# Transform Domain Adaptive Filters

### *Self-Orthogonalizing Adaptive Filters*

We would like to improve the convergence properties of the LMS algorithm when the input is highly correlated. We know that highly correlated input implies a large eigenvalue spread which in turn causes LMS to convergence slowly.

Our approach will be to modify the input signal to the LMS adaptive filter so that the eigenvalue spread is reduced. Of course this modification will come at the expense of increased computation.

To show that it *is* possible to design an algorithm in which the convergence rate is constant irrespective of the input statistics, we consider the self-orthogonalizing adaptive filtering algorithm (Newton's method)

$$\hat{\mathbf{w}}(n+1) = \hat{\boldsymbol{\omega}}(\nu) + \alpha \mathbf{P}^{-1}\boldsymbol{\upsilon}(\nu)\varepsilon(\nu)$$

where $0 < \alpha < 1$ and usually set to the value $\alpha = \dfrac{1}{2M}$, **R** is the correlation matrix of the input vector **u**($n$), and $e(n)$ is the error signal. We rewrite the above filter update in terms of the misalignment vector [ $\varepsilon(n) = \hat{\mathbf{w}}(n) - \mathbf{w}_o$ ]

$$\hat{\mathbf{w}}(n+1) - \boldsymbol{\omega}_o = \hat{\boldsymbol{\omega}}(\nu) - \boldsymbol{\omega}_o + \alpha\mathbf{P}^{-1}\boldsymbol{\upsilon}(\nu)[\delta(\nu) - \hat{\boldsymbol{\omega}}^T(\nu)\boldsymbol{\upsilon}(\nu)]$$

or

$$\begin{aligned}
\varepsilon(n+1) &= \varepsilon(n) + \alpha\mathbf{R}^{-1}\mathbf{u}(n)d(n) - \alpha\mathbf{R}^{-1}\mathbf{u}(n)\mathbf{u}^T(n)\hat{\mathbf{w}}(n) \\
&= \varepsilon(n) + \alpha\mathbf{R}^{-1}\mathbf{u}(n)d(n) - \alpha\mathbf{R}^{-1}\mathbf{u}(n)\mathbf{u}^T(n)[\varepsilon(n) + \mathbf{w}_o] \\
&= [\mathbf{I} - \alpha\mathbf{R}^{-1}\mathbf{u}(n)\mathbf{u}^T(n)]\varepsilon(n) + \alpha\mathbf{R}^{-1}\mathbf{u}(n)[d(n) - \mathbf{u}^T(n)\mathbf{w}_o] \, . \\
&= [\mathbf{I} - \alpha\mathbf{R}^{-1}\mathbf{u}(n)\mathbf{u}^T(n)]\varepsilon(n) + \alpha\mathbf{R}^{-1}\mathbf{u}(n)e_o(n)
\end{aligned}$$

Applying the expectation operator to both sides of the above and invoking the independence assumption [$\hat{\mathbf{w}}(n)$ is independent of **u**($n$)] and the orthogonality principle $E[\mathbf{u}(n)\varepsilon_o(\nu)] = \mathbf{0}$, we have

$$\begin{aligned}
E[\varepsilon(\nu+1)] &= \{\mathbf{I} - \alpha\mathbf{P}^{-1}E[\boldsymbol{\upsilon}(\nu)\boldsymbol{\upsilon}^T(\nu)]\}E[\varepsilon(\nu)] + \alpha\mathbf{P}^{-1}E[\boldsymbol{\upsilon}(\nu)\varepsilon_o(\nu)] \\
&= (1-\alpha)E[\varepsilon(\nu)] \qquad . \\
&= (1-\alpha)^{\nu}E[\varepsilon(0)]
\end{aligned}$$

For $0 < \alpha < 1$, we have convergence in the mean

$$\lim_{n\to\infty} E[\varepsilon(\nu)] = \mathbf{0}$$

or equivalently,

$$\lim_{n\to\infty} E[\hat{\boldsymbol{\omega}}(\nu)] = \boldsymbol{\omega}_o \, .$$

We note that our convergence rate is *independent* of the input statistics.

For LMS we had previously seen

$$E[\varepsilon(\nu+1)] = (\mathbf{I} - \mu\mathbf{P})^{\nu+1} E[\varepsilon(0)]$$
$$\Leftrightarrow$$
$$E[\omega(\nu+1)] = (\mathbf{I} - \mu\Lambda)^{\nu+1} E[\omega(0)]$$

where the last line is under unitary transformation.

---

**Example**. Consider the case of white Gaussian noise with correlation matrix $\mathbf{R} = \sigma^2\mathbf{I}$. Adaptive adjustment using the self-orthogonalizing adaptive filtering algorithm yields ($\alpha = 1 / (2M)$)

$$\begin{aligned}\hat{\mathbf{w}}(n+1) &= \hat{\boldsymbol{\omega}}(\nu) + \alpha\mathbf{P}^{-1}\boldsymbol{\upsilon}(\nu)\varepsilon(\nu) \\ &= \hat{\boldsymbol{\omega}}(\nu) + \frac{1}{2M\sigma^2}\boldsymbol{\upsilon}(\nu)\varepsilon(\nu)\end{aligned}$$

which we recognize as LMS with $\mu = \dfrac{1}{2M\sigma^2}$. This make sense since for the white Gaussian noise input (eigenvalue spread of unity), LMS converges as fast as possible ($\chi = 1$) and no transformation can further speed the convergence rate.

---

Our strategy to build a self-orthogonalizing adaptive filter will follow two steps

Step 1: The input vector $\mathbf{u}(n)$ is transformed into a vector of uncorrelated variables (eigenvalue spread of unity).

Step 2: The transformed input vector is used as input to an LMS algorithm.

Note that a vector of uncorrelated variables has a correlation matrix which is diagonal.

---

**The General Algorithm**

**Step 1:** Consider the unitary transform **Q** (columns of **Q** are eigenvectors of **R**) applied to the input vector

$$\mathbf{v}(n) = \boldsymbol{\Theta}^T\boldsymbol{\upsilon}(\nu).$$

Applied this way, this unitary transformation is known as the Karhunen-Loeve Transform (KLT). The correlation matrix of $\mathbf{v}(n)$ can be written as

$$\begin{aligned}E[\mathbf{v}(n)\boldsymbol{\varpi}^T(\nu)] &= E[\boldsymbol{\Theta}^T\boldsymbol{\upsilon}(\nu)\boldsymbol{\upsilon}^T(\nu)\boldsymbol{\Theta}] \\ &= \boldsymbol{\Theta}^T\mathbf{P}\boldsymbol{\Theta} \\ &= \Lambda\end{aligned}$$

where

$$\Lambda = \mathrm{diag}[\lambda_0 \quad \mathrm{L} \quad \lambda_{M-1}].$$

Under KLT, our outputs form a sequence of uncorrelated variables (Step 1).

We have

$$\begin{aligned}\hat{\mathbf{w}}(n+1) &= \hat{\boldsymbol{\omega}}(\nu) + \alpha\mathbf{P}^{-1}\boldsymbol{\upsilon}(\nu)\varepsilon(\nu) \\ &= \hat{\boldsymbol{\omega}}(\nu) + \alpha\boldsymbol{\Theta}\Lambda^{-1}\boldsymbol{\Theta}^T\boldsymbol{\upsilon}(\nu)\varepsilon(\nu) \\ &= \hat{\boldsymbol{\omega}}(\nu) + \alpha\boldsymbol{\Theta}\Lambda^{-1}\boldsymbol{\varpi}(\nu)\varepsilon(\nu)\end{aligned}$$

assuming application of KLT to the input vector.

**Step 2:** The above filter coefficients may be written in terms of the transformed filter coefficients

$$
\begin{aligned}
\hat{\mathbf{w}}(n+1)' &= \mathbf{\Theta}^T \hat{\boldsymbol{\omega}}(\nu+1) \\
&= \mathbf{\Theta}^T \hat{\boldsymbol{\omega}}(\nu) + \alpha \mathbf{\Theta}^T \mathbf{\Theta} \Lambda^{-1} \boldsymbol{\varpi}(\nu)\varepsilon(\nu) \\
&= \hat{\boldsymbol{\omega}}(\nu)' + \alpha \Lambda^{-1} \boldsymbol{\varpi}(\nu)\varepsilon(\nu)
\end{aligned}
$$

for which the *k*th element is

$$
\hat{w}_k(n+1)' = \hat{\omega}_\kappa(\nu+1)' + \frac{\alpha}{\lambda_\kappa} \varpi_\kappa(\nu)\varepsilon(\nu) .
$$

We recognize three important elements of the update equation for the transform-domain adaptive filter

1) the input has undergone a unitary transformation, $v_k(n) = \boldsymbol{\theta}_\kappa^T \boldsymbol{\upsilon}(\nu)$ (KLT)
2) the correction to each filter coefficient is scaled by the corresponding eigenvalue.
3) the coefficient update is a form of LMS (Step 2)

Notes:

1) If we are interested in the value of $\hat{\mathbf{w}}(n)$ (as is the case in a system modeling configuration), we must apply the inverse transform

$$
\hat{\mathbf{w}}(n) = \mathbf{\Theta} \hat{\boldsymbol{\omega}}(\nu)' .
$$

2) The definition of the error signal, $e(n)$ is not effected by the transformation:

$$
\begin{aligned}
e(n) &= d(n) - y(n) \\
&= \mathbf{w}_{opt}^T \mathbf{u}(n) - \hat{\mathbf{w}}^T(n)\mathbf{u}(n) \\
&= [\mathbf{w}_{opt} - \hat{\mathbf{w}}(n)]^T \mathbf{u}(n) \\
&= [\mathbf{w}_{opt} - \hat{\mathbf{w}}(n)]^T \mathbf{Q}^T \mathbf{Q} \mathbf{u}(n) \\
&= [\mathbf{w}'_{opt} - \hat{\mathbf{w}}'(n)]^T \mathbf{v}(n) \\
&= \mathbf{w}'^T_{opt} \mathbf{v}(n) - \hat{\mathbf{w}}'^T(n)\mathbf{v}(n) \\
&= d'(n) - y'(n)
\end{aligned}
$$