Least Mean Square (LMS) Algorithm
Sumeetpal Singh
Engineering Department
Email: sss40@eng.cam.ac.uk
1 Outline

• The LMS algorithm
• Overview of LMS issues concerning step-size bound and convergence
• Some simulation examples
• The normalised LMS (NLMS)
2 Least Mean Square (LMS)

• Steepest Descent (SD) was

\[ h(n + 1) = h(n) - \frac{\mu}{2} \nabla J(h(n)) \]

\[ = h(n) + \mu E \{ u(n) e(n) \} \]

• Often \( \nabla J(h(n)) = -2E \{ u(n) e(n) \} \) is unknown or too difficult to derive

• Remedy is to use the instantaneous approximation \(-2u(n)e(n)\) for \( \nabla J(h(n)) \)

• Using this approximation we get the LMS algorithm

\[ e(n) = d(n) - h^T(n)u(n), \]

\[ h(n + 1) = h(n) + \mu e(n)u(n) \]
• This is desirable because
  – We do not need knowledge of $\mathbf{R}$ and $\mathbf{p}$ anymore
  – If statistics are changing over time, it adapts accordingly
  – Complexity: $2M + 1$ multiplications and $2M$ additions per iteration. Not $M^2$ multiplications like SD

• Undesirable because we have to choose $\mu$ when $\mathbf{R}$ not known, subtle convergence analysis
3 Application: Noise Cancellation

- Mic 1: Reference signal
  \[ d(n) = s(n) + v(n) \]
  - signal of interest
  - noise
  \( s(n) \) and \( v(n) \) statistically independent

- Aim: recover signal of interest

- Method: use another mic, Mic 2, to record noise only, \( u(n) \)

- Although \( u(n) \neq v(n) \), \( u(n) \) and \( v(n) \) are correlated

- Now \textbf{filter} recorded noise \( u(n) \) to minimise \( E\{e(n)^2\} \), i.e. to cancel \( v(n) \)

- Recovered signal is \( e(n) = d(n) - h(n)^T u(n) \) and not \( y(n) \)

- \textbf{Run} Matlab demo on webpage
• We are going to see an example with speech $s(n)$ generated as a mean 0 variance 1 Gaussian random variable

• Mic 1’s noise was $0.5 \sin(n\frac{\pi}{2} + 0.5)$

• Mic 2’s noise was $10 \sin(n\frac{\pi}{2})$

• Mic 1 and 2’s noise are both sinusoids but with different amplitudes and phase shifts

• You could increase the phase shift but you will need a larger value for $M$

• **Run** Matlab demo on webpage
LMS convergence in mean

- Write the reference signal model as
  \[ d(n) = u^T(n) h_{opt} + \varepsilon(n) \]
  \[ \varepsilon(n) = d(n) - u^T(n) h_{opt} \]
  where \( h_{opt} = R^{-1}p \) denotes the optimal vector (Wiener filter) that \( h(n) \) should converge to.

- For this reference signal model, the LMS becomes
  \[ h(n + 1) = h(n) + \mu u(n) \times \left( u^T(n) h_{opt} + \varepsilon(n) - u^T(n) h(n) \right) \]
  \[ = h(n) + \mu u(n) u^T(n) \times \left( h_{opt} - h(n) \right) + \mu u(n) \varepsilon(n) \]
  \[ h(n + 1) - h_{opt} = \left( I - \mu u(n) u^T(n) \right) (h(n) - h_{opt}) + \mu u(n) \varepsilon(n) \]
• This looks like a noisy version of the SD recursion

\[ h(n + 1) - h_{\text{opt}} = (I - \mu R) \left( h(n) - h_{\text{opt}} \right) \]

• Verify that \( E \{ u(n) \varepsilon(n) \} = 0 \) using \( h_{\text{opt}} = R^{-1} p \)

• Introducing the expectation operator gives

\[
E \left\{ h(n + 1) - h_{\text{opt}} \right\} \\
= E \left\{ \left( I - \mu u(n) u^T(n) \right) \left( h(n) - h_{\text{opt}} \right) \right\} \\
+ \mu E \{ u(n) \varepsilon(n) \} = 0 \\
\approx \left( I - \mu E \left\{ u(n) u^T(n) \right\} \right) E \{ h(n) - h_{\text{opt}} \} \\
\text{(Independence approximation)} \\
= (I - \mu R) E \{ h(n) - h_{\text{opt}} \} \]
• *Independence approximation* assumes $h(n) - h_{\text{opt}}$ is independent of $u(n)u^T(n)$
  - Since $h(n)$ function of $u(0), u(1), \ldots, u(n-1)$ and all previous desired signals this is not true
  - However, the approximation is better justified for a “block” LMS type update scheme where the filter is updated at multiples of some block length $L$, i.e. when $n = kL$ and not otherwise
• Idea is to not update $h(n)$ except when $n$ is an integer multiple of $L$, i.e. $n = kL$ for $k = 0, 1, \ldots$

$$h(n + 1) = h(n) + \mu(n + 1)e(n)u(n)$$

$$e(n) = d(n) - h(n)^T u(n)$$

$$\mu(n) = \begin{cases} 
\mu & \text{if } n/L = \text{integer} \\
0 & \text{otherwise}
\end{cases}$$

• Also $L$ should be much larger than filter length $M$

• This means $h(kL) = h(kL + 1) = \cdots = h(kL + L - 1)$

• Re-use the previous derivation which is still valid:

$$E \{h(n + 1) - h_{opt}\} = E \left\{ \left( I - \mu(n + 1)u(n)u(n)^T \right) (h(n) - h_{opt}) \right\}$$
• When \( n + 1 = kL + L \) we have

\[
E \{ h(n + 1) - h_{\text{opt}} \} = E \left\{ (I - \mu u(n)u(n)^T) (h(kL) - h_{\text{opt}}) \right\} \\
\approx E \left\{ I - \mu u(n)u(n)^T \right\} E \left\{ h(kL) - h_{\text{opt}} \right\}
\]

\[
E \{ h(kL + L) - h_{\text{opt}} \} \approx (I - \mu R) E \left\{ h(kL) - h_{\text{opt}} \right\}
\]

• This analysis uses the fact that \((u(0), \ldots, u(i))\) and \((u(j), \ldots, u(j + M - 1))\), for \( j > i \), become independent as \( j - i \) increases. True for some ARMA time-series.

• We are back to the SD scenario and so

\[
E \{ h(n) \} \to h_{\text{opt}} \quad \text{if } 0 < \mu < \frac{2}{\lambda_{\text{max}}}
\]

• Behaviour predicted using the analysis of the block LMS agrees with experiments and computer simulations even for \( L = 1 \)

• We will always use \( \mu(n) = \mu \) for all \( n \). Block LMS version just to understand long-term behaviour.
• The point of the LMS was that we don’t have access to $\mathbf{R}$, so how to compute $\lambda_{\text{max}}$?

• Using the fact that

$$
\sum_{k=1}^{M} \lambda_k = \text{tr} (\mathbf{R}) = M \mathbb{E} \left\{ u^2(n) \right\}
$$

we have that $\lambda_{\text{max}} < \sum_{k=1}^{M} \lambda_k = M \mathbb{E} \left\{ u^2(n) \right\}$

• Note that we can estimate $\mathbb{E} \left\{ u^2(n) \right\}$ by a simple sample average and the new tighter bound on the stepsize is

$$
0 < \mu < \frac{2}{M \mathbb{E} \left\{ u^2(n) \right\}} < \frac{2}{\lambda_{\text{max}}}
$$
With a fixed stepsize, \( \{h(n)\}_{n \geq 0} \) will never settle at \( h_{opt} \), but rather oscillate about \( h_{opt} \). Even if \( h(n) = h_{opt} \) then

\[
h(n + 1) - h_{opt} = \mu u(n)e(n) = \mu u(n) \left( d(n) - u^T(n) h_{opt} \right),
\]

and because \( u(n)e(n) \) is random, \( h(n + 1) \) will move away from \( h_{opt} \).

5 LMS main points

- Simple to implement
- Works fine is many applications if filter order and stepsize is chosen properly
- There is a trade-off effect with the stepsize choice
  - large \( \mu \) yields better tracking ability in a non-stationary environment but will have larger fluctuations of \( h(n) \) about converged value
  - small \( \mu \) has poorer tracking ability but less of such fluctuations
6 Adaptive stepsize: Normalised LMS (NLMS)

- We showed that LMS was stable provided
  \[ \mu < \frac{2}{ME \{u^2(n)\}} \]

- What if \( E \{u^2(n)\} \) varied, which would be true for a non-stationary input signal

- LMS should be able to adapt its step-size automatically

- The instantaneous estimate of \( ME \{u^2(n)\} \) is \( u^T(n)u(n) \)

- Now replace the LMS stepsize with \( \frac{\mu'}{u^T(n)u(n)} = \frac{\mu'}{\|u(n)\|^2} \) where \( \mu' \) is a constant that should < 2, e.g., \( 0.25 < \mu' < 0.75 \). We make \( \mu' \) smaller because of the poor quality estimate for \( ME \{u^2(n)\} \) in the denominator
This choice of stepsize gives the Normalized Least Mean Squares (NLMS)

\[ e(n) = d(n) - u^T(n) h(n) \]

\[ h(n+1) = h(n) + \frac{\mu}{\|u(n)\|^2} e(n) u(n) \]

where \(\mu'\) is relabelled to \(\mu\). NLMS is the LMS algorithm with a data-dependent stepsize.

Note small amplitudes will now adversely effect the NLMS. To better stabilise the NLMS use

\[ h(n+1) = h(n) + \frac{\mu}{\|u(n)\|^2 + \epsilon} e(n) u(n) \]

where \(\epsilon\) is a small constant, e.g. 0.0001.
7 Comparing NLMS and LMS

- Compare the stability of the LMS and NLMS for different values of stepsize. You will see that the NLMS is stable for $0 < \mu < 2$. You will still need to tune $\mu$ to get the desired convergence behaviour (or fluctuations of $h(n)$ once it has stabilized) though.

- Run the NLMS example on the course website