldsymbol{x})$$

Matrice de Gram G :

matrice NxN de terme i, j : $\boldsymbol{x}^i.\boldsymbol{x}^j$

matrice de similarité entre données

Formulation duale pour la regression

- On se donne un ensemble d'apprentissage
  - $D = \{(\boldsymbol{x}^1, y^1), \dots, (\boldsymbol{x}^N, y^N)\}$, on note $X = \{\boldsymbol{x}^1, \dots, \boldsymbol{x}^N\}$
- On considère un modèle linéaire pour une regression
  - $f(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x}$
  - soit $\boldsymbol{x}^\perp \in X^\perp$, l'ensemble orthogonal à X
  - $(\boldsymbol{w}^T + \boldsymbol{x}^\perp)\boldsymbol{x}^{\mathrm{i}} = \boldsymbol{w}^T \boldsymbol{x}^{\mathrm{i}}, \forall \boldsymbol{x}^{\mathrm{i}} \in X$
  - Ajouter à $\boldsymbol{w}$ une contribution qui est en dehors de l'espace engendré par X, n'a pas d'effet sur la prédiction des **données de l'ensemble d'apprentissage**
  - Si le critère d'apprentissage dépend uniquement de la prédiction des données d'apprentissage, il n'est pas nécessaire de considérer des contributions à $\boldsymbol{w}$ en dehors de X
  - On peut donc écrire $\boldsymbol{w}$ sous la forme
    - $\boldsymbol{w} = \sum_{\boldsymbol{i}=\boldsymbol{1}}^{\boldsymbol{N}} \alpha_i \boldsymbol{x}^{\boldsymbol{i}}$
    - Les paramètres $\alpha_i, i = 1 \dots N$ sont appelés les paramètres duaux
  - On peut alors écrire directement la fonction comme uns combinaison des paramètres duaux
  - $\boldsymbol{f}(\boldsymbol{x}^{\boldsymbol{j}}) = \sum_{\boldsymbol{i}=\boldsymbol{1}}^{\boldsymbol{N}} \alpha_i (\boldsymbol{x}^{\boldsymbol{i}})^{\boldsymbol{T}} \boldsymbol{x}^{\boldsymbol{j}}$

Formulation duale pour la regression

- Plus généralement si on considère une regression avec des fonctions vectorielles $\phi(x)$:
  - $f(\boldsymbol{x}) = \boldsymbol{w}^T \phi(\boldsymbol{x})$
  - La solution sera dans l'espace engendré par $\{\phi(\boldsymbol{x}^1), \dots, \phi(\boldsymbol{x}^1)\}$
  - $\boldsymbol{w} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \alpha_i \phi(\boldsymbol{x}^{\boldsymbol{i}})$
  - $f(\boldsymbol{x}^j) = \sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \alpha_i \boldsymbol{\phi}(\boldsymbol{x}^{\boldsymbol{i}})^{\boldsymbol{T}} \boldsymbol{\phi}(\boldsymbol{x}^{\boldsymbol{j}}) = \sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \alpha_i \boldsymbol{K}(\boldsymbol{x}^{\boldsymbol{i}}, \boldsymbol{x}^{\boldsymbol{j}})$
  - $\boldsymbol{K}(\boldsymbol{x}^{\boldsymbol{i}}, \boldsymbol{x}^{\boldsymbol{j}}) = \boldsymbol{\phi}(\boldsymbol{x}^{\boldsymbol{i}})^{\boldsymbol{T}} \boldsymbol{\phi}(\boldsymbol{x}^{\boldsymbol{j}}) = \boldsymbol{K}_{\boldsymbol{ij}}$ est appelée fonction noyau
  - $\boldsymbol{K} = [\boldsymbol{K}_{\boldsymbol{ij}}]$ est la matrice de Gram
  - On peut résoudre le problème de regression dans l'espace dual en cherchant la solution de:
    - $\sum_{i=1}^{n} (y^i - \alpha^T k^i)^2$ avec k^i la ieme colonne de $\boldsymbol{K}$
  -

Représentation Duale

- La plupart des machines à apprentissage linéaires ont une représentation duale
  - Exemples
    - Adaline, regression, regression ridge, etc
  - L'information sur les données est entièrement fournie par la matrice de Gram $\boldsymbol{K}$, qui joue un rôle central
  - La fonction de décision ou de regression $F(x)$ s'exprime comme une combinaison linéaire de produits scalaires entre la donnée d'entrée x ou $\phi(x)$ et les exemples d'apprentissage

- Les machines à noyau généralisent ces idées
  - Une fonction noyau K est définie sur $R^n \mathrm{x} R^n$ (on suppose $x \in R^n$) par

$$K(x,z) =< \Phi(x), \Phi(z) >$$

où Φ est une fonction de R^n dans un espace muni d'un produit scalaire

Produit Scalaire et Noyaux

- Projection non linéaire dans un espace de grande dimension H
  - (en général dim H >> n, et peut même être infinie)
  - $\Phi: R^n \rightarrow R^p$ avec $p \gg n$
  - Machine linéaire dans H - Primal :
    - $F(x) = \sum_{j=1}^{p} w_j \Phi_j(x) + b$
  - Machine linéaire dans H - Dual :
    - $w = \sum_{i=1}^{N} d^i \alpha_i \Phi(\boldsymbol{x}^i)$, $\quad \mathrm{F(x)} = \sum_{i=1}^{N} d^i \alpha_i \Phi(\boldsymbol{x}^i).\Phi(\boldsymbol{x}) + b$,
- Calculer les produits scalaires dans l'espace initial : choisir F /
  - $\Phi(x).\Phi(x') = K(x, x')$
  - $\mathrm{F(x)} = \sum_{i=1}^{N} d^i \alpha_i K(x^i, x) + b$
  - avec $\boldsymbol{K}$: fonction noyau (symmetrique)

- Généralise le produit scalaire dans l'espace initial
- Le calcul de F ne dépend pas directement de la taille de H : les calculs sont faits dans l'espace initial.
- La machine linéaire dans H peut être construite à partir d'une fonction K sans qu'il soit nécessaire de définir explicitement Φ : en pratique, on spécifiera directement K.
- Cette idée permet d'étendre de nombreuses techniques linéaires au non linéaire: il suffit de trouver des noyaux appropriés
  - Exemples
    - ACP, analyse discriminante, regression, etc

Caractérisation des noyaux

- Quand peut on utiliser cette idée ?
- Cas d'un espace fini
  - Soit $X = \{x^1, \ldots, x^N\}$, $K(x, x')$ une fonction symétrique sur X, K est une fonction noyau ssi la matrice N x N dont l'élément (i, j) est $K(x^i, x^j)$ est positive semi-définie (valeurs propres ≥ 0 ou $x^T \boldsymbol{K} x > 0 \; \forall x$, avec $\boldsymbol{K} = Matrice\left(K\left(x^i, x^j\right); i, j = 1..N\right)$)
- Cas général : Conditions de Mercer (noyaux de Mercer)
  - Il existe une application Φ et un développement

$$K(x, x') = \sum_{i=1}^{\infty} \Phi(x)_i . \Phi(x')_i$$

  - Ssi :

$$\forall g / \int g(x)^2 dx \quad \text{est fini,} \int K(x, x') g(x) g(x') dx dx' \geq 0$$

Caractérisation des noyaux
Espace de Hilbert à noyau autoreproduisant

- Une fonction $K: X * X \rightarrow R$ qui est soit continue soit définie sur un domaine fini peut s'écrire sous la forme d'un produit scalaire :

$$K(x,z) = <\phi(x), \phi(z)>$$

avec $\Phi : x \rightarrow \Phi(x) \in \boldsymbol{F}$ espace de Hilbert

ssi c'est une fonction symétrique et toutes les matrices formées par la restriction de K à un échantillon fini sur *X* sont semi-définies positives).

- Résultat à la base de la caractérisation effective des fonctions noyaux
- Il permet de caractériser *K* comme un noyau sans passer par Φ
- C'est une formulation équivalente aux conditions de Mercer

Caractérisation des noyaux
Espace de Hilbert à noyau autoreproduisant

- L'espace de Hilbert associé au noyau *K* est l'ensemble *F* des fonctions qui s'écrivent comme combinaisons linéaires:

$$F = \left\{ \sum_{i=1}^{l} \alpha_i K(x_i,.) \,/\, l \in N, x_i \in X, \alpha_i \in R, i = 1..l \right\}$$

- Le produit scalaire défini sur cet espace :

$$\text{Soient} \quad f(.) = \sum_{i=1}^{l} \alpha_i K(x_i,.), \qquad g(.) = \sum_{j=1}^{n} \beta_j K(x_j,.)$$

$$< f, g > = \sum_{i=1}^{l} \sum_{j=1}^{n} \alpha_i \beta_j K(x_i, z_j) = \sum_{i=1}^{l} \alpha_i g(x_i) = \sum_{j=1}^{n} \beta_j f(z_j)$$

- Noyau auto-reproduisant
  - Si on prend $g(.) = K(x,.)$ alors

$$< f, K(x,.) > = \sum_{i=1}^{l} \alpha_i K(x_i, x) = f(x)$$

Exemples de noyaux

$$K(x,z) = < x.z >^2$$

$$K(x,z) = \left(\sum_{i=1}^{n} x_i.z_i \right)^2 = \sum_{i,j=1}^{n} (x_i.x_j)(z_i.z_j)$$

$$K(x,z) = < \phi(x).\phi(z) > \text{ avec } \phi(x) / \phi(x)_{i,j} = (x_i.x_j)_{i,j=1,n}$$

i.e. tous les monomes de d° 2 : $\begin{pmatrix} n+1 \\ 2 \end{pmatrix}$

$$K(x,z) = (< x.z > + c)^2$$

$$K(x,z) = < \phi(x).\phi(z) > \text{ avec } \phi(x) / \phi(x)_{i,j} = ((x_i.x_j)_{i,j=1,n}, (\sqrt{2}cx_i)_{i=1,n}, c)$$

i.e. ss ensemble des polynomes de d° 2

Exemples de noyaux (suite)

$$K(x,x^i)=\begin{cases}(x.x^i+1)^d & \Phi \text{ polynome d'ordre d}\\ \exp-\gamma\left\|x-x^i\right\|^2 & \Phi \text{ noyau gaussien}\\ Sigmoïde(vx.x^i+c)\end{cases}$$

- En pratique, on utilise des noyaux:
  - Linéaires
  - Gaussiens
  - Polynomiaux

Construction des noyaux en pratique

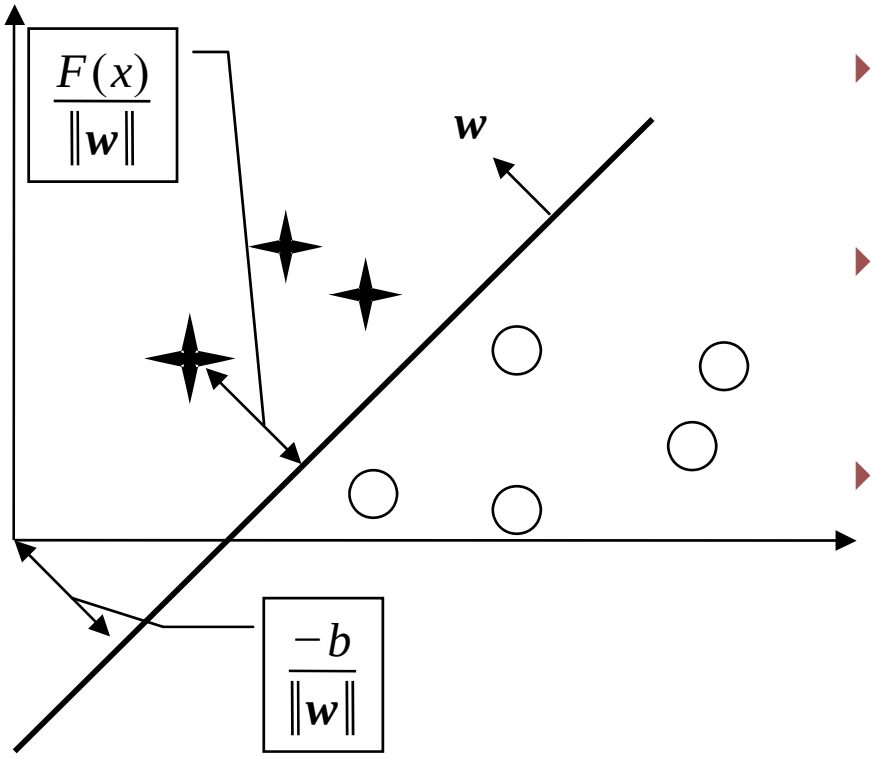
- Les résultats de Mercer servent à prouver les propriétés des fonctions noyaux. En pratique, elles sont peu utiles
- Pour construire des noyaux, on procède par combinaison à partir de noyaux connus
- Si K_1 et K_2 sont des noyaux sur X^2, K_3 défini sur F, les fonctions suivantes sont des noyaux :
  - $K(x,z) = K_1(x,z) + K_2(x,z)$
  - $K(x,z) = K_1(x,z).K_2(x,z)$
  - $K(x,z) = aK_1(x,z)$
  - $K(x,z) = K_3(\phi(x),\phi(z))$
  - …..

Machines à vecteurs supports

Machines à vecteurs support

- Exposé du cours : discrimination 2 classes
- Cas général : discrimination multi-classes, régression, densité
- Idées
  - Projeter -non linéairement- les données dans un espace de "très" grande taille H
  - Faire une séparation linéaire de bonne qualité dans cet espace
  - Raisonner dans H, mais résoudre le problème d'optimisation dans l'espace de départ (noyaux)

- Notion de marge



- Hyperplan H
  - $F(\boldsymbol{x}) = \boldsymbol{w}.\boldsymbol{x} + b = 0$
- Marge géométrique pour x^i
  - $M(\boldsymbol{x}^i) = d^i \left(\frac{\boldsymbol{w}.\boldsymbol{x}^i}{\|\boldsymbol{w}\|} + \frac{b}{\|\boldsymbol{w}\|}\right) = d^i \frac{F(\boldsymbol{x}^i)}{\|\boldsymbol{w}\|}$
- Marge de $\boldsymbol{w}$ par rapport à un ensemble de données *D*
  - $Min_i M(x^i)$
- Hyperplan de marge maximale
  - $Max_w(Min_i M(x^i))$

Marge géométrique vs marge fonctionnelle

- Marge géométrique
  - $d^i \frac{F(\boldsymbol{x}^i)}{\|\boldsymbol{w}\|}$
- Marge fonctionnelle
  - $d^i F(x^i)$
- Remplacer $\boldsymbol{w}$ par $k\boldsymbol{w}$ ne change pas la fonction de décision ou la marge géométrique, mais change la marge fonctionnelle.
- Pour les SVM, on fixera la marge fonctionnelle à 1 et on optimisera la marge géométrique.

Prémisses : Séparation linéaire à hyperplan optimal (1974)

- Hyp : *D* linéairement séparable

- Fonction de décision : $F(x) = w.x + b$ $\quad D=\{x^i,d^i\}_{i=1..N}$ avec $d^i=\pm 1$
- Pb apprentissage :
  - trouver l'hyperplan optimal *H** qui sépare *D* i.e.
    - $d^i.F(x^i) \geq 1, \forall i$

- avec une marge géométrique maximale *M* = $\min_i \frac{d^i.F(x^i)}{\|w\|} = \frac{1}{\|w\|}$

i.e. : Problème Primal :

$$\begin{cases} \text{Minimiser } \|w\|^2 \\ S.C. \quad d^i.F(x^i) \geq 1 \end{cases}$$

- Dans le cas de noyaux linéaires, on résout en général le problème primal
  - Il existe de nombreux algorithmes rapides
- Dans le cas de noyaux non linéaires, on va en général résoudre une version duale du problème
  - On introduit ensuite quelques notions d'optimisation sous contrainte pour décrire la formulation duale du problème

Intermède

Optimisation
Problèmes sous contraintes égalités, inégalités

Optimisation
Problèmes sous contraintes égalités, inégalités

- Soient
  - $f, g_{i,i=1,\dots,k}, h_{j,j=1,\dots,m}$ des fonctions définies sur $\mathbb{R}^n$ à valeur dans $\mathbb{R}$
- On considère le problème primal suivant (pb. (0)) :

$$Min\ f(\boldsymbol{w}), \boldsymbol{w} \in \Omega \subset \mathbb{R}^n$$

Sous contraintes

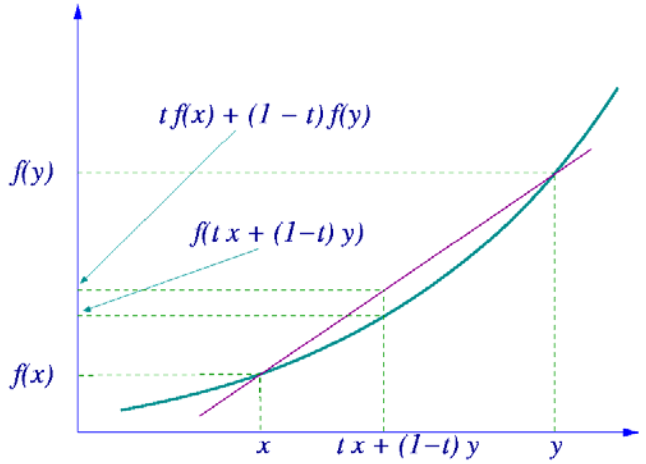
$g_i(\boldsymbol{w}) \leq 0, i = 1, \dots, k$ noté $\boldsymbol{g}(\boldsymbol{w}) \leq 0$

$h_j(\boldsymbol{w}) = 0, j = 1, \dots, m$ noté $\boldsymbol{h}(\boldsymbol{w}) = 0$

  - **Fonction objectif** $f(\boldsymbol{w})$
  - **Région admissible** $R = \{\boldsymbol{w} \in \Omega: \boldsymbol{g}(\boldsymbol{w}) \leq 0, \boldsymbol{h}(\boldsymbol{w}) = 0\}$, région de Ω où f est définie et les contraintes vérifiées
  - $\boldsymbol{w}^*$ est un **minimum global** si il n'existe pas d'autre point $\boldsymbol{w}$ tel que $f(\boldsymbol{w}) < f(\boldsymbol{w}^*)$, c'est un optimum local si $\exists \epsilon > 0: f(\boldsymbol{w}) \geq f(\boldsymbol{w}^*)$, sur la boule $\|\boldsymbol{w} - \boldsymbol{w}^*\| < \epsilon$
  - Une contrainte $g_i(w) \leq 0$ est dite **active** si la solution $\boldsymbol{w}^*$ vérifie $\boldsymbol{g}(\boldsymbol{w}^*) = 0$ et inactive sinon
  - La valeur optimale de la fonction objectif ($f(\boldsymbol{w})$ solution du pb. (0)) est appelée la valeur du problème d'optimisation primal)

Optimisation - Fonctions convexes

- $f(w)$ est convexe pour $\boldsymbol{w} \in \mathbb{R}^n$si
- $\forall t \in [0,1], \forall \boldsymbol{w}, \boldsymbol{v} \in \mathbb{R}^n, \forall \mathrm{t} \in [0,1], \quad f(t\boldsymbol{w} + (1-t)\boldsymbol{v}) \leq tf(\boldsymbol{w}) + (1-t)f(\boldsymbol{v})$



- Un ensemble $\Omega \subset \mathbb{R}^n$est convexe si $\forall \boldsymbol{w}, \boldsymbol{v} \in \mathbb{R}^n, \forall \mathrm{t} \in [0,1], t\boldsymbol{w} + (1-t)\boldsymbol{v} \in \Omega$
- Si une fonction est convexe, tout minimum local est un minimum global
- Un problème d'optimisation pour lequel Ω est convexe, la fonction objectif et les contraintes sont convexes est dit convexe

Optimisation non contrainte

- Th. Fermat
  - Une Condition Nécessaire pour que w* soit un min. de $f(\boldsymbol{w})$, $f \in C^1$ est $\frac{\partial f(\boldsymbol{w}^*)}{\partial w} = 0$
  - Si f est convexe c'est une Condition Suffisante

Optimisation avec contraintes égalités
Lagrangien

- Optimisation avec contraintes égalité (pb (1)):

$$Min\ f(\boldsymbol{w}), \boldsymbol{w} \in \Omega \subset \mathbb{R}^n$$

Sous les contraintes

$$h_j(\boldsymbol{w}) = 0, \quad j = 1, \ldots, m \qquad \text{noté } \boldsymbol{h}(\boldsymbol{w}) = 0$$

- On définit le Lagrangien $L(\boldsymbol{w}, \boldsymbol{\beta})$ associé à ce problème par

$$L(\boldsymbol{w}, \boldsymbol{\beta}) = f(\boldsymbol{w}) + \sum_{j=1}^{m} \beta_j h_j(\boldsymbol{w})$$

  - les β_j sont les coefficients de Lagrange
- Rq.
  - Si $\boldsymbol{w}^*$ est une solution du problème d'optimisation sous contrainte, Il est possible que $\frac{\partial f(\boldsymbol{w}^*)}{\partial w} \neq 0$

Optimisation avec contraintes égalités
Th. Lagrange

- Th. Lagrange
  - Une condition necessaire pour que $\boldsymbol{w}^*$, soit solution de (pb. (1)), avec $f, h_i \in C^1$ est
    - $\frac{\partial L(\boldsymbol{w}^*, \boldsymbol{\beta}^*)}{\partial \boldsymbol{w}} = 0$
    - $\frac{\partial L(\boldsymbol{w}^*, \boldsymbol{\beta}^*)}{\partial \boldsymbol{\beta}} = 0$
  - Si $L(\boldsymbol{w}, \boldsymbol{\beta}^*)$ est une fonction convexe de $\boldsymbol{w}$, c'est une condition suffisante
  - Rq
    - La première condition donne un nouveau système d'équations
    - La seconde donne les contraintes

Optimisation sous contraintes égalité + inégalités - Lagrangien augmenté

- De même, on définit le Lagrangien augmenté pour le pb. (0) :

$$L(\boldsymbol{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = f(\boldsymbol{w}) + \sum_{i=1}^{k} \alpha_i g_i(\boldsymbol{w}) + \sum_{j=1}^{m} \beta_j h_j(\boldsymbol{w})$$

- Rappel problème primal : pb. (0)

$Min\ f(\boldsymbol{w}), \boldsymbol{w} \in \Omega \subset \mathbb{R}^n$

Sous contraintes

$g_i(\boldsymbol{w}) \leq 0, i = 1, \ldots, \text{k}$ noté $\boldsymbol{g}(\boldsymbol{w}) \leq 0$

$h_j(\boldsymbol{w}) = 0, j = 1, \ldots, \text{m}$ noté $\boldsymbol{h}(\boldsymbol{w}) = 0$

Optimisation sous contraintes égalité + inégalités - Lagrangien augmenté

- Reformulation du problème primal
  - Soit $\theta_P(\boldsymbol{w}) = f(\boldsymbol{w}) + \max_{\boldsymbol{\alpha},\boldsymbol{\beta};\boldsymbol{\alpha}\geq 0} (\alpha.g(\boldsymbol{w}) + \beta.h(\boldsymbol{w}))$
  - $\theta_P(\boldsymbol{w}) = \begin{cases} f(\boldsymbol{w}) \text{ si } \boldsymbol{w} \text{ est admissible} \\ +\infty \text{ sinon} \end{cases}$
    - car $\max_{\boldsymbol{\alpha},\boldsymbol{\beta};\boldsymbol{\alpha}\geq 0} \left(\alpha g(w) + \beta h(w)\right) = 0$ pour un point admissible i.e. qui satisfait les contraintes primales
  - $\theta_P(\boldsymbol{w}) = f(\boldsymbol{w})$ pour $\boldsymbol{w}$ admissible
    - $\theta_P(\boldsymbol{w})$ prend la même valmeur que la fonction objectif pour tout $\boldsymbol{w}$ admissible
  - Rq: on retrouve le problème primal original :
    - $\min_{\mathbf{w}\in\Omega} \theta_P(\boldsymbol{w}) = \min_{\mathbf{w}\in\Omega} \max_{\boldsymbol{\alpha},\boldsymbol{\beta};\boldsymbol{\alpha}\geq 0} L(\boldsymbol{w},\boldsymbol{\alpha},\boldsymbol{\beta})$ donne la même solution que le problème original (pb 0)

Optimisation – formulation duale

- Formulation duale du problème d'optimisation
  - Le problème d'optimisation dual correspondant au problème primal pb (0) est :

$$Maximiser_{\alpha,\beta} \ \theta(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \min_{w \in \Omega} L(\boldsymbol{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

$$\text{sous contrainte } \boldsymbol{\alpha} \geq 0$$

    - Max $\theta(\boldsymbol{\alpha}, \boldsymbol{\beta})$ est appelé la valeur du dual
    - Rq : $\min_{w \in \Omega} L(\boldsymbol{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ est une fonction de $\boldsymbol{\alpha}, \boldsymbol{\beta}$ uniquement
  - Propriété (dualité faible) : la valeur du dual est bornée supérieurement par la valeur du primal

$$\max_{\boldsymbol{\alpha},\boldsymbol{\beta};\boldsymbol{\alpha} \geq 0} \ \min_{w \in \Omega} L(\boldsymbol{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \leq \min_{w \in \Omega} \ \max_{\boldsymbol{\alpha},\boldsymbol{\beta};\boldsymbol{\alpha} \geq 0} \ L(\boldsymbol{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

Dans certains cas, on a égalité, cf. dualité forte

Optimisation : dualité faible

- $\max_{\boldsymbol{\alpha},\boldsymbol{\beta};\boldsymbol{\alpha}\geq 0} \min_{\boldsymbol{v}\in\Omega} L(\boldsymbol{v},\boldsymbol{\alpha},\boldsymbol{\beta}) \leq \min_{\mathbf{w}\in\Omega} \max_{\boldsymbol{\alpha},\boldsymbol{\beta};\boldsymbol{\alpha}\geq 0} L(\boldsymbol{w},\boldsymbol{\alpha},\boldsymbol{\beta})$:
  - $\theta_D(\alpha,\beta) = \min_{v\in\Omega} L(\boldsymbol{v},\boldsymbol{\alpha},\boldsymbol{\beta})$:
    - $\theta_D(\alpha,\beta) \leq L(\boldsymbol{w},\boldsymbol{\alpha},\boldsymbol{\beta})$ pour un $\boldsymbol{w}$ qque admissible
    - $L(\boldsymbol{w},\boldsymbol{\alpha},\boldsymbol{\beta}) = f(w) + \alpha.g(w) + \beta.h(w)$ avec $\alpha \geq 0, g(\boldsymbol{w}) \leq 0, h(\boldsymbol{w}) = 0$
    - $L(\boldsymbol{w},\boldsymbol{\alpha},\boldsymbol{\beta}) \leq f(w)$
    - $\theta_D(\alpha,\beta) \leq f(w)$
    - $\max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\alpha}\geq 0} \min_{v\in\Omega} L(v,\boldsymbol{\alpha},\boldsymbol{\beta}) \leq f(w)$ pour un w qque admissible
  - $\theta_P(\boldsymbol{w}) = \max_{\boldsymbol{\alpha},\boldsymbol{\beta};\boldsymbol{\alpha}\geq 0} L(\boldsymbol{w},\boldsymbol{\alpha},\boldsymbol{\beta})$
    - $\theta_P(\boldsymbol{w}) = f(\boldsymbol{w}) + \max_{\boldsymbol{\alpha},\boldsymbol{\beta};\boldsymbol{\alpha}\geq 0} (\alpha.g(\boldsymbol{w}) + \beta.h(\boldsymbol{w}))$
    - $\theta_P(\boldsymbol{w}) = \begin{cases} f(\boldsymbol{w}) \text{ si } \boldsymbol{w} \text{ est admissible} \\ +\infty \text{ sinon} \end{cases}$ car $\max_{\boldsymbol{\alpha},\boldsymbol{\beta};\boldsymbol{\alpha}\geq 0} \left(\alpha g(w) + \beta h(w)\right) = 0$ pour un point admissible
    - $\theta_P(\boldsymbol{w}) = f(\boldsymbol{w})$ pour $\boldsymbol{w}$ admissible
- D'où l'inégalité de dualité faible

- Théorème de dualité forte
  - Etant donné un problème d'optimisation

    $Min\ f(\boldsymbol{w}), \boldsymbol{w} \in \Omega \subset \mathbb{R}^n$ convexe et $f \in C^1$ convexe

    Sous contraintes

    $g_i(\boldsymbol{w}) \leq 0, i = 1, \ldots,$k noté $\boldsymbol{g}(\boldsymbol{w}) \leq 0$

    $h_j(\boldsymbol{w}) = 0, j = 1, \ldots,$m noté $\boldsymbol{h}(\boldsymbol{w}) = 0$

    où les g_i et les h_j sont affines $(h_j(\boldsymbol{w}) = A_{\boldsymbol{j}}\boldsymbol{w} + b_j)$

    - alors les valeurs du primal et du dual sont égales
    - Les conditions d'existence d'un optimum sont données par le théorème de Kuhn et Tucker

Optimisation
Th. Kuhn et Tucker

- On considère (pb. (0)) avec Ω convexe et $f \in C_1$ convexe, g_i, h_j affines $(h = A.w + b)$
- Une CNS pour que w* soit un optimum est qu'il existe α* et β* :

$$\begin{cases} \dfrac{\partial L(w^*, \alpha^*, \beta^*)}{\partial w} = 0 \\ \dfrac{\partial L(w^*, \alpha^*, \beta^*)}{\partial \beta} = 0 \\ \alpha_i^* \, g_i(w^*) = 0, i = 1..k \\ g_i(w^*) \le 0, i = 1..k \\ \alpha_i^* \ge 0, i = 1..k \end{cases}$$

- Sous les hypothèses de convexité, la formulation duale est une alternative à la formulation primale qui peut se révéler plus simple à traiter (e.g. SVM non linéaires)

Optimisation

- Rq
  - La 3e condition dite condition complémentaire de Karush-Kuhn-Tucker implique que pour une contrainte active $\alpha_i^* \geq 0$ alors que pour une contrainte inactive $\alpha_i^* = 0$
    - Soit une contrainte est active ($\alpha_i^* \geq 0$ et $g_i(w^*) = 0$), w^* est un point frontière de la région admissible
    - Soit elle est inactive ($\alpha_i^* = 0$) et w^* est dans la région admissible
  - Si le point solution w^* est dans la région admissible (contrainte inactive) alors les conditions d'optimalité sont données par le th. de Fermat et $\alpha_i^* = 0$. Si il est sur la frontière (contrainte active), les conditions d'optimalité sont données par le th. de Lagrange avec $\alpha_i^* > 0$.

- Fin de l'intermède

SVM – formulations primale et duale

Cas d'un noyau linéaire

- SVM
  - Ω, f et les contraintes sont convexes, L est quadratique
  - On étudie le cas, $D = \{(x^i, d^i)\}_{i=1...N}$ linéairement séparables, $d^i \in \{-1,1\}$
- Pb. Primal

$$Min(w.w) \quad \text{(i.e. max la marge)}$$

sous les contraintes

$$d^i(w.x^i + b) \geq 1, i = 1..N$$

- Lagrangien primal

$$L(w,b,\alpha) = \frac{1}{2} w.w - \sum_{i=1}^{N} \alpha_i (d^i (w.x^i + b) - 1)$$

- Lagrangien dual

$$L(w,b,\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} d^i d^j \alpha_i \alpha_j (x^i.x^j)$$

- Avec $\alpha_i \geq 0$ dans les 2 cas

SVM – formulations primale et duale

- Pb. Dual

$$\text{Max} \quad L(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} d^i d^j \alpha_i \alpha_j (x^i . x^j)$$

$$\text{sous les contraintes}$$

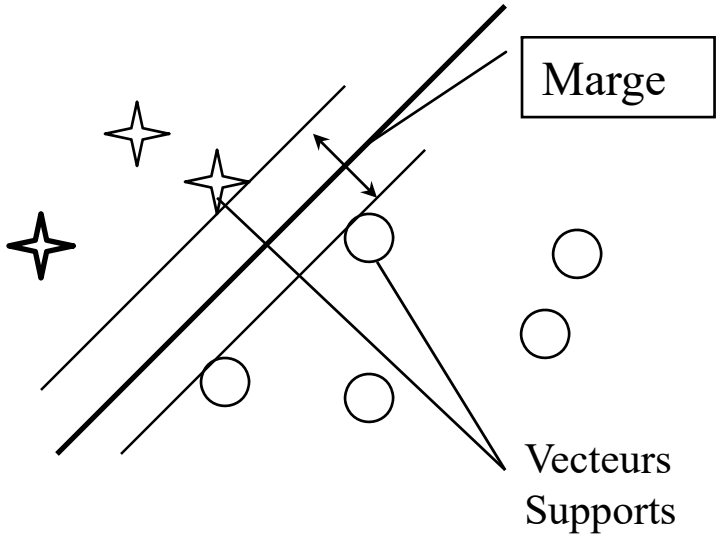
$$\begin{cases} \sum_{i=1}^{N} d^i \alpha_i = 0 \\ \alpha_i \geq 0, i = 1..N \end{cases}$$

- C'est un problème d'optimisation quadratique sous contrainte

- Solution : $\boldsymbol{w}^*$ dépend uniquement des points supports, i.e. des points sur la marge qui vérifient : $d^i F^*\left(x^i\right) = 1$
- la fonction de décision prend la forme

$$F(x, \alpha^*, \beta^*) = \sum_{i\,vecteur\,\mathrm{sup}\,port} d^i \alpha^*_i \, (x^i . x) + b^*$$

  - Rq: Quelque soit la dimension de l'espace, le nombre de degrés de liberté est "égal" au nombre de points de support
  - *F** dépend uniquement du produit scalaire $x^i . x$



Machines à vecteurs supports
cas de noyaux non linéaires

- Faire une séparation à marge max. dans un espace défini par une fonction noyau.
- Tous les résultats sur le classifieur linéaire à marge max. se transposent en remplaçant $x^i.x$ par $K(x^i, x)$

$$\Phi : R^n \rightarrow R^p$$

$$W = \sum_{x^i \, V.S.} d^i \alpha_i \Phi(x^i) \qquad F(x) = \sum_{x^i \, V.S.} d^i \alpha_i \Phi(x^i) \Phi(x) + b$$

$$\Phi(x).\Phi(x') = K(x, x')$$

$$F(x) = \sum_{x^i \, V.S.} d^i \alpha_i K(x, x^i) + b$$

Machines à vecteurs supports

- Apprentissage :
  - On résout le problème d'optimisation dual :

$$\text{Maximiser} \quad L(\alpha) = \sum_i \alpha_i - \sum_{i,j} \alpha_i \alpha_j d^i d^j K(x^i, x^j)$$

$$S.C \quad \alpha_i \geq 0 \quad \text{et} \quad \sum_i \alpha_i d^i = 0$$

  - Problème minimisation quadratique sous contraintes dans l 'espace de départ
  - Difficile en pratique : différents algorithmes.
  - Dans la solution optimale $\alpha_i > 0$ uniquement pour les points support.
    - Seuls les produits scalaires *K* apparaissent, et pas les Φ.

Propriétés de généralisation -exemples

- Th 1 $$E\,[P\,(erreur(x\,))] \le \frac{E\,[\#\,\text{vecteurs supports}]}{\#\,\text{exemples apprentissage} - 1}$$
  - peu de points support → meilleure généralisation
  - indépendant de la taille de l'espace de départ
- Th 2
  - Si $$\exists\, q/\forall i \;= 1..N\;,\; \left\| x^{\,i} \right\| \le q$$
  - l'hyperplan optimal passe par l'origine et a pour marge ρ
  - Alors $$E\,[P\,(erreur(x\,))] \le \frac{E\,[\frac{q}{2}^{2}]}{\rho^{2}}$$

    - Dans les 2 cas, E[P()] est l'espérance sur tous les ensembles de taille l-1, et E[membre droit] est l'espérance sur tous les ensembles d'apprentissage de taille l (leave one out).

Machines à vecteurs supports
Cas non linéairement séparable

- Marges molles
  - L'algorithme est instable
    - Dans les cas non linéairement séparables
    - Dans le cas de données réelles même linéairement séparables
    - **Solution adoptée en pratique**
      - autoriser des erreurs, i.e. prendre pour contraintes :

$$d^i (W \cdot \Phi(x^i) + b) \geq 1 - \eta^i$$

$$\eta^i \geq 0$$

- $\eta^i = 0, x^i$ est correctement classifié et est du bon coté de la marge
- $0 < \eta^i <= 1, x^i$ est correctement classifié, est à l'intérieur de la marge
- $\eta^i > 1, x^i$ est mal classé
- η^i : *slack variable*

Machines à vecteurs supports
Cas non linéairement séparable

- But
  - Maximiser la marge tout en pénalisant les points qui sont mal classés
- Formalisation
  - Plusieurs expressions possibles du problème
  - L'une des plus courantes :

$$Min(w.w) + C\sum_{i=1}^{N} \eta^{i} \qquad \text{(i.e. max la marge)}$$

$$S.C.$$

$$d^{i}(w.x^{i} + b) \geq 1 - \eta^{i}, i = 1..N$$

$$\eta^{i} \geq 0, i = 1..N$$

  - C fixé par validation croisée joue le rôle de paramètre de régularisation

Machines à vecteurs supports
Cas non linéairement séparable

- Marges molles – formulation duale

$$\text{Maximiser} \qquad L(\alpha) = \sum_i \alpha_i - \sum_{i,j} \alpha_i \alpha_j d^i d^j K(x^i, x^j)$$

$$S.C \quad 0 \le \alpha_i \le C \quad \text{et} \quad \sum_i \alpha_i d^i = 0$$

Algorithmes d'optimisation

- **Algorithmes d'optimisation standard pour la programmation quadratique sous contrainte**
  - **e.g. Sequential Minimal Optimization (SMO)**
- **Algorithmes stochastiques - SVM Results –(Bottou 2007)**
  - Task : Document classification - RCV1 documents belonging to the class CCAT (2 classes classification task)
    - Programs SVMLight and SVMPerf are well known SVM solvers written by Thorsten Joachims. SVMLight is suitable for SVMs with arbitrary kernels. Similar results could be achieved using Chih-Jen Lin's LibSVM software. SVMPerf is a specialized solver for linear SVMs. It is considered to be one of the most efficient optimizer for this particular problem.

Algorithm (hinge loss)	Training Time	Primal cost	Test Error
SVMLight	23642 secs	0.2275	6.02%
SVMPerf	66 secs	0.2278	6.03%
Stochastic Gradient (svmsgd)	**1.4 secs**	0.2275	6.02%
Stochastic Gradient (svmsgd2	**1.4 secs**	0.2275	6.01%

Gaussian process regression

Motivations

- Most ML algorithm for regression predict a mean value
- Gaussian processes are Bayesian methods that allow us to predict, not only a mean value, but a distribution over the output values
  - In regression, for each input value x, the predicted distribution is Gaussian and is then fully characterized by its mean and variance

Gaussian distributions refresher

- Multivariate Gaussian distribution $x \sim \mathcal{N}(\mu, \Sigma), x \in R^n$
  - $p(x) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{1/2}} \exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu))$
- Summation (a)
  - Let x and y two random variables with the same dimensionality, $p(x) = \mathcal{N}(\mu_x, \Sigma_x)$ and $p(y) = \mathcal{N}(\mu_y, \Sigma_y)$
  - Then their sum is also Gaussian: $\mathrm{p(x+y)} = \mathcal{N}(\mu_x + \mu_y\ , \Sigma_x + \Sigma_y)$
- Marginalization (b)
  - Let $x, p(x) = \mathcal{N}(\mu\ , \Sigma)$, consider a partition of x into two sets of variables $x = \begin{pmatrix} x_a \\ x_b \end{pmatrix}$.
  - Let us denote $\mu = \begin{pmatrix} \mu_a \\ \mu_b \end{pmatrix}, \Sigma = \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix}$
  - Then the marginals are also Gaussians: $p(x_a) = \int_{x_b} p(x_a, x_b; \mu, \Sigma) = \mathcal{N}(\mu_a\ , \Sigma_{aa})$,
  - Σ being symmetric, $\Sigma_{ab} = \Sigma_{ba}$
- Conditioning (c)
  - The conditionals are also Gaussians
  - $p(x_a|x_b) = \mathcal{N}(\mu_{a|b}\ , \Sigma_{a|b})$ with $\mu_{a|b} = \mu_a + \Sigma_{ab}\Sigma_{bb}^{-1}(x_b - \mu_b)$ and $\Sigma_{a|b} = \Sigma_{aa} - \Sigma_{ab}\Sigma_{bb}^{-1}\Sigma_{ba}$
- Marginalization bis (d)
  - Let x and y two random vectors such that $p(x) = \mathcal{N}(\mu, \Sigma_x)$ and $p(y|x) = \mathcal{N}(Ax + b, \Sigma_y)$
  - The marginal of y is $p(y) = \int\ \ p(y|x)p(x)dx = \mathcal{N}(A\mu + b, \Sigma_y + A\Sigma_x A^T)$

Introducing the Gaussian processes
From Bayesian linear regression to Gaussian processes

- Consider the linear parameter model:
  - $y(\boldsymbol{x}) = \boldsymbol{w}^T \phi(\boldsymbol{x})$
  - where $\boldsymbol{w} \in R^M, \phi(\boldsymbol{x}) \in R^M$ are M fixed basis functions
    - For example, ϕ could be a linear function $\phi(\boldsymbol{x}) = (\boldsymbol{x}, 1)$ or ϕ could be a vector of gaussian kernels $\phi_i(x) = \exp(-\frac{(x-\mu_i)^2}{2s^2})$
- We consider a Bayesian setting
  - With $\boldsymbol{w}$ following a prior distribution given by an isotropic Gaussian
    - $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{0}, \alpha^{-1} I)$
      - α^{-1} is the precision parameter = the inverse variance
  - For any value of $\boldsymbol{w}$, $y(\boldsymbol{x}) = \boldsymbol{w}^T \phi(\boldsymbol{x})$ defines a specific function of $\boldsymbol{x}$
  - $p(\boldsymbol{w})$ thus defines a distribution over functions $y(\boldsymbol{x})$

Introducing the Gaussian processes
From Bayesian linear regression to Gaussian processes

- How to characterize the distribution over functions $y(\boldsymbol{x})$?
- In practice, we will want to evaluate $y(\boldsymbol{x})$ at specific values x
  - e.g. at the training points or for a test point
- Let us consider a finite data sample $x^1, \dots, x^N$
- Let us denote $\boldsymbol{y} = (y^1, \dots, y^N)^T$, with $y^i = y(x^i)$
- We want to characterize the distribution of $\boldsymbol{y}$
  - $\boldsymbol{y} = \boldsymbol{\Phi}\mathbf{w}$, with $\boldsymbol{\Phi} = \left[\boldsymbol{\phi}(\boldsymbol{x}^1), \dots, \boldsymbol{\phi}(\boldsymbol{x}^N)\right]^T$ called the design matrix $\Phi_{ij} = \phi_j(x^i)$
  - $\mathbf{w}$ is Mx1, $\boldsymbol{\Phi}$ is NxM, $\boldsymbol{y}$ is Nx1
  - $\boldsymbol{y}$ being a linear combination of Gaussian variables (the elements of $\mathbf{w}$) is itself Gaussian and fully characterized by its mean and variance
    - $E[y] = \boldsymbol{\Phi}\mathrm{E}[\mathbf{w}] = 0$
    - $Cov[\boldsymbol{y}] = E[\boldsymbol{y}\boldsymbol{y}^T] = \boldsymbol{\Phi}\mathrm{E}[w\boldsymbol{w}^{\mathrm{T}}]\boldsymbol{\Phi}^{\mathrm{T}} = \frac{1}{\alpha}\boldsymbol{\Phi}\boldsymbol{\Phi}^{\mathrm{T}} = \mathbf{K}$
    - $\mathbf{K}$ is a Gram matrix with elements $K_{nm} = k(x^n, x^m) = \frac{1}{\alpha}\phi(x^n)^T\phi(x^m)$
      - $k(x, x')$ is the kernel function

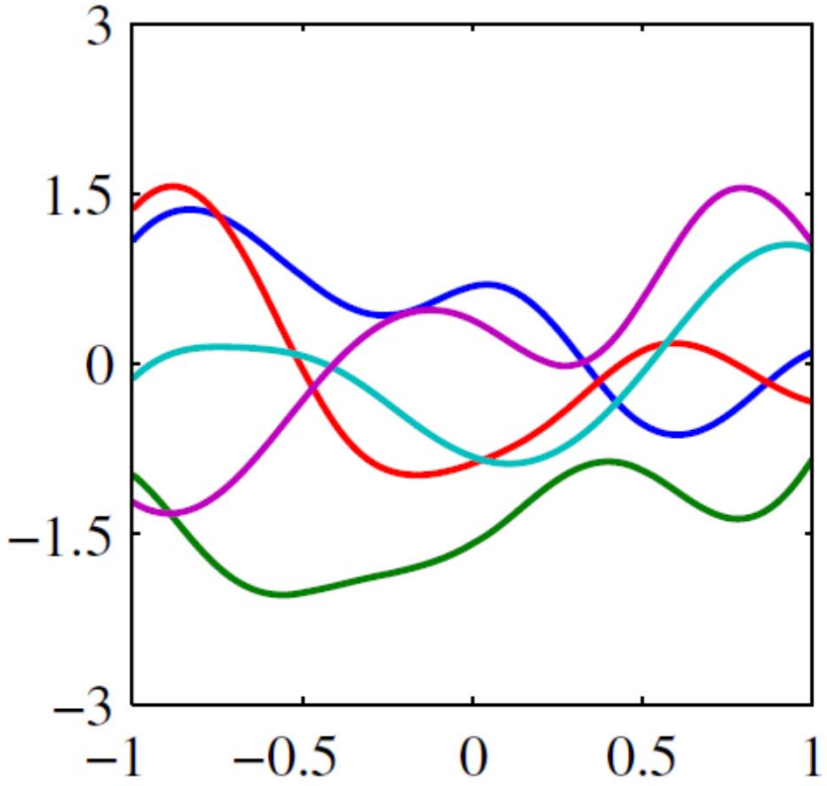
$$\boldsymbol{y} = \mathcal{N}(\mathbf{0}, \boldsymbol{K})$$

- This is a first example of Gaussian process, defined by a linear model
- Usually, the kernel function is not defined through basis functions, but directly by specifying a Kernel function, e.g. a Gaussian kernel

Introducing the Gaussian processes
From Bayesian linear regression to Gaussian processes

- Samples of functions drawn from Gaussian processes for a « Gaussian Kernel »
- $k(x, x') = \exp(-\frac{\|x-x'\|^2}{2\sigma^2})$
- We specify a set of input points $x = (x^1, \ldots, x^N)$ in $[-1,1]$ and an NxN covariance matrix K.
- We draw a vector $(y^1, \ldots, y^N)$ from the Gaussian defined by $\boldsymbol{y} = \mathcal{N}(\boldsymbol{0}, \boldsymbol{K})$

- Bishop C. PRML



Gaussian processes

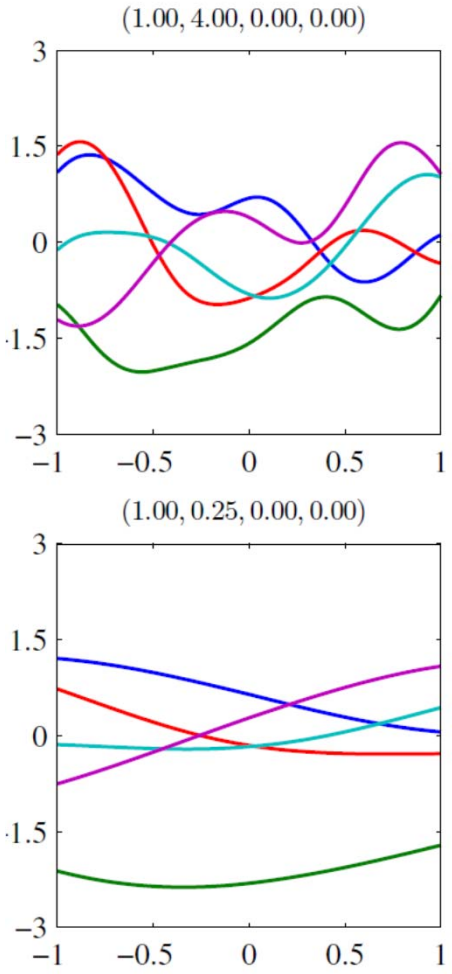
- Definition
  - A stochastic process is a collection of random variables $\{f(x); x \in \mathcal{X}\}$ indexed by elements of set $\mathcal{X}$ (in the following one will consider $\mathcal{X} = R$).
    - This is a probability distribution over the functions $f(x)$
  - A Gaussian process is a stochastic process such that the set of values of $f(x)$ evaluated at any number of points $x^1, \ldots x^N$ is jointly Gaussian, i.e.:
    - $$\begin{bmatrix} f(x^1) \\ \vdots \\ f(x^N) \end{bmatrix} \sim N\left(\begin{bmatrix} m(x^1) \\ \vdots \\ m(x^N) \end{bmatrix}, \begin{bmatrix} k(x^1, x^1) \ldots k(x^1, x^m) \\ \vdots \quad \ddots \quad \vdots \\ k(x^m, x^1) \ldots k(x^m, x^m) \end{bmatrix} \right)$$
- Properties
  - A Gaussian process is entirely specified by its
    - Mean function $\mathrm{m}(\mathrm{x}) = E[f(x)]$
    - Covariance function $k(x, x') = E[(f(x) - m(x))(f(x') - m(x')]$
  - One denotes $\mathrm{f} \sim GP(m, k)$ meaning that f is distributed as a GP with mean m and covariance k functions

Gaussian processes

- Intuition
  - Gaussian distributions model finite collections of real valued variables
  - Gaussian processes extend multivariate gaussians to infinite collections of real-valued vairables
    - GP are distributions over random functions
    - Let H be a class of functions $f: X \to Y$. A random function $f(.)$ from H is a function which is randomly drawn from H
    - Intuitively, one can think of $f(.)$ as an infinite vector drawn from an infinite multivariate Gaussian. Each dimension of the Gaussian corresponds to an element x from the index and the corresponding compoinent of the random vector is the value $f(x)$
- What could be the functions $m(.)$ and $k(.,.)$?
  - Any real valued function $m(.)$ is acceptable
  - K should be a valid covariance matrix corresponding to a Gaussian distribution
    - This is the case if K is positive semi-definite (remember Mercer conditions for kernels)
      - Any valid kernel can be used as a covariance function

Gaussian processes

- Example
  - Zero mean Gaussian process $GP(0, k(.,.))$ defined for functions $h: X \rightarrow R$
  - $k(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\frac{\theta_1}{2}\left|\left|\boldsymbol{x} - \boldsymbol{x}'\right|\right|^2)$
  - The function values are distributed around 0
  - $f(x)$ and $f(x')$ will have a high covariance $k(x, x')$ if x and x' are nearby and a low covariance otherwise
    - i.e. they are locally smooth



Bishop PRML, Top $\theta_1 = 4$, bottom $\theta_1 = 0.25$

Gaussian processes for regression

- We consider a Gaussian process regression model (1 dimensional for simplification)
  - $y = f(x) + \epsilon$, with $x \in R^n$ and $y \in R$
    - $\epsilon \sim \mathcal{N}(0, \sigma^2)$ independently chosen for each observation accounts for the noise at each observation
  - Let us consider a set of training examples $S = \{(x^1, y^1), \ldots, (x^N, y^N)\}$ from an unknown distribution
  - Let us denote $Y = (y^1, \ldots, y^N)^T$ and $F = (f^1, \ldots, f^N)^T$ with $f^i = f(x^i)$
  - From the definition of a Gaussian process, one assume a prior distribution over functions $f(.)$. We assume a zero mean Gaussian process prior:
    - $p(F) = \mathcal{N}(0, K)$ with K a Gram matrix defined by a kernel function $K_{ij} = k(x_i, x_j)$
- We will
  - Characterize the joint distribution of $Y = (y^1, \ldots, y^N)^T$
  - In order to define the predictive distribution for test points $p(y_{N+1}|Y)$

Gaussian processes for regression
Characterizing the joint distribution of $Y = (y^1, \dots, y^N)^T$

- The joint distribution of $Y = (y^1, \dots, y^N)^T$ is
  - $p(Y) = \int p(Y|F)p(F)df = \mathcal{N}(O, C)$
  - With the covariance matrix C defined as $C(x^i, x^j) = k(x^i, x^j) + \frac{1}{\sigma^2}\delta_{ij}$
    - δ_{ij} is the Kronecker symbol
- Demonstration
  - We will first show $p(Y|F) = \mathcal{N}(F, \frac{1}{\sigma^2} I_N)$
    - $p(Y|F) = p(y^1, \dots, y^N|F)$
    - $p(Y|F) = \prod_{i=1}^N p(y^i | f^i)$
    - $p(Y|F) = \prod_{i=1}^N \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2\sigma^2}(y^i - f^i)^2)$
    - $p(Y|F) = \left(\frac{1}{2\pi\sigma^2}\right)^{N/2} \exp(-\frac{1}{2\sigma^2}\|Y - F\|^2)$
    - $p(Y|F) = \mathcal{N}(F, \frac{1}{\sigma^2} I_N)$

Gaussian processes for regression
Characterizing the joint distribution of $Y = (y^1, \dots, y^N)^T$

- Demonstration of $p(Y) = \int p(Y|F)p(F)df = \mathcal{N}(O, C)$

  - $p(F) = \mathcal{N}(0, K)$
  - $p(Y|F) = \mathcal{N}(F, \frac{1}{\sigma^2} I_N)$
  - $p(Y) = \int p(Y|F)p(F)df$
  - By property (d) in Gaussian refresher we get:
  - $p(Y) = \mathcal{N}\left(O, \frac{1}{\sigma^2} I_N + K\right) = \mathcal{N}(O, C)$
    - With $C_{ij} = k(x^i, x^j) + \frac{1}{\sigma^2}\delta_{ij}$

Gaussian processes for regression
Predictive distribution

- For the regression, our goal is to predict the value y for a new observation x
  - Let us consider a training set $\mathrm{D} = \{(\mathrm{x}^{\mathrm{i}}, \mathrm{y}^{\mathrm{i}}); \mathrm{i} = 1 \ldots \mathrm{N}\}$, and denote $Y^N = (y^1, \ldots, y^N)^T$, let y^{N+1} the value one wants to predict for observation x^{N+1}, $Y^{N+1} = (Y^N, y^{N+1})^T$
- Let us first explicit the joint distribution over Y^{N+1}
  - $p(T^{N+1}) = \mathcal{N}(0, C_{N+1})$ with $C_{N+1} = \begin{pmatrix} C_N & k \\ k^T & c \end{pmatrix}$
    - C_N the covariance matrix of Y_N
    - $k \in R^N$ $k_i = k(x^i, x^{N+1}); i = 1 \ldots N$
    - $c = \mathrm{k}(\mathrm{x}^{\mathrm{N+1}}, \mathrm{x}^{\mathrm{N+1}}) + \sigma^2 \in R$
  - Proof
    - This is a direct application of the result shown before $p(Y) = \mathcal{N}(O, C)$

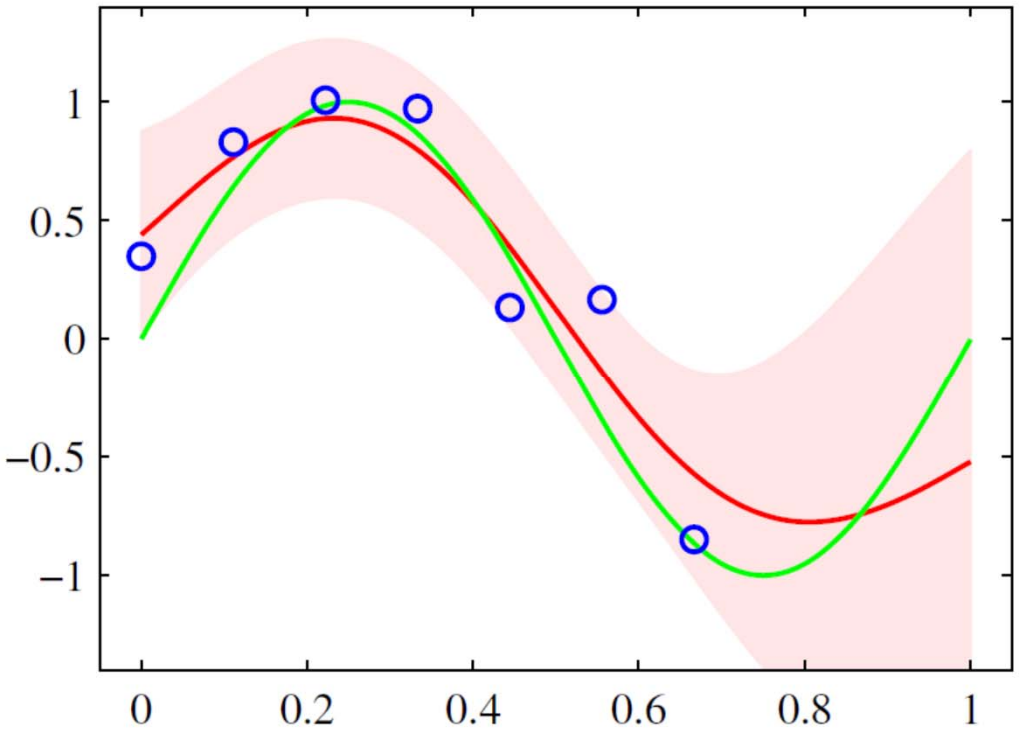
Gaussian processes for regression
Predictive distribution

- Prediction is achieved via the conditional distribution $p(y_{N+1}|Y)$
  - By definition of a Gaussian process, $p(y_{N+1}|Y, X)$ is a Gaussian.
  - Its mean and covariance are given by:
    - $m(x_{N+1}) = k^T C_N^{-1} Y$
    - $\sigma^2(x_{N+1}) = c - k^T C_N^{-1} k$
  - Proof
    - This is a direct application of property (c) (conditioning)
- Property
  - $m(x_{N+1})$ writes as $m(x_{N+1}) = \sum_{i=1}^{N} a_i k(x_i, x_{N+1})$
    - With $a_i$the i^{th} component of $C_N^{-1} Y$

Gaussian processes for regression
Predictive distribution

- This means that for any new datum x^{N+1}, one can compute
  - A mean prediction $m(x_{N+1})$
  - An uncertainty associated to this prediction $\sigma^2(x_{N+1})$

Illustration of Gaussian process regression applied to the sinusoidal data set in Figure A.6 in which the three right-most data points have been omitted. The green curve shows the sinusoidal function from which the data points, shown in blue, are obtained by sampling and addition of Gaussian noise. The red line shows the mean of the Gaussian process predictive distribution, and the shaded region corresponds to plus and minus two standard deviations. Notice how the uncertainty increases in the region to the right of the data points.



Bishop C. PRML

Learning hyperparameters

- The kernel functions can be chosen a priori
- Alternatively, they may be defined as parametric functions (e.g. squared exponential kernel as in the example and the parameters may be learned e.e. by maximum likelihood
  - Log likelihood for a Gaussian process regression model
  - $\log p(Y|\theta) = -\frac{1}{2}\log|C_N| - \frac{1}{2}Y^T C_N^{-1} Y - \frac{N}{2}\log(2\pi)$
  - Training can be performed using gradient descent on the parameters θ