Kernel Characterization

A function $K: X \times X \rightarrow \mathbb{R}$, which is either continuous or has a countable domain, can be decomposed $K(x,z) = \langle \phi(x), \phi(y) \rangle$ into a feature map $\phi$ into a Hilbert space $F$ applied to both its arguments followed by the evaluation of the inner product in $F$ if and only if it satisfies the finitely positive semi-definite property. (Shawe-Taylor, Cristianini 2004)
Computing Kernels

Input space

Feature space

Direct

Kernels

K(x, y) = \langle \phi(x), \phi(y) \rangle
Constructing Hilbert spaces from Kernels

- Construct a linear vector space with inner product from kernels. Let $f(x)$, $g(x)$ be defined by
  $f(x) = \sum_i \alpha(i)K(x(i),x)$ and $g(x) = \sum_j \beta(j)K(z(j),x)$

Define inner product by
  $\langle f, g \rangle = \sum_{i,j} \alpha(i)\beta(j) K(x(i),z(j))$. Can easily show that this is a valid inner product.

- Completeness and separability
- Reproducing property: (RKHS)
  $\langle f, K(x, \cdot) \rangle = \sum \alpha(i)K(x(i),x) = f(x)$
Consider changing SVM to LS SVM by making following modifications:

$$
\min_{(w,e)} \frac{1}{2} \|w\|^2 + \frac{1}{2} C \sum e(i)^2
$$

subject to $d(i) - (w^T \Phi(x(i)) + b) = e(i), \forall i$, and $C > 0$. Note that $e(i)$ is error term.

Key differences with between SVM and LS SVM:

- $\varepsilon$ - insensitive cost replaced by quadratic error cost.
- Inequality constraint replaced by equality constraint.
Dual Solution to LS SVM

Let $\alpha$ be vector of Lagrange multipliers and $d$ be vector of outputs then solution has following form:

$$
\begin{pmatrix}
0 & 1^T \\
1 & K+I/C
\end{pmatrix}
\begin{pmatrix}
b \\
\alpha
\end{pmatrix}
=
\begin{pmatrix}
0 \\
d
\end{pmatrix}
$$

where $K(x,z) = \Phi(x)^T \Phi(z)$ and denotes 1 vector of 1s.

$$
f(x) = \sum \alpha(i) K(x^T x(i)) + b
$$
Comments about LS SVM

- Solution to LS SVM depends on $d$, dimensionality of feature space $\Phi(x)$ in primal space and $m$, number of training samples in dual space.
- Both solutions involve solving a set of linear equations. Work in space that has lower dimension.
- Adaptive on-line solutions can now be implemented.
- Algorithm easily constructed for pattern classification problems.
- In dual space, practically all input training examples are support vectors as Lagrange multipliers, $\alpha$ are proportional to error, $e$. 
LS SVM Solution

- Solution in primal or dual space involves a solution to a set of respectively \((m+1, d+1)\) linear equations.
- Dual space solution: unlike SVM solution all input training examples are support vectors
- Objectives: want good performance with low to moderate computational complexity
  - On-line vs. batch
  - Sparseness: reduced system, subspace method
  - Criteria for choosing SV
  - Numerical stability: matrix computations
On-line Adaptive LS Algorithms

- **Primal Solution** (using linear network): use recursive least squares algorithms.
- **Dual Solution** (e.g. Gaussian kernels): modified windowed recursive least squares algorithms.
- **Matrix Inversion Algorithms**
  - Inverting block matrices
  - Woodbury formulas
We formulate an adaptive version of LS SVM in dual space with window size of N. At time k, let \( x(k) = [x_k|...|x_{k+N-1}] \) denote inputs and \( d(k) = (d_k,...,d_{k+N-1})^T \). Let \( U(k) = K(x(k)) + I_N / C \).

Parameters of LS SVM given by \( b(k) \) and \( \alpha(k) = (\alpha_k,...,\alpha_{k+N-1})^T \)

\[
\begin{pmatrix} 0 & 1_N^T \\ 1_N & U(k) \end{pmatrix} \begin{pmatrix} b(k) \\ \alpha(k) \end{pmatrix} = \begin{pmatrix} 0 \\ d(k) \end{pmatrix}
\]
Adaptive LS SVM formulation

Assuming $U(k)$ is nonsingular we can use formula for the inverse of a block matrix to get that

$$b(k) = d(k)^T P(k) 1_N / 1_N^T P(k) 1_N$$

$$\alpha(k) = P(k) (1 - b(k)d(k))$$

$$P(k) = U(k)^{-1}$$

Key to algorithm is computing $P(k)$ which can be done recursively using inversion of block matrices.
Partition $U(k)$

In order to find $P(k)$ recursively first partition $U(k)$ into a block matrix containing old and new information.

\[
U(k) = \begin{pmatrix}
I(k) & L(k)^T \\
L(k) & D(k)
\end{pmatrix}
\]

\[
U(k+1) = \begin{pmatrix}
D(k) & R(k+1) \\
R(k+1)^T & r(k+1)
\end{pmatrix}
\]

$D(k)$ is a square matrix of size $N-1 \times N-1$ that is common to both $U(k)$ and $U(k+1)$. Let $Q(k) = D(k)^{-1}$
Windowed time update Recursive LS SVM algorithm

Adaptive least squares algorithm for finding $b(k)$ and $\alpha(k)$ recursively: $P(k) = U(k)^{-1}$, $D(k)$ contains common information between $U(k)$, $U(k+1)$, $Q(k) = D(k)^{-1}$

1) Initialization: Get $U(0)$, $D(0)$, compute $Q(0)$, and set $k=1$.
2) Get data $(x(k), y(k))$ and compute $P(k)$ from $Q(k-1)$.
3) Compute $b(k)$ and $\alpha(k)$.
4) Compute $Q(k)$ from $P(k)$.
5) $k \leftarrow k+1$, go to 2).

Each update of algorithm runs in $O(N^2)$ time.
Let K be kernel matrix of all training data. Reduce computations by considering a subset of K.

- K is symmetric and in many cases has eigenvalues that decay exponentially.
- Here $K_{SS}$ is $m_s \times m_s$ where $m_s \ll m$.
- Let $K_S = [K_{SS} \ K_{SN}]$.
  - Reduced system methods work with $K_{SS}$
  - Subspace methods work with $K_S$

$$K = \begin{pmatrix} K_{SS} & K_{SN} \\ K_{NS} & K_{NN} \end{pmatrix}$$
Subspace Methods

- Reduced system method use $m_S << m$ training examples $\Phi(X_S)$ resulting in kernel matrix $K_{SS} = \Phi(X_S) \Phi(X_S)^T$, but algorithm only uses a subset of information from kernel matrix $K$

- Subspace methods restrict weight to lie in subspace of $m_s$ training examples, $w = \Phi(X_S) \alpha$
  - Information matrix $A = (K_{SS}/C + K_S K_S^T)$ where $K_S = \Phi(X_S) \Phi(X)^T$ contains much more information than just $K_{SS}$
  - Information matrix dimensionality is still $m_S$
  - Higher complexity, but improved performance
Subspace LS regression equations

Optimization Problem:

\[ \min_{(w,e)} \frac{1}{2} ||w||^2 + \frac{1}{2} C \sum e(i)^2 \]

subject to

\[ d(i) - (w^T \Phi(x(i)) + b) = e(i), \forall i \]

\[ w = \Phi(X_S) \alpha. \]

Solution: \[ A = (K_{SS}/C + K_S K_S^T) (m_S \text{ by } m_S) \]

\[ A\alpha + K_S1b = K_Sd \]

\[ 1^T K_S^T \alpha + mb = 1^Td \]
Criteria for choosing SV

- Random updates
- Time updates
- Active learning: update based only on inputs
- Training error based criteria: update based on inputs and outputs
Choose training examples that give the most information.

- Assume you have hypothesis of function or concept to be learned.
  - Active Learning (unlabeled versus labeled training examples) choose unlabeled training examples that give the most information
    - Linear threshold functions: choose points close to hyperplane
  - Use input and output labels to choose training examples that will reduce training error
On-line constant sized window algorithms

Initialization: Learn from \( m_s \) training examples
Iteration: At each update decide whether to modify number of SV

- Add training example when new error can reduce training error (check angle between kernel of new example and current training error)
- Delete training example where deleted training example makes small contribution
LS SVM Related Research

- Kernel Ridge Regression
- Kernel Fisher Discriminant Analysis (pattern recognition problems)
- Radial Basis Functions
- Gaussian Processes
- Kriging
Other Sparse Methods

- Reduced system methods
  - LS SVM (Suykens, de Kruif)
  - Radial basis functions
    - Resource Allocation Network
    - Orthogonal Decompositions
    - Vector Quantization

- Subspace methods
  - LS SVM (Suykens: Nystrom and Renyi information criterion, deKruif)
  - Kernel recursive least squares (Engel et al, Rifkin)
  - Gaussian Processes (Nystrom and on-line sparse methods)
  - Sparse greed matrix approx. (Smola and Scholkopf)
Example 1: Noisy sinc function

Considered LS-SVM regression problem, formulation similar to LS-SVM classification

\[ d = \text{sinc}(t) + v \]

Using subspace methods and intelligent updating we can get roughly same performance with ten chosen SVs as 100 random points using LS SVM

Noise has deviation .3, \( \sigma = .7 \) and \( C = 1.1 \). Trained LS SVM (MSE .0109) and subspace method (MSE = .0108). Kernel eigenvalues decrease at an exponential rate

Subspace method has higher deviation than LS SVM
LS SVM and Subspace MSE
Sinc approximation with 10 SV