Appendix K

Gradient Adaptive Step Size (GASS) Algorithms in $\mathbb{R}$

Since its introduction, some 50 years ago, the Least Mean Square (LMS) algorithm [308] has become the most frequently used algorithm for the training of adaptive finite impulse response (FIR) filters. The LMS minimises the cost function $J(k) = \frac{1}{2}e^2(k)$, and is given by [308]

$$e(k) = d(k) - \mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k) \tag{K.1}$$

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\mathbf{x}(k) \tag{K.2}$$

where $e(k)$ denotes the instantaneous error at the output of the filter, $d(k)$ is the desired signal, $\mathbf{x}(k) = [x(k-1), \ldots, x(k-N)]^{\mathrm{T}}$ is the input signal vector, N is the length of the filter, $(\cdot)^{\mathrm{T}}$ is the vector transpose operator, and $\mathbf{w}(k) = [w_1(k), \ldots, w_N(k)]^{\mathrm{T}}$ is the filter coefficient (weight) vector. The parameter μ is the stepsize (learning rate), which is critical to the performance, and defines how fast the algorithm is converging.

Analysis and practical experience have shown that the presence of inputs with rich dynamics and ill-conditioned input correlation matrix often leads to divergence of LMS. One modification of LMS is the normalised LMS (NLMS) algorithm [113, 308], for which the adaptive learning rate is given by

$$\eta(k) = \frac{\mu}{\| \mathbf{x}(k) \|_2^2 + \varepsilon} \tag{K.3}$$

and the term ε is included to prevent divergence for close to zero input vector $\mathbf{x}(k)$.

The performance of both LMS and NLMS is affected by the inclusion of the 'independence' assumptions in their derivation. To make this class of algorithms more robust and faster converging, a number of gradient adaptive step size (GASS) algorithms have been proposed, which include those by Benveniste [24], Mathews [207], and Farhang [13]. The aim is to ensure $dJ/d\mu = 0$, which leads to $\mu(\infty) = 0$ in the steady state. Another way of introducing more robustness into the LMS update is to make the compensation term ε in the denominator of the NLMS stepsize (Equation K.3) adaptive. One such algorithm is the Generalised Normalised Gradient Descent (GNGD) algorithm [180].

Complex Valued Nonlinear Adaptive Filters: Noncircularity, Widely Linear and Neural Models
Danilo Mandic and Vanessa Su Lee Goh

K.1 Gradient Adaptive Stepsize Algorithms Based on $\partial J/\partial \mu$

A gradient adaptive learning rate $\mu(k)$ can be introduced into the LMS algorithm (Equation K.2), based on

$$\mu(k+1) = \mu(k) - \rho \nabla_\mu J(k)_{|\mu=\mu(k-1)} \tag{K.4}$$

where parameter ρ denotes the stepsize. The gradient $\nabla_\mu J(k)_{|\mu=\mu(k-1)}$ can be evaluated as

$$\nabla_\mu J(k) = \frac{1}{2}\frac{\partial e^2(k)}{\partial e(k)}\frac{\partial e(k)}{\partial y(k)}\frac{\partial y(k)}{\partial \mathbf{w}(k)}\frac{\partial \mathbf{w}(k)}{\partial \mu(k-1)} = -e(k)\mathbf{x}^{\mathrm{T}}(k)\frac{\partial \mathbf{w}(k)}{\partial \mu(k-1)} \tag{K.5}$$

Denote $\boldsymbol{\gamma}(k) = \partial \mathbf{w}(k)/\partial \mu(k-1)$ to obtain a general update for the stepsize in the form

$$\mu(k+1) = \mu(k) + \rho e(k)\mathbf{x}^{\mathrm{T}}(k)\boldsymbol{\gamma}(k) \tag{K.6}$$

Algorithms within this class differ, depending on the way they evaluate the term $\boldsymbol{\gamma}(k)$.

Benveniste's update [24] is rigorous and evaluates the sensitivity $\boldsymbol{\gamma}(k)$ based on (K.2) as

$$\begin{aligned}\frac{\partial \mathbf{w}(k)}{\partial \mu(k-1)} = \frac{\partial \mathbf{w}(k-1)}{\partial \mu(k-1)} &+ e(k-1)\mathbf{x}(k-1) + \mu(k-1)\frac{\partial e(k-1)}{\partial \mu(k-1)}\mathbf{x}(k-1) \\ &+ \mu(k-1)e(k-1)\frac{\partial \mathbf{x}(k-1)}{\partial \mu(k-1)}\end{aligned} \tag{K.7}$$

The last term in Equation (K.7) vanishes, since the input $\mathbf{x}(k-1)$ is independent of the learning rate $\mu(k-1)$, whereas the term $\partial e(k-1)/\partial \mu(k-1)$ becomes

$$\frac{\partial e(k-1)}{\partial \mu(k-1)} = \frac{\partial \left(d(k-1) - \mathbf{x}^{\mathrm{T}}(k-1)\mathbf{w}(k-1)\right)}{\partial \mu(k-1)} = -\mathbf{x}^{\mathrm{T}}(k-1)\frac{\partial \mathbf{w}(k-1)}{\partial \mu(k-1)} \tag{K.8}$$

The expression[1] for the gradient $\nabla_\mu J(k)$ in (K.5) now becomes

$$\begin{aligned}\nabla_{\mu(k-1)} J(k) &= -e(k)\mathbf{x}^{\mathrm{T}}(k)\boldsymbol{\gamma}(k) \\ \boldsymbol{\gamma}(k) &= \big[\underbrace{\mathbf{I} - \mu(k-1)\mathbf{x}(k-1)\mathbf{x}^{\mathrm{T}}(k-1)}_{\text{filtering term}}\big]\boldsymbol{\gamma}(k-1) + e(k-1)\mathbf{x}(k-1)\end{aligned} \tag{K.9}$$

The term in the square brackets represents a time-varying adaptive filter, which provides low pass filtering of the instantaneous gradient $e(k-1)\mathbf{x}(k-1)$. This way, Benveniste's update is very accurate and robust to the uncertainties due to the noisy instantaneous gradients $e(k-1)\mathbf{x}(k-1)$.

[1] For a small value of μ, assume $\mu(k-1) \approx \mu(k)$ and therefore

$$\frac{\partial \mathbf{w}(k)}{\partial \mu(k-1)} \approx \frac{\partial \mathbf{w}(k)}{\partial \mu(k)} = \boldsymbol{\gamma}(k)$$

Algorithm by Farhang and Ang [13] is based on a recursive calculation of $\boldsymbol{\gamma}$ from Equation (K.9) in the form

$$\boldsymbol{\gamma}(k) = \alpha\boldsymbol{\gamma}(k-1) + e(k-1)\mathbf{x}(k-1), \quad 0 \leq \alpha \leq 1 \tag{K.10}$$

that is, a time varying instantaneous filtering term in the square brackets in Equation (K.9) is replaced by a lowpass filter with a fixed coefficient α. For each weight update $w_j(k)$, we then have

$$\gamma_j(k) = \alpha\gamma_j(k-1) + e(k-1)x_j(k-1)$$

Mathews' algorithm [207] is a simplification of the algorithm by Farhang and Ang, where $\alpha = 0$, that is, it uses noisy instantaneous estimates of the gradient, resulting in the learning rate update

$$\mu(k+1) = \mu(k) + \rho e(k)e(k-1)\mathbf{x}^{\mathrm{T}}(k)\mathbf{x}(k-1) \tag{K.11}$$

K.2 Variable Stepsize Algorithms Based on $\partial J/\partial \varepsilon$

The generalised normalised gradient descent (GNGD) algorithm [180] makes the regularisation factor ε within the NLMS algorithm (Equation K.3) gradient adaptive, and is based on

$$\varepsilon(k+1) = \varepsilon(k) - \rho\nabla_{\varepsilon(k-1)}J(k)_{|\varepsilon=\varepsilon(k-1)} \tag{K.12}$$

Similarly to the GASS algorithms based on $\partial J/\partial\mu$, the gradient $\nabla_{\varepsilon(k-1)}J(k)_{|\varepsilon=\varepsilon(k-1)}$ can be evaluated as

$$\frac{\partial J(k)}{\partial\varepsilon(k-1)} = \frac{\partial J(k)}{\partial e(k)}\frac{\partial e(k)}{\partial y(k)}\frac{\partial y(k)}{\partial \mathbf{w}(k)}\frac{\partial \mathbf{w}(k)}{\partial\varepsilon(k-1)} = -e(k)\mathbf{x}^{\mathrm{T}}(k)\frac{\partial \mathbf{w}(k)}{\partial\varepsilon(k-1)} \tag{K.13}$$

The partial derivative $\partial\mathbf{w}(k)/\partial\varepsilon(k-1)$ in Equation (K.13) now becomes

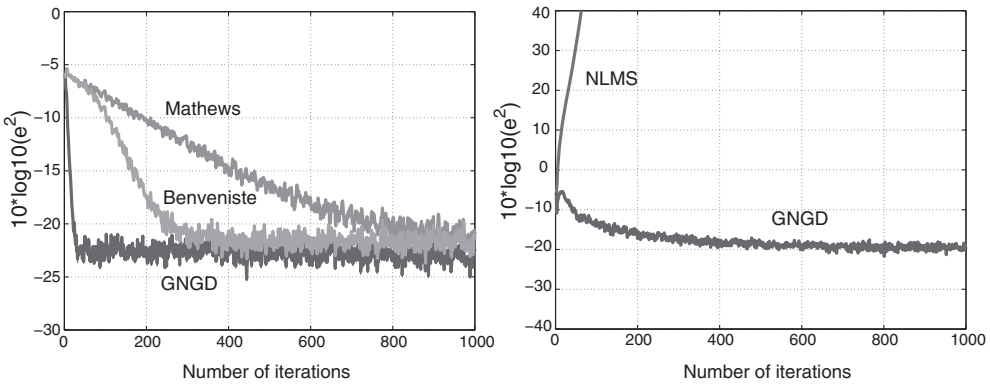
$$\begin{aligned}\frac{\partial \mathbf{w}(k)}{\partial\varepsilon(k-1)} &= \frac{\partial \mathbf{w}(k-1)}{\partial\varepsilon(k-1)} - \frac{\eta e(k-1)\mathbf{x}(k-1)}{\left(\parallel \mathbf{x}(k-1)\parallel_2^2 + \varepsilon(k-1)\right)^2} + \frac{\partial e(k-1)}{\partial\varepsilon(k-1)}\eta(k-1)\mathbf{x}(k-1)\\ \frac{\partial e(k-1)}{\partial\varepsilon(k-1)} &= \frac{\partial\left[d(k-1) - \mathbf{x}^T(k-1)\mathbf{w}(k-1)\right]}{\partial\varepsilon(k-1)} = -\mathbf{x}^T(k-1)\frac{\partial \mathbf{w}(k-1)}{\partial\varepsilon(k-1)}\end{aligned} \tag{K.14}$$

to give

$$\frac{\partial \mathbf{w}(k)}{\partial\varepsilon(k)} = \left[\mathbf{I} - \eta(k-1)\mathbf{x}(k-1)\mathbf{x}^{\mathrm{T}}(k-1)\right]\frac{\partial \mathbf{w}(k-1)}{\partial\varepsilon(k-1)} - \frac{\eta e(k-1)\mathbf{x}(k-1)}{\left(\parallel \mathbf{x}(k-1)\parallel_2^2 + \varepsilon(k-1)\right)^2} \tag{K.15}$$

For simplicity, denote $\boldsymbol{\gamma}(k) = \partial\mathbf{w}(k)/\partial\varepsilon(k)$, to yield the update of the regularisation factor in the form

$$\varepsilon(k+1) = \varepsilon(k) + e(k)\mathbf{x}^{\mathrm{T}}(k)\boldsymbol{\gamma}(k) \tag{K.16}$$



(a) Comparison of GASS algorithms (b) Comparison of GNGD and NLMS

Figure K.1 Learning curves for the GASS algorithms. Left: comparison between Benveniste's, Farhang's, Mathews' algorithms and GNGD for the prediction of the coloured signal (Equation 8.22). Right: Comparison between GNGD and NLMS for prediction of the nonlinear signal (Equation 8.24), for the stepsize $\mu = 2.1$

In the standard GNGD algorithm [180], the term in the square brackets in Equation (K.15) is set to zero, giving

$$\varepsilon(k+1) = \varepsilon(k) - \rho\mu \frac{e(k)e(k-1)\mathbf{x}^{\mathrm{T}}(k)\mathbf{x}(k-1)}{\left(\| \mathbf{x}(k-1) \|_2^2 + \varepsilon(k-1)\right)^2} \tag{K.17}$$

Figure K.1(a) shows learning curves for the GASS algorithms (based on $\partial J/\partial\mu$) and the GNGD algorithm (based on $\partial J/\partial\varepsilon$) for prediction of the coloured AR(4) process (See Equation 8.22). The GNGD exhibited fastest convergence, the algorithms by Benveniste and Farhang and Ang had similar performance, whereas Mathews' algorithm was slowest to converge. Figure K.1(b) illustrates the excellent stability of GNGD compared with NLMS, for the prediction of nonlinear signal (See Equation 8.24). The learning rate was chosen to be $\mu = 2.1$, for which the NLMS diverged whereas GNGD converged.