4F7 Adaptive Filters (and Spectrum Estimation)

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 stepsize bound, convergence, misadjustment
• Some simulation examples
• The normalised LMS (NLMS)
• Further topics
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 $R$ and $p$ 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 $R$ not known, subtle convergence analysis
3 Application: Noise Cancellation

- Mic 1: Reference signal
  \[ d(n) = s(n) + v(n) \]
  - \( s(n) \) and \( v(n) \) statistically independent
- Aim: recover signal of interest
- Method: use another mic, Mic 2, to record noise only, \( u(n) \)
- Obviously \( u(n) \neq v(n) \) but \( u(n) \) and \( v(n) \) are hopefully correlated
- Now filter recorded noise \( u(n) \) to make it more like \( v(n) \)
- Recovered signal is \( e(n) = d(n) - h(n)^T u(n) \) and not \( y(n) \)
- 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

\[ d(n) = u^T(n) h_{opt} + \varepsilon(n) \]

where \( h_{opt} \) denotes the optimal vector (Wiener filter) that \( h(n) \) should converge to and \( \varepsilon(n) = d(n) - u^T(n) h_{opt} \)

• For this reference signal model, the LMS becomes

\[
\begin{align*}
h(n+1) &= h(n) + \mu u(n) \\
&\times \left( u^T(n) h_{opt} + \varepsilon(n) - u^T(n) h(n) \right) \\
&= h(n) + \mu u(n) u^T(n) \\
&\times (h_{opt} - h(n)) + \mu u(n) \varepsilon(n)
\end{align*}
\]

\[
\begin{align*}
h(n+1) - h_{opt} &= \left( I - \mu u(n) u^T(n) \right) (h(n) - h_{opt}) \\
&\quad + \mu u(n) \varepsilon(n)
\end{align*}
\]

• This looks like the SD recursion

\[
\begin{align*}
h(n+1) - h_{opt} &= (I - \mu R) (h(n) - h_{opt})
\end{align*}
\]
Introducing the expectation operator gives

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

(Independence assumption)

\[
= (I - \mu R) E \{ h(n) - h_{\text{opt}} \}
\]

• *Independence assumption* states that \( h(n) - h_{\text{opt}} \) 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, only true if \( \{ u(n) \} \) is an independent sequence of random vectors but not so otherwise. Assumption 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
• We are back to the SD scenario and so

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

• Assuming \( h(n) \), being a function of \( u(0), u(1), \ldots, u(n - 1) \), is independent of \( u(n) \) is not true. However, behaviour predicted using this simplifying assumption agrees with experiments and computer simulations.

• The point of the LMS was that we don’t have access to \( R \), so how to compute \( \lambda_{\text{max}} \)?

• Solution by a time average: using the fact that for square matrices \( \text{tr}(AB) = \text{tr}(BA) \), and \( R = QAQ^T \),
\[
\sum_{k=1}^{M} \lambda_k = \text{tr} (\Lambda) \\
= \text{tr} \left( \Lambda Q^T Q \right) \\
= \text{tr} \left( Q \Lambda Q^T \right) \\
= \text{tr} (R) \\
= ME \left\{ u^2 (n) \right\}
\]

and we see that \( \lambda_{\text{max}} < \sum_{k=1}^{M} \lambda_k = ME \left\{ u^2 (n) \right\} \)

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

\[
0 < \mu < \frac{2}{ME \left\{ u^2 (n) \right\}} < \frac{2}{\lambda_{\text{max}}}
\]

- With a fixed stepsize, \( \{h(n)\}_{n \geq 0} \) will never settle at \( h_{\text{opt}} \), but rather
oscillate about $h_{\text{opt}}$. Even if $h(n) = h_{\text{opt}}$ then

$$h(n + 1) - h_{\text{opt}} = \mu u(n) e(n),$$

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

- Assume $d(n) = u(n)^T h_{\text{opt}} + \varepsilon(n)$ where $\varepsilon(n)$ is a zero mean independent noise sequence. Best case for $E\{e^2(n)\}$, using $e(n) = u^T(n) h_{\text{opt}} + \varepsilon(n) - u^T(n) h(n)$, is when $h(n) = h_{\text{opt}}$, so that $E\{e^2(n)\} = E\{\varepsilon^2(n)\} (= J_{\text{min}})$. However, since $h(n) \neq h_{\text{opt}}$, we would expect $E\{e^2(n)\} > J_{\text{min}}$. We call

$$\frac{E\{e^2(n)\} - J_{\text{min}}}{J_{\text{min}}}$$

the misadjustment
LMS main points

• Simple to implement
• Works fine in 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 has large misadjustment noise
  – small \( \mu \) has poorer tracking ability but has smaller misadjustment noise
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 \( u^T(n) u(n) = \frac{\mu'}{||u(n)||^2} \) 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.

- There is another interpretation of the NLMS. It can be shown \(h(n+1)\) is the solution to the following optimization problem (Example Sheet):

\[
h(n+1) = \arg\min_{h} \|h - h(n)\|
\]

s.t. \(d(n) - u^T(n)h = 0\)

*In light of new data, the parameters of an adaptive system should only be perturbed in a minimal fashion.*

- Another motivation for the NLMS is to reduce the sensitivity of the filter update to large amplitude fluctuations in \(u(n)\)
- But small amplitudes will now adversely effect the NLMS. The solution
is
\[ h(n+1) = h(n) + \frac{\mu}{\|u(n)\|^2 + \epsilon} e(n) u(n) \]

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
8 Nonlinear LMS

- LMS algorithm uses linear combination of
  \( u(n), u(n-1), \ldots, u(n-M+1) \) (i.e. \( \mathbf{u}(n) \)) to predict \( d(n) \)

  \[
  y(n) = \mathbf{h}^T(n) \mathbf{u}(n)
  \]

- Why not nonlinear, e.g., quadratic

  \[
  y(n) = \sum_{i=0}^{M-1} h_{1,i}(n) u(n-i)
  \]

  \[
  + \sum_{i=0}^{M-1} \sum_{j=i}^{M-1} h_{2,i,j}(n) u(n-i) u(n-j)
  \]

- This case can be written in the usual form

  \[
  y(n) = \underbrace{\theta^T(n)}_{\text{Linear Parameters}} \times \underbrace{\Psi(n)}_{\text{Function of } \mathbf{u}(n)}
  \]

  and all we have done with the LMS can be applied