Efficiently Computing LS

Monday, April 10, 2017 9:03 AM

Summary of Least squares problems

pseudo-inverse when A is full column rank $\quad \hat{x} = (A^T A)^{-1} A^T y$

pseudo-inverse when A is full row rank $\quad \hat{x} = A^T (A A^T)^{-1} y$

Tikhonov regularization $\quad \hat{x} = (A^T A + \delta I)^{-1} A^T y$

generalized Tikhonov reg. $\quad \hat{x} = (A^T A + \delta D^T D)^{-1} A^T y$

weighted least squares $\quad \hat{x} = (A^T W^T W A)^{-1} A^T W^T W y$

BLUE $\quad \hat{x} = (A^T R^{-1} A)^{-1} A^T R^{-1} y$

each solves a symmetric, positive definite set of equations

There are better ways to solve these than with direct computation

if we have a triangular system

$$\begin{bmatrix} a_{11} & 0 & & \\ a_{21} & a_{22} & & 0 \\ \vdots & & \ddots & \\ a_{N1} & & & a_{NN} \end{bmatrix} x = y$$

solve for x

$$x_1 = y_1 / a_{11}$$

$$x_2 = \frac{1}{a_{22}} (y_2 - a_{21} \cdot x_1)$$

we know each of these terms

$$\vdots$$

we can solve this with $O(N^2)$ operations instead of

$x = A^{-1} y \leftarrow O(N^3)$

Exploiting Structure

Monday, April 10, 2017 9:20 AM

if $A\hat{x} = y$

solving for x

some clever algorithms

if A is diagonal	$O(N)$	
if A is triangular	$O(N^2)$ *	
if A is orthogonal	$O(N^2)$ *	$\hat{x} = A^T y$
if A is Toeplitz — correlation matrices	$O(N^2)$	$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 10 & 1 & 2 & 3 \\ 7 & 10 & 1 & 2 \\ 3 & 7 & 10 & 1 \end{pmatrix}$
if A is Circulant	$O(N \log N)$	
if A is general	$O(N^3)$	

Levinson-Durbin algorithm

Performing a Cholesky decomposition on sym+def A

$A = LL^T$ ← we can use the triangular method twice to find a solution to $Ax = y$ $O(N^2)$

Same for QR decomposition (can be found using Gram-Schmidt)

$A = QR$ $O(N^2)$

Q: orthonormal; R: upper triangular

used in communications

often used if the $Ax=y$ equation needs to be solved for multiple y's. But, they are often used even for a single y because they are more numerically stable than solutions involving A^{-1}.

Iterative methods for Least Squares

Monday, April 10, 2017 9:39 AM

$$\boxed{\hat{x} = (\underbrace{A^T A}_{\text{costs } O(MN^2)})^{-1} A^T y} \qquad A \in \mathbb{R}^{M \times N}$$

$$\underbrace{A^T A \hat{x} = A^T y}_{\text{solving this system directly is } O(N^3)}$$

example: MRI $\quad M = 5 \cdot 10^6$ by $N = 2 \cdot 10^6$

$MN^2 \sim 10^{19}$

for such a system, we need another solution

Recasting as an optimization

$$\underbrace{A^T A}_{H} \, x = \underbrace{A^T y}_{b}$$

$$H x = b$$

sym+def

The following is a quadratic program and is convex

$$\min_{x \in \mathbb{R}^N} \frac{1}{2} x^T H x - x^T b$$

proceed by setting the derivative to zero

$$\nabla_x \left(\frac{1}{2} x^T H x - x^T b \right) \Big|_{x = \hat{x}} = 0$$

$$\Downarrow$$

$$H x - b = 0$$

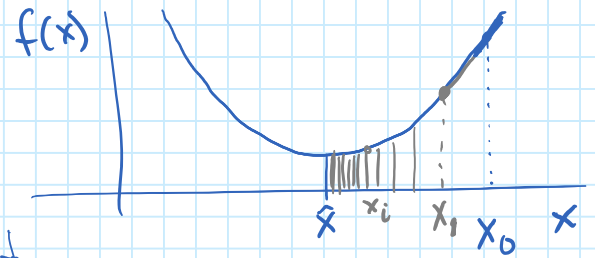
$$\Downarrow$$

$$H x = b$$

Gradient Descent

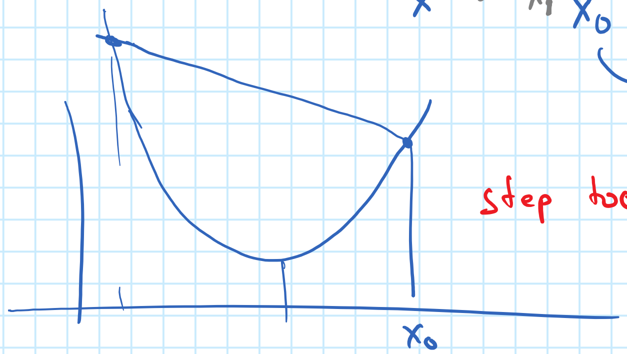
Monday, April 10, 2017 9:48 AM

$f(x) = xHx - xb$ $\leftarrow$ one dimensional example

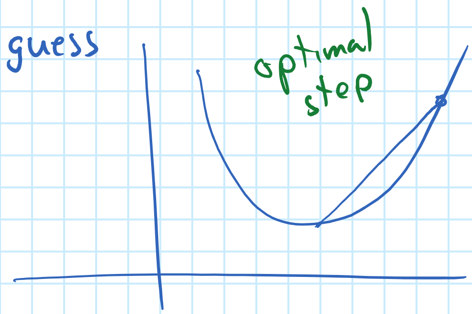


$$x_1 = x_0 - \alpha_0 \nabla f(x)\big|_{x=x_0}$$

initial guess



step too large



we can define $\underbrace{b - Hx_k}_{-\nabla f(x)|_{x=x_k}} = r_k$ — residual

$$x_{k+1} = x_k + \alpha_k r_k \quad \leftarrow \text{update equation}$$

to make $f(x_{k+1})$ as small as possible

$f(x_k + \alpha_k r_k)$ $\leftarrow$ we can solve for α that minimizes this

$$\frac{\partial}{\partial \alpha} f(x_k + \alpha r_k) = 0$$

$$= \nabla f(x_{k+1})^T \frac{\partial}{\partial \alpha} x_{k+1} \quad \leftarrow \text{using chain rule}$$

$$= \nabla f(x_{k+1})^T r_k$$

we need $r_k \perp \nabla f(x_{k+1})$

Thus $r_{k+1} \perp r_k$

Gradient Continued

Monday, April 10, 2017 10:00 AM

$$r_{k+1}^T r_k = 0$$

$$(b - Hx_{k+1})^T r_k = 0$$

$$(b - H(x_k + \alpha_k r_k))^T r_k = 0$$

$$\vdots$$

$$\alpha_k = \frac{r_k^T r_k}{r_k^T H r_k}$$

ensures rapid convergence.

for a large problem 20-50 steps.

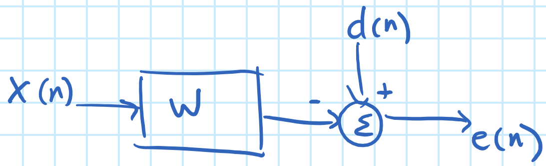
In general, moving in the proper direction with a non-zero, but sufficiently small step size will yield convergence.

LMS algorithm

we are trying to minimize $E\{e^2[n]\}$ where

$$e[n] = d[n] - \underline{w}^T \underline{x}[n]$$

$$\underline{x}[n] = \begin{pmatrix} x(n) \\ x(n-1) \\ \vdots \\ x(n-p) \end{pmatrix}$$



$$\nabla_w E\{e^2[n]\} = E\{\nabla_w e^2[n]\} = E\{2e[n]\nabla_w e[n]\}$$

$$= \underbrace{-E\{2e[n]\,\underline{x}[n]\}}_{1} = 0$$

gives the solution exactly

LMS Algorithm

Monday, April 10, 2017 10:12 AM

$$E\{e[n]\,\underline{x}[n]\} = E\{(d[n] - \underbrace{\underline{w}^T \underline{x}[n]}_{\text{scalar}})\,\underline{x}[n]\}$$

$$= \underbrace{E\{d[n]\,\underline{x}[n]\}}_{\substack{\text{cross correlation of} \\ x[n] \text{ with } d[n]}} - E\{\underline{x}[n]\,\underline{x}^T[n]\}\underline{w}$$

$$= \underline{r}_{dx} - R_x \underline{w} = 0 \quad \Rightarrow \quad \underbrace{R_x \underline{w} = \underline{r}_{dx}}_{\text{The normal equation}}$$

we have seen problems like this before!

Instead of solving the normal equation directly, we will use gradient descent

$$\underline{w}_{n+1} = \underline{w}_n - \frac{1}{2}\mu \nabla_w E\{e^2[n]\}$$

$$\boxed{\underline{w}_{n+1} = \underline{w}_n + \mu E\{e[n]\,\underline{x}[n]\}}$$

from ①

how to find?

$$\hat{E}\{e[n]\,\underline{x}[n]\} = \frac{1}{N}\sum_{k=n-N+1}^{n} e(k)\,\underline{x}(k)$$

(sample mean)

The LMS algorithm uses

$$E\{e[n]\,\underline{x}[n]\} \simeq e[n]\,\underline{x}[n]$$

$$\boxed{\underline{w}_{n+1} = \underline{w}_n + \mu e[n]\,\underline{x}[n]}$$

LMS Algorithm Practical Issues

Wednesday, April 19, 2017 9:06 AM

$y[n] = \underline{W}^T[n]\, \underline{x}(n)$ $\qquad \underline{x}(n) = \begin{pmatrix} x(n) \\ x(n-1) \\ \vdots \\ x(n-p) \end{pmatrix} \qquad w[n] = \begin{pmatrix} w_0(n) \\ w_1(n) \\ \vdots \\ w_p(n) \end{pmatrix}$

$e(n) = d(n) - y(n)$

"d" for desired or target signal

$\underline{W}[n+1] = \underline{W}[n] + \mu\, e(n)\, \underline{x}[n]$ minimizes the expected error, $E\{e^2(n)\}$

$R_x = E\{\underline{x}[n]\, \underline{x}^T[n]\}$ ← correlation matrix

if $\{\lambda_i\}$ are the eigenvalues of R_x then

$0 < \mu < 2/\lambda_1$

$= \lambda_{max}$

$\lambda_{max} \le \sum_{k=0}^{p} \lambda_k = Tr(R_x) = \sum_{k=0}^{p} E\{x^2(n-k)\} = \ldots$

if $x(n)$ is stationary (or WSS)

$= (p+1) \cdot (E\{x^2(n)\}) \approx \sum_{k=0}^{p} x^2(n-k) = \underline{x}^T(n)\, \underline{x}(n)$

approx result from assumption in convergence proof.

$0 < \mu < 2/\lambda_{max} \Rightarrow 0 < \mu < 2/\underline{x}^T(n)\,\underline{x}(n)$

we define $\mu(n) = \dfrac{\beta}{\underline{x}^T(n)\, \underline{x}(n)}$ where $0 < \beta < 2$

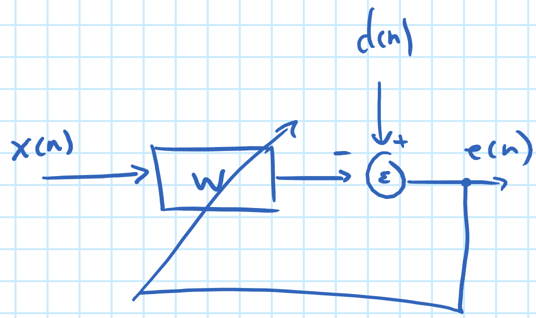
in practice, we choose $\beta = 0.5$

(or in the range 0.1 to 1)

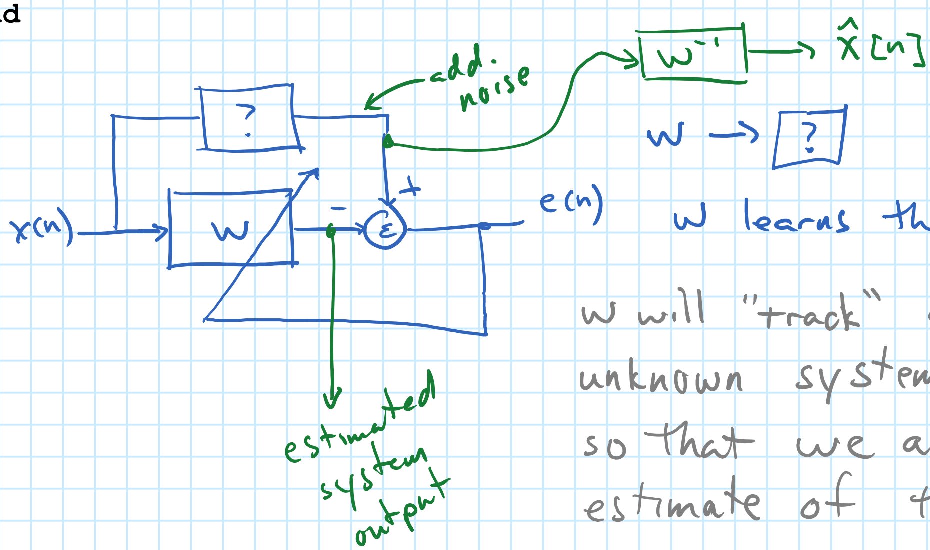
nLMS Adaptive Filters

Tuesday, April 18, 2017 4:34 PM

$$\boxed{\underline{w}[n+1] = \underline{w}[n] + \frac{\beta}{\|\underline{x}(n)\|^2}\,\underline{x}(n)\,e(n)}$$

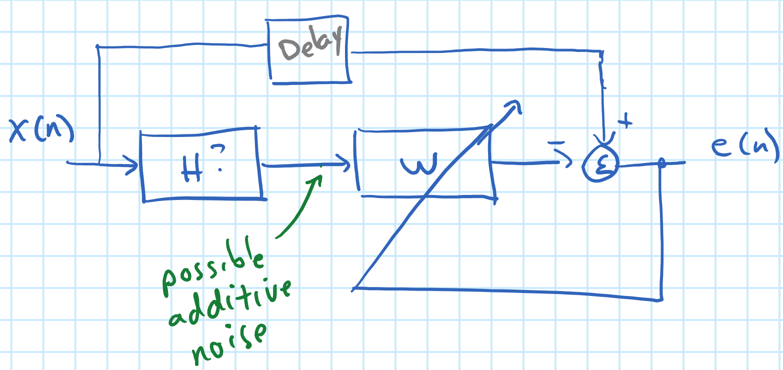



```
function [y,e,w]=mylms(x,d,w,beta)
% length x = length d

x=x(:); d=d(:); w=w(:);
y = zeros(size(x));
e = zeros(size(x));

for i=length(w):length(x),
  xx      = x(i:-1:i-length(w)+1);
  y(i)    = w' * xx;
  e(i)    = d(i) - y(i);
  w       = w + beta/(xx'*xx+1e-8) * e(i) * xx;
end
```



w learns the unknown system

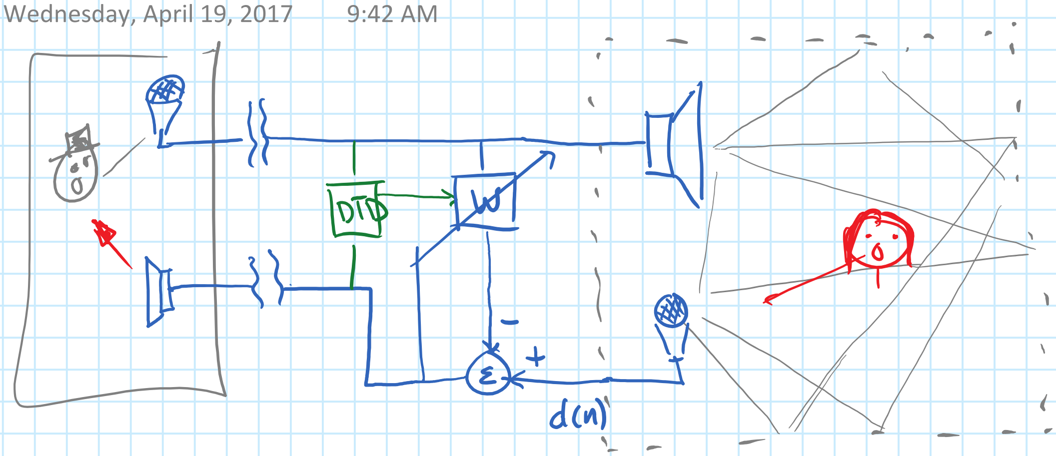
w will "track" or follow the unknown system as it changes so that we always have an estimate of the system.



$w \to H^{-1}$

Echo Cancellation

Wednesday, April 19, 2017 9:42 AM



Adaptive LMS filters adapt most quickly when x[n] is white noise.

adaptation rate is a function of

$\frac{\lambda_{max}}{\lambda_{min}}$ → larger spread = slower adaptation