LMS Algorithm Summary

- Least mean square algorithm
  \[ w(k+1) = w(k) + \mu e(k) x(k) \]
- LMS algorithm is an iterative noisy gradient descent algorithm that approximates SD from one training example
- Convergence of LMS rule of thumb: \( 0 < \mu < 2 / \text{tr} (R) \). \( \text{tr}(R) > \lambda_{\text{max}} \) is more conservative bound than steepest descent bound.
- LMS algorithm tradeoff between speed of convergence and excess MSE (vary step size \( \mu \))
Step size tradeoff

- Larger step size $\mu$ quicker convergence, but more excess mean squared error, convergence time $= 2n / (\mu \text{ tr}(R))$
- Smaller step size $\mu$ slower convergence, but less excess mean squared error, $J_{\text{excess}} \approx \mu J_{\text{min}} \text{ tr}(R)/2$
- Misadjustment (dimensional quantity) proportional to step size

$$M = J_{\text{excess}} / J_{\text{min}} = .5 \mu \text{ tr } (R)$$

Example: $M = 10\%$ : excess MSE 10\% and convergence time is 20\text{n}.
Other Iterative Algorithms

- LMS algorithm with variable step size:
  \[ w(k+1) = w(k) + \mu(k)e(k)x(k) \]
  When step size \( \mu(k) = \mu/k \) algorithm converges almost surely to optimal weights.

- Conjugate gradient (CG) algorithm: Gradients and line search used to form CGs. Algorithm converges in \( n \) steps.

- Newton’s algorithm: Let \( g(n) \) be gradient and \( H(n) \) be Hessian of \( w(n) \) then approximate energy function by
  \[ J(w) \approx J(w(n)) + (w-w(n))^T g(n) + .5 (w-w(n))^T H(n)(w-w(n)) \]
  Take gradient of approximation and set to zero to get
  \[ w(n+1) = w(n) + H(n)^{-1} g(n) \]
  Algorithm involves inverting Hessian matrix (costly).
Recursive Least Square Algorithm

- Can develop an on-line version of LS algorithm called Recursive LS (RLS) algorithm.
- Algorithm based on using Sherman-Morrison-Woodbury formula:
  \[
  (A + vv^T)^{-1} = A^{-1} - A^{-1}v(1+v^TA^{-1}v)v^TA^{-1}
  \]
  where \(A = X^TX\) contains old data and \(v = x(m+1)\) contains new data at time \(m+1\).
- Similar to Kalman filter equations where we update estimate recursively adding new information or innovations.
- Update is \(O(n^2)\) operations.
RLS Algorithm Comments

- Often exponentially weighted algorithm implemented. Update correlation matrix, gain factor, and weights.
- Parameters of RLS algorithm: Initial correlation matrix and weight decay factor.
- Convergence is typically an order of magnitude faster than LMS algorithm. Algorithm theoretically converges to zero excess mean squared error and convergence does not depend on eigenvalues.
- Many variations to account for more stable matrix computations: QR and Cholesky factorizations.
Nonlinear Methods

- Multilayer feedforward networks: error back propagation algorithm

- Kernel methods:
  - Support Vector Machines (SVM)
  - Least squares methods
  - Radial Basis Functions (RBF)
Kernel Methods

In many classification and detection problems a linear classifier is not sufficient. However, working in higher dimensions can lead to “curse of dimensionality”.

Solution: Use kernel methods where computations done in dual observation space.

\[ \Phi: X \rightarrow Z \]
Linear SVM (dual representation)

SVM quadratic programming problem involves computing inner products.

\[
\begin{align*}
\max \quad W(\alpha) &= \sum \alpha(i) - \frac{1}{2} \sum \alpha(i) \alpha(j) d(i) d(j) (x(i)^T x(j)) \\
\text{subject to} \quad \alpha(i) &\geq 0, \text{ and } \sum \alpha(i) d(i) = 0.
\end{align*}
\]

The hyperplane decision function can be written as

\[
f(x) = \text{sgn} \left( \sum \alpha(i) d(i) x^T x(i) + b \right)
\]
Polynomial Transformations

- $\Phi$ is a transformation from input space to higher order polynomial.
- Example: let $x = (x_1, x_2)^T$, then quadratic polynomial can be represented as $\Phi(x) = (x_1^2, x_2^2, (2)^{\frac{1}{2}} x_1 x_2)^T$ and computations can be represented by kernels using inner products $k(x, y) = \Phi(x) \Phi(z)^T = x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 z_1 x_2 z_2$
- With polynomials feature space dimension will get large but kernel trick allows us to do computations using kernel functions $k(x, z) = (x^T z)^2$ without using transformation functions $\phi(x)$. 
Mercer Kernels

- \( \Phi \) is a transformation from input space \( \mathbb{R}^n \) to feature space \( \mathbb{R}^d \) (which could be infinite dimensional).
- We consider Mercer kernels that can be represented as
  \[
  K(x,z) = \sum_i \lambda_i \varphi_i(x) \varphi_i(z) = \phi(x)^T \phi(z)
  \]
  where the kernel functions are positive semi-definite
  \[
  \int K(x,z)g(x)g(z)dx \, dz \geq 0
  \]
  for any \( g(x) \) that are square integrable functions.
- Kernel trick allows us to do computations using kernel functions \( K(x,z) \) without knowledge of transformation functions \( \phi(x) \).
Examples of Kernel Functions

- Polynomial Kernel of order $p$: $K(x,z) = (x^T z)^p$
- Polynomial Kernel of order $\leq p$: $K(x,z) = (x^T z + 1)^p$
- Gaussian Kernel: $K(x,z) = \exp(-||x-z||^2/(2\sigma^2))$
- Sigmoidal Kernel: $K(x,z) = \tanh(ax^T z - b)$ for some values of $a$ and $b$
- Bilinear transformation: $K(x,z) = f(x)f(z)$
- Other kernels: splines, strings, $K(x,z) = \min(x,z)$
Kernel Properties

- Symmetric: $K(x,z) = K(z,x)$
- Schwarz Inequality: $|K(x,z)| \leq (K(x,x),K(z,z))^{1/2}$
- Closure properties
- Reproducing Kernel Hilbert Spaces (RKHS)
- Autocorrelation function of a second order process, (Gaussian processes)
Closure Properties and Feature Space Representation

Let $K_1(x,z)$ and $K_2(x,z)$ be kernel functions

- **Additive**, $K(x,z) = K_1(x,z) + K_2(x,z)$: $\Phi(x) = (\Phi_1(x), \Phi_2(z))^T$
- **Positive scaling**, $K(x,z) = \alpha K_1(x,z)$ for $\alpha > 0$:
  $$\Phi(x) = \alpha^{\frac{1}{2}} \Phi_1(x)$$
- **Product**, $K(x,z) = K_1(x,z) K_2(x,z)$:
  $$\Phi(x)(i,j) = \Phi_1(x)(i), \Phi_2(z)(j)$$ (tensor product)
- **Symmetric positive semidefinite matrix $A$**, $K(x,z) = x^T A z$:
  $$\Phi(x) = L^T \Phi_1(x)$$ ($A=L L^T$ Cholesky factorization)
- **Bilinear transformation**, $K(x,z) = f(x)f(z)$: $\Phi(x) = f(x)$
Support Vector Machine

Optimal margin classifier with slack variables and kernel functions described by Support Vector Machine (SVM).

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

subject to \(\xi_i \geq 0 \ \forall i\), \(d(i) (w^T \phi(x(i)) + b) \geq 1 - \xi_i\), \(\forall i\), and \(C > 0\).

(Hinge loss function)

In dual space

\[
\max W(\alpha) = \sum \alpha(i) - \frac{1}{2} \sum \alpha(i) \alpha(j) d(i)d(j) K(x(i),x(j))
\]

subject to \(C \geq \alpha(i) \geq 0\), and \(\sum \alpha(i)d(i) = 0\).

Weights can be found by \(w = \sum \alpha(i) d(i) \phi(x(i))\).
Representation of decision surface

- In primal space decision surface is a linear hyperplane in feature space and can be represented as
  \[ f(x) = \text{sgn} \left( w^T \phi(x(i)) + b \right) \]

- In dual space decision surface can be represented via kernels and Lagrange multipliers as
  \[ f(x) = \text{sgn} \left( \sum \alpha_i d(i) K(x, x(i)) + b \right) \]
Dual representation of SVM
Other Kernel Applications

- Multiclass problems
- Regression problems (\(\varepsilon\)-insensitive loss function)
- Quadratic cost function with equality constraints
  - LS SVM pattern classification, Kernel Fisher Discriminant Analysis
  - LS SVM regression, Kernel ridge regression
- Unsupervised Learning
  - Principal Component Analysis (PCA)
  - Kernel PCA
SVM Regression

For regression problems target output $y \in \mathbb{R}$. Estimates will have error and we need to decide on what error criteria to use. Let error $e = d - f(x)$, then $\varepsilon$-insensitive cost criterion defined as $(|e| - \varepsilon) u(\varepsilon - |e|)$. 
SVM Regression QP Formulation

$$\min_{(w, b, \xi, \xi^*)} \frac{1}{2}||w||^2 + C \sum (\xi_i + \xi_i^*)$$

subject to

$$\xi_i \geq 0 \ \forall i, \ d(i) - w^T \phi(x(i)) - b \leq \varepsilon + \xi_i$$

$$\xi_i^* \geq 0 \ \forall i, \ w^T \phi(x(i)) + b - d(i) \leq \varepsilon + \xi_i^*$$

for $1 \leq i \leq m$

Points that lie outside of $\varepsilon$ tube are SV.
SVM Regression

QP Dual Space Representation

Like SVM classification we use Lagrange multipliers to add inequality constraints to objective function and then take partial derivatives to get the following dual space representation.

\[
\max \ W(\alpha) = \sum d(i)(\alpha(i) - \alpha(i)*)
\]

\[
-\frac{1}{2} \sum \sum (\alpha(i) - \alpha(i)*) (\alpha(j) - \alpha(j)*) K(x(i), x(j)) - \varepsilon \sum (\alpha(i) + \alpha(i)*)
\]

subject to \( 0 \leq \alpha(i) \leq C, 0 \leq \alpha(i)* \leq C, \) and \( \sum (\alpha(i) - \alpha(i)*) = 0 \).

Dual space representation

\[
f(x) = \sum (\alpha(i) - \alpha(i)*) K(x, x(i)) + b
\]
Least Squares SVM Regression

Consider changing SVM to LS SVM by making following modifications:

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

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

Key differences with between SVM and LS SVM:

- \(\epsilon\) - insensitive cost replaced by quadratic error cost.
- Inequality constraint replaced by equality constraint.
Primal Solution

Substitute for $e_i$ and take partial derivatives of objective function with respect to $w$ and $b$ and set to 0. This yields least square solution given by

$$(R(l) + I/(lC))w = P(l)$$

$$m_X(l)^T w + b = m_Y(l)$$

where $m_X(l)$ and $m_Y(l)$ are sample means of $\Phi(X)$ and $Y$ respectively. $R(l)$ and $P(l)$ are sample autocorrelation of $\Phi(X)$ and crosscorrelation of $\Phi(X)$ and $Y$ respectively.

If $(R(l) + I/(lC))$ is nonsingular we have that

$$w = (R(l) + I/(lC))^{-1}P(l)$$

giving MSE solution plus regularization term.
Finding Dual Solution

Introduce Lagrange multipliers

\[ L(w,b,e,\alpha) = \frac{1}{2}||w||^2 + \frac{1}{2}C \sum e(i)^2 - \sum \alpha(i) (d(i) - (w^T \Phi(x(i)) + b) - e(i)) \]

where \( \alpha(i) \geq 0 \).
KKT Conditions

Again take partial derivatives and set to 0.
\[ \frac{\partial L(w,b,e,\alpha)}{\partial w} = 0, \frac{\partial L(w,b,e,\alpha)}{\partial b} = 0, \]
\[ \frac{\partial L(w,b,e,\alpha)}{\partial \alpha} = \frac{\partial L(w,b,e,\alpha)}{\partial e(I)} = 0. \]

We therefore have that
\[ w = \sum \alpha(i) \Phi(x(i)) \]
\[ \sum \alpha(i) = 0 \]
\[ \alpha(i) = C e(i), \quad 1 \leq i \leq m \]
\[ d(i) - (w^T \Phi(x(i)) + b) - e(i) = 0, \quad 1 \leq i \leq m \]
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 \( l \) vector of \( 1 \)s.

\[
f(x) = \sum \alpha(i) K(x^T x(i)) + b
\]
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, $e$. 
Similarity to other methods

- Kernel Fisher Discriminant Analysis
- Proximal Support Vector Machines
- Radial Basis Functions
- Gaussian Processes
- Kriging
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