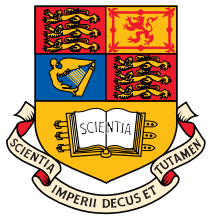
Spectrum Estimation & Adaptive SP

Lecture 7: Complex–Valued Adaptive Filters

Danilo Mandic

room 813, ext: 46271



Department of Electrical and Electronic Engineering
Imperial College London, UK

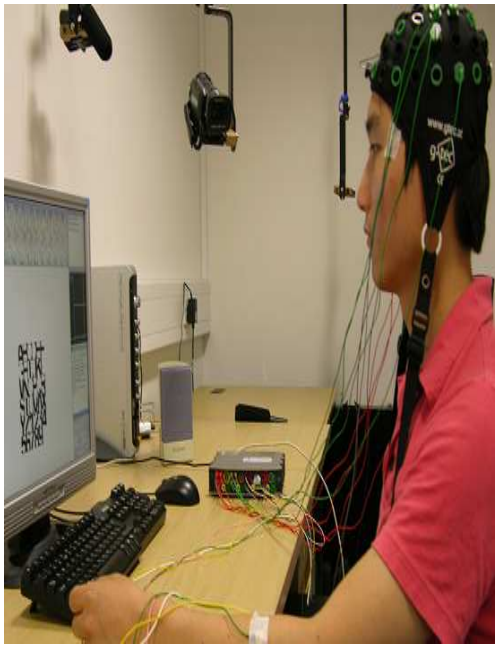
d.mandic@imperial.ac.uk, URL: www.commsp.ee.ic.ac.uk/∼mandic

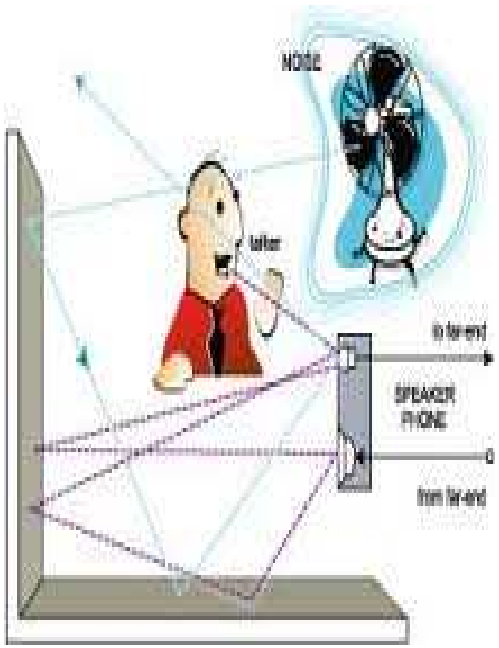
Outline:

Really, to de-mystify and make rigorous several concepts

- Recursive solution to the Wiener filter $\looparrowright$ Recursive Least Squares
- Multidimensional and multichannel sensors – a unified approach to adaptive filtering of such signals
- Circularity – a unique signature of bivariate signals $\looparrowright$ second order circularity (properness) $\looparrowright$ augmented complex statistics
- Duality between the processing in $\mathbb{R}^2$ and $\mathbb{C}$ (isomorphism)
- The issue of complex gradient $\looparrowright$ $\mathbb{CR}$ calculus
- Covariance, pseudocovariance, and widely linear models
- Complex least mean square (CLMS) and augmented CLMS (ACLMS)
- Multivariate adaptive filters (any number of data channels)
- Applications: communications, radar and sonar, target tracking, renewable energy, smart grid

Where can we apply multidimensional (multichannel) adaptive filters?





Brain Computer Interface

Decoding brain activity

to control computers

Spect. Est., ASP

Audio applications

Echo cancellation

Multiple mics and speakers

Spect. Est,

Radar and sonar

Trajectory tracking

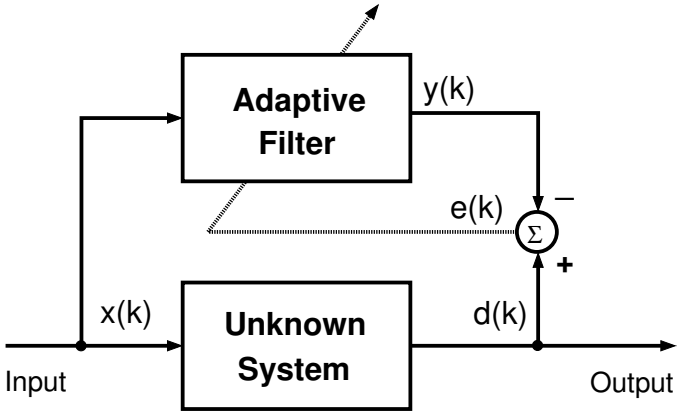
Radar, sonar: Manouver prediction

Spect. Est., ASP.

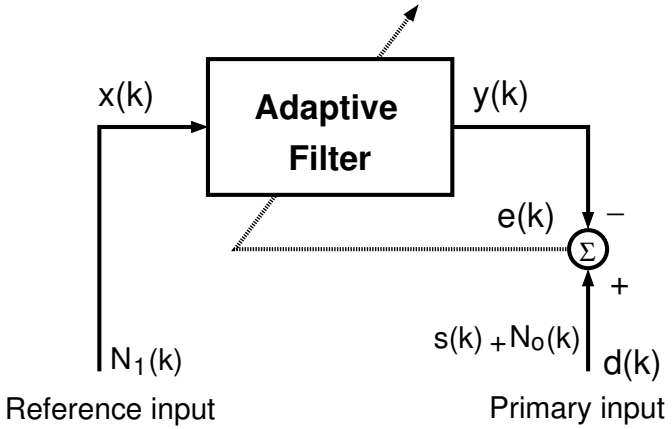
Also: seismics and oil industry, finance, social groups behaviour, economics

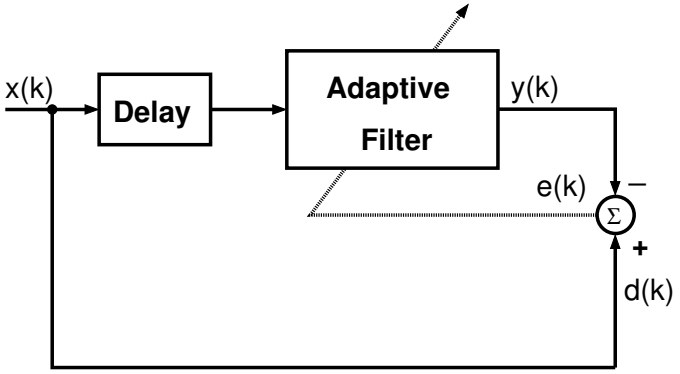
And we can use multidimensional filters within our usual adaptive filtering configurations!

System Identification

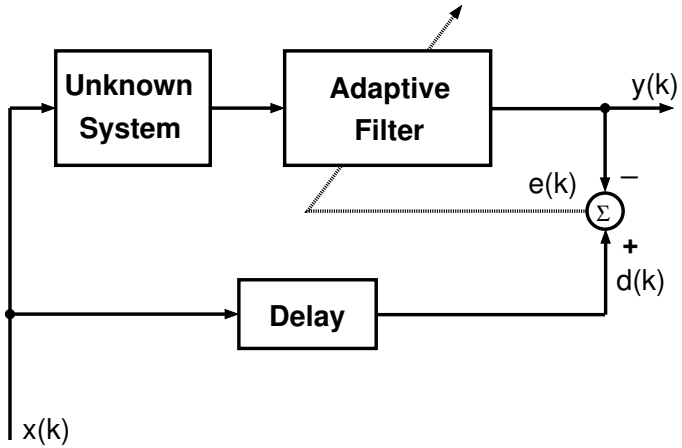


Noise Cancellation





Adaptive Prediction



Inverse System Modelling

Let us first look at the big picture:
There are two main families of adapt. filt. algorithms

- **Gradient descent based methods** (e.g. the Least Mean Square (LMS)) provide a recursive form of an **approximate minimisation** of the mean square error (MSE), where the cost function

$$J \sim E\{e^2(n)\} \quad \text{for LMS} \quad J(n) \sim e^2(n) \qquad \textbf{stochastic}$$

  Knowledge of the autocorrelation of the input process and cross-correlation between the input an teaching signal required.

  Rapid convergence or sufficiently small excess MSE not guaranteed.

- **Least squares (LS) techniques** are based on the **exact minimisation** of a sum of instantaneous squared errors

$$J(n) = \sum_{i=0}^{n} e^2(i) = \sum_{i=0}^{n} \left| d(i) - \mathbf{x}^T(i)\mathbf{w}_n \right|^2 \qquad \textbf{deterministic}$$

  Deterministic cost function $\looparrowright$ no statistical information about $\mathbf{x}$ and d!

☞ The LS error depends on particular $x(n)$ and $d(n)$ $\looparrowright$ for different signals we obtain different filters.

☞ The MSE does not depend on particular $x(n)$ and $d(n)$, just on their statistics $\looparrowright$ produces equal weights for signals with the same statistics.

The Recursive Least Squares (RLS) algorithm

(recursive solution to the "deterministic" Wiener filtering problem)

Aim: For a filter of order N, find the weight vector $\mathbf{w}_n = \mathbf{w}_{opt}(n)$ which minimizes the sum of squares of output errors up until the time instant n

$$\mathbf{w}_n = \mathbf{w}_{opt}^{LS} \text{ over n time instants} \quad \Rightarrow \quad \mathbf{w}_n = arg \ \min_{\mathbf{w}} \sum_{i=0}^{n} \left| d(i) - \mathbf{x}^T(i)\mathbf{w}_n \right|^2$$

(the summation over i is performed for the **latest** set of coefficients $\mathbf{w}_n$)

☞ Notice, the weights $\mathbf{w}_n$ are **held constant** over the whole observation $[0, n]$ – similar to the Wiener setting (but a deterministic cost function)

To find $\mathbf{w}_n$: set $\nabla_{\mathbf{w}} J(n) = \mathbf{0}$ to solve for $\mathbf{w}_{opt}(n) = \mathbf{w}_n = \mathbf{R}^{-1}(n)\mathbf{p}(n)$

- **Careful here, as:** $\mathbf{R}(n) = \sum_{i=0}^{n} \mathbf{x}(i)\mathbf{x}^T(i), \quad \mathbf{p}(n) = \mathbf{r}_{d\mathbf{x}} = \sum_{i=0}^{n} d(i)\mathbf{x}(i)$

that is, $\mathbf{R}(n)$, $\mathbf{p}(n)$ are purely ***deterministic*** (*cf.* $\mathbf{R} = E\{\mathbf{x}\mathbf{x}^T\}, \mathbf{p} = E\{d\mathbf{x}\}$ in Wiener filt.)

Recursive least squares (RLS). The aim is to recursively update:

(a) $J(n+1) = J(n) + |e(n+1)|^2$ (b) $\mathbf{p}(n+1) = \mathbf{p}(n) + d(n+1)\mathbf{x}(n+1)$

(c) $\mathbf{R}(n+1) = \mathbf{R}(n) + \mathbf{x}(n+1)\mathbf{x}^T(n+1)$

The weight update: $\mathbf{w}_{n+1} = \mathbf{w}_{opt}(n+1) = \mathbf{w}(n+1) = \mathbf{R}^{-1}(n+1)\mathbf{p}(n+1)$

Ways to compute a matrix inverse

while direct, this technique is computationally wasteful, $N^3 + 2N^2 + N$ multiplic.

There are many ways (Neumann series, geometric series). For $|a| < 1$

$$\sum_{i=0}^{\infty} a^i = \frac{1}{1-a}, \qquad \text{now swap} \quad \mathrm{b} = 1 - \mathrm{a} \quad \text{to get} \quad \sum_{\mathrm{i}=1}^{\infty} (1-\mathrm{b})^{\mathrm{i}} = \frac{1}{\mathrm{b}}$$

☞ **we have found the inverse of b via a series in $1-b$.**

Generalization to matrices: Consider a nonsingular matrix $\mathbf{A}$, to yield

$$\lim_{i=\to\infty} (\mathbf{I} - \mathbf{A})^i = \mathbf{0} \quad \Rightarrow \quad \mathbf{A}^{-1} = \sum_{i=0}^{\infty} (\mathbf{I} - \mathbf{A})^i$$

Then, for an invertible matrix $\mathbf{X}$, and its inverse $\mathbf{A}$, we can write

$$\lim_{i\to\infty} \left(\mathbf{I}-\mathbf{X}^{-1}\mathbf{A}\right)^i = \lim_{i\to\infty} \left(\mathbf{I}-\mathbf{A}\mathbf{X}^{-1}\right)^i = \mathbf{0} \quad \Rightarrow \quad \mathbf{A}^{-1} = \sum_{i=0}^{\infty} \Big(\mathbf{X}^{-1}\big(\mathbf{X}-\mathbf{A}\big)\Big)^i \mathbf{X}^{-1}$$

If $(\mathbf{A}-\mathbf{X})$ has rank 1, we finally have: $\mathbf{A}^{-1} = \mathbf{X}^{-1} + \dfrac{\mathbf{X}^{-1}\big(\mathbf{X}-\mathbf{A}\big)\mathbf{X}^{-1}}{1-\mathrm{tr}\Big(\mathbf{X}^{-1}\big(\mathbf{X}-\mathbf{A}\big)\Big)}$

The RLS formulation $\looparrowright$ $\mathbf{R}^{-1}$ calculation is $\mathcal{O}(N^3)$

(we use the standard matrix inversion lemma, also known as Woodbury's identity)

Key to RLS: all we have to do is to employ a recursive update of $\mathbf{R}^{-1}$

Start from the matrix inversion lemma (also known as ABCD lemma)

$$(A+BCD)^{-1} = A^{-1} - A^{-1}B\left(DA^{-1}B + C^{-1}\right)^{-1}DA^{-1}$$

and set $A = \mathbf{R}(n), B = \mathbf{x}(n+1), C = 1, D = \mathbf{x}^T(n+1)$, to give

$$\mathbf{R}(n+1) = \mathbf{R}(n) + \mathbf{x}(n+1)\mathbf{x}^T(n+1) = A + BCD$$

Then $\mathbf{R}^{-1}(n+1)$ is given by

$$\mathbf{R}^{-1}(n+1) = \mathbf{R}^{-1}(n) - \frac{\mathbf{R}^{-1}(n)\mathbf{x}(n+1)\mathbf{x}^T(n+1)\mathbf{R}^{-1}(n)}{\mathbf{x}^T(n+1)\mathbf{R}^{-1}(n)\mathbf{x}(n+1)+1} \quad \looparrowright \quad \mathcal{O}(N^2)$$

$\looparrowright$ **we never compute $\mathbf{R}(n+1)$ or $\mathbf{R}^{-1}(n)$ directly, but recursively**

The optimal RLS weight vector: $\mathbf{w}_{opt}(n+1) = \mathbf{w}(n+1) = \mathbf{R}^{-1}(n+1)\mathbf{p}(n+1)$

Minimum LSE: $J_{min}^{LS} = \| \mathbf{d}(n) \|_2^2 - \mathbf{r}_{dx}^T(n)\mathbf{w}_n$

where $\mathbf{r}_{dx}(n) = \mathbf{p}(n), \mathbf{d}(n) = [d(0), \ldots, d(n)]^T$.

RLS: Adaptive noise cancellation

The system has two microphones (primary & reference) *`rlsdemo` in Matlab

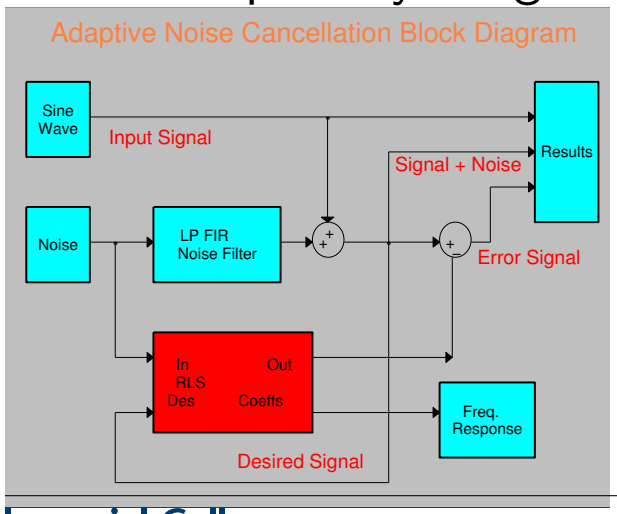
Similar scenario to noise-cancelling headphones.

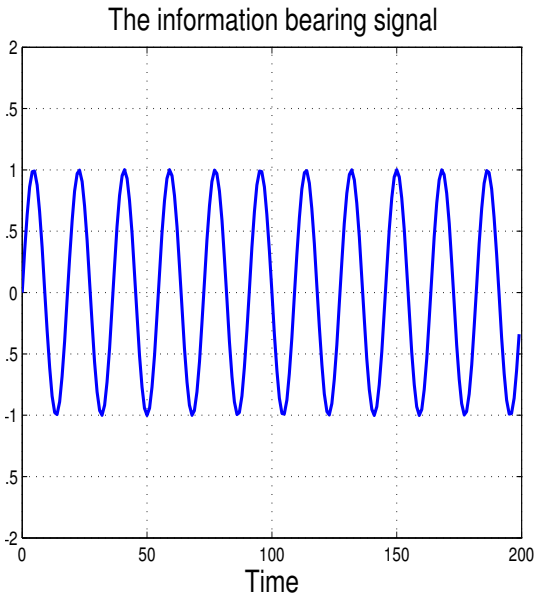
Primary signal:

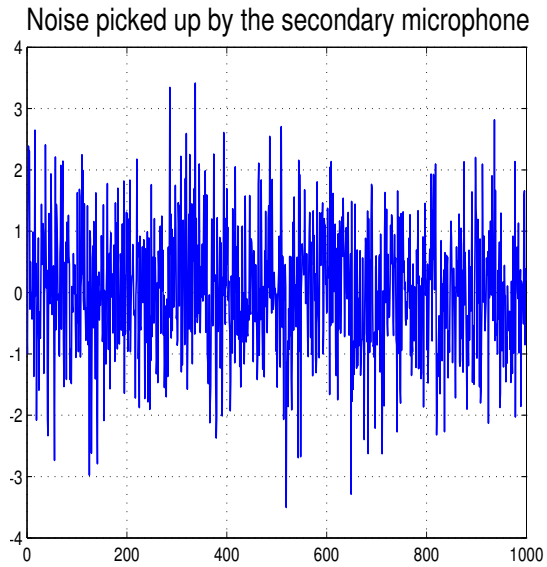
$$\sin(n) + q(n)$$

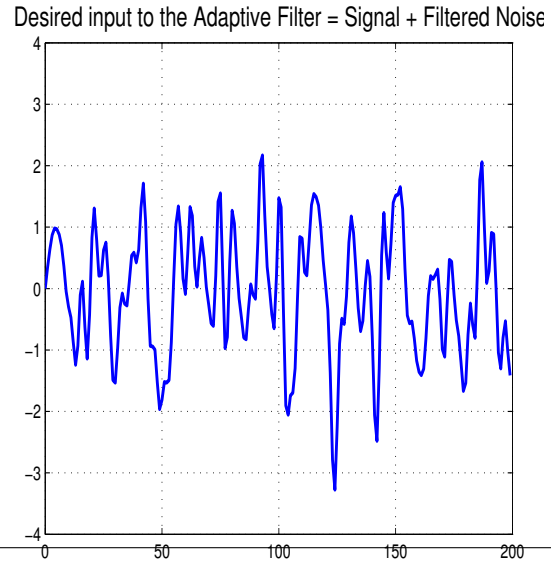
Reference signal:

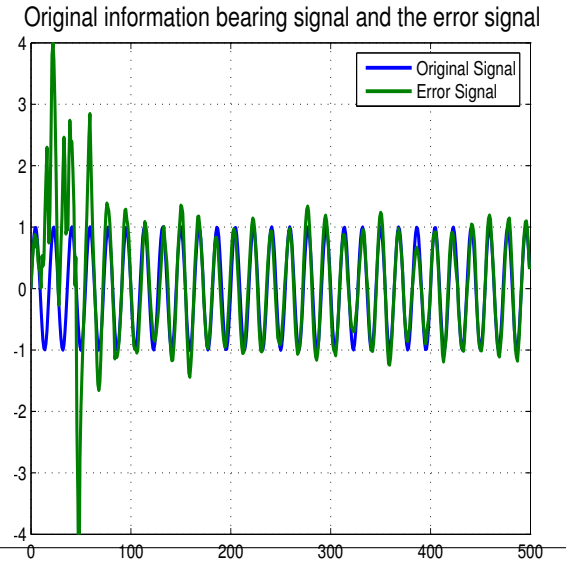
any noise correlated with the noise $q(n)$ in the primary signal









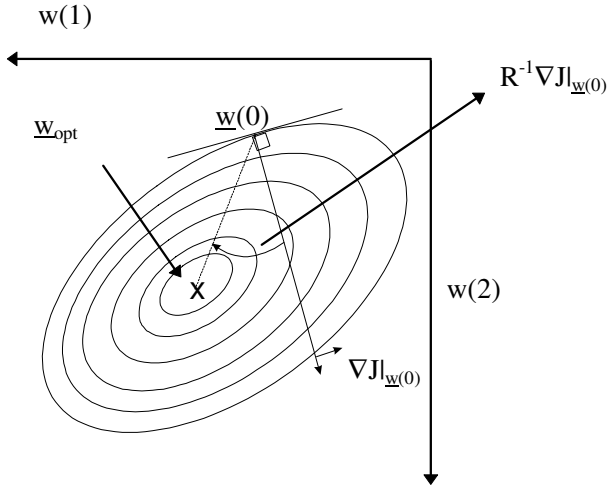


Some things to remember about RLS

- Results are exactly the same as for the normal least squares $\looparrowright$ no approximation involved
- RLS does not perform matrix inversion at any stage $\looparrowright$ it calculates the matrix inverse recursively
- Unlike gradient algorithms, there is no explicit notion of learning rate
- RLS guarantees convergence to the optimal weight vector
- Variants of RLS (sliding window, forgetting factor) deal with pragmatic issues (nonstationarity etc)
- The price to pay is increased computational complexity compared to the LMS

Comparison between LMS & RLS

The role of $\mathbf{R}^{-1}$ in RLS is to rotate the direction of descent of the LMS algorithm towards the minimum of the cost function independent of the nature, or colouration, of the input.



The operation $\mathbf{R}^{-1}\mathbf{x}[k]$ is a **pre-whitening operation**, compare with transform domain adaptive filtering.

The convergence of the RLS in a high SNR environment ($> 10dB$) is of an order $\mathcal{O}(2p)$, whereas for LMS $\mathcal{O}(10p)$.

The misadjustment performance of RLS is essentially zero because it is deterministic and matches the data where $\lambda = 1$.

Part 1: Fundamentals of Complex-Valued Adaptive Filters

Complex adaptive signal processing: Applications

Renewable Energy

2D and 3D anemometers

control of wind turbine

Body motion sensor

3D - position, gyroscope, speed

gait, biometrics

Wearable technologies

Biomechanics

virtual reality

Standard adaptive filtering algorithms in $\mathbb{C}$

Considered straightforward extensions of the corresponding algorithms in $\mathbb{R}$ – replace the vector transpose by the Hermitian transpose:

- Covariance

$$\mathcal{C} = E\{\mathbf{x}\mathbf{x}^T\} \qquad \rightsquigarrow \qquad \mathcal{C} = E\{\mathbf{z}\mathbf{z}^H\}$$

- Autoregressive model and Wiener solution

$$\mathbf{w} = \mathbf{R}^{-1}\mathbf{p} \qquad \rightsquigarrow \qquad \mathbf{w}^* = \mathbf{R}^{-1}\mathbf{p}$$

- Least mean square (LMS) $\rightsquigarrow$ complex LMS [Widrow *at al.* 1975]

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\mathbf{x}(k) \qquad \rightsquigarrow \qquad \mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\mathbf{z}^*(k)$$

- Real time recurrent learning (RTRL) [Williams & Zipser 1989] $\rightsquigarrow$ complex RTRL [Goh & Mandic 2004]

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\mathbf{\Pi}(k) \qquad \rightsquigarrow \qquad \mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\mathbf{\Pi}^*(k)$$

This is however valid only for circular complex random processes

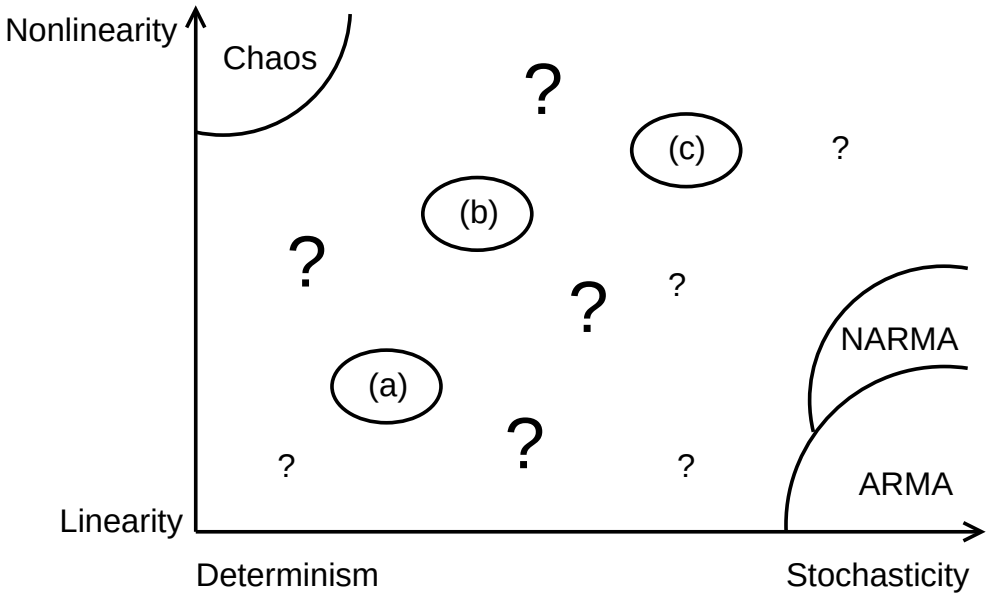
Wind sensors - 2D and 3D anemometers

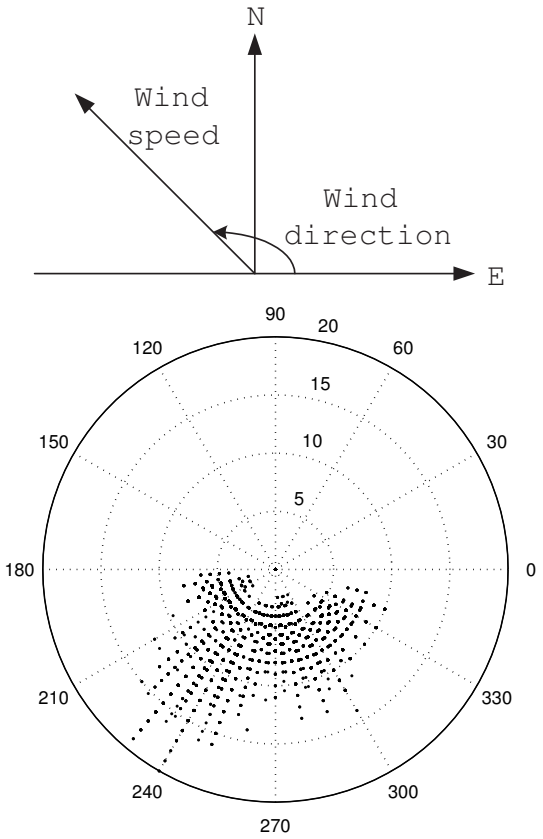
Property of division algebras $\looparrowright$ complex (non)circularity

Noncircularity of the wind distribution $v(k) = |v(k)|e^{\jmath\Phi(k)}$

Deterministic vs. Stochastic nature

Linear vs. Nonlinear nature





Change in signal modality can indicate e.g. health hazard (fMRI, HRV)

Real world signals are denoted by '???'

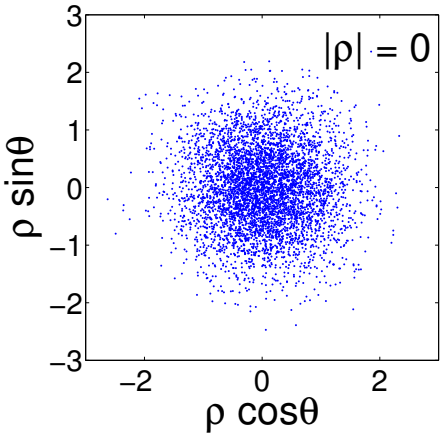
- $\exists$ a unique signature of complex signals?
- $\looparrowright$ **degree of noncircularity**

Circular vs noncircular complex random variables

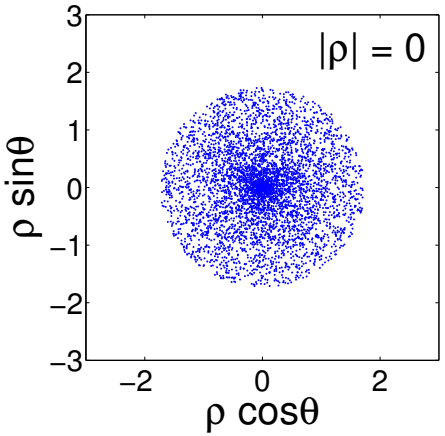
Circularity = Rotation invariant distribution $p(\rho, \theta) = p(\rho, \theta - \phi)$

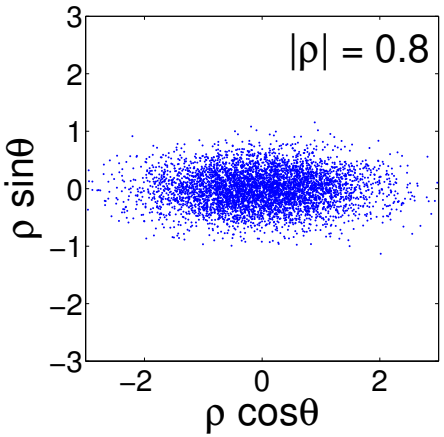
circular variable = construct $z = \rho\cos(\theta) + j\rho\sin(\theta)$, $\theta \sim \mathcal{U}[0, 2\pi)$, $\rho \sim$ any pdf

Gaussian circular

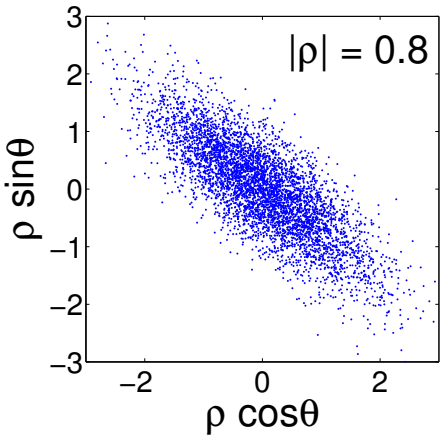


Uniform circular





Gaussian uncorrelated noncircular



Gaussian correlated noncircular

- Covariance of z is $c = E\{zz^*\}$
- Pseudocovariance is $p = E\{zz\}$
- Circularity quotient: $\rho = \frac{p}{c}$
- Circularity coefficient: $|\rho| = \frac{|p|}{c}$

Circularity quotient and coefficient quantify the **degree of non-circularity**

Recap: $\mathbb{CR}$-derivatives

Can we exploit results from Multivariate Calculus in $\mathbb{R}^2$?

GOAL: Find the derivative of a complex function $f(z)$ w.r.t. $z = x + \jmath y$. In standard Multivariate Calculus in $\mathbb{R}^{N\times 1}$ the derivative of a function $g(\mathbf{x})$, $\mathbf{x} = [x_1, x_2, \ldots, x_N]$ is defined as $\frac{\partial g}{\partial \mathbf{x}} = \left[\frac{\partial g}{\partial x_1}, \ldots, \frac{\partial g}{\partial x_N}\right]^T$

- **Step 1:** Define the vector $\mathbf{x} = [x, \jmath y]^T$, hence $z = \mathbf{1}^T\mathbf{x}$.
- **Step 2:** Express the derivative of f with respect to "real" vector $\mathbf{x}$ i.e $\frac{\partial f}{\partial \mathbf{x}} = \left[\frac{\partial f}{\partial x} \quad \frac{\partial f}{\jmath \partial y}\right]^T$
- **Step 3:** Transform derivative vector in Step 2 back into $\mathbb{C}$

$$\frac{\partial f}{\partial z} = \mathbf{1}^T \frac{\partial f}{\partial \mathbf{x}} = \frac{\partial f}{\partial x} + \frac{\partial f}{\jmath \partial y} = \frac{\partial f}{\partial x} - \jmath \frac{\partial f}{\partial y}$$

- **Step 4:** Normalise the derivative since f is "differentiated twice"

$\mathbb{R} - \text{der} : \frac{\partial f}{\partial z} = \frac{1}{2}\left[\frac{\partial f}{\partial x} - \jmath \frac{\partial f}{\partial y}\right]$. Similarly, $\mathbb{R}^* - \text{der} : \frac{\partial f}{\partial z^*} = \frac{1}{2}\left[\frac{\partial f}{\partial x} + \jmath \frac{\partial f}{\partial y}\right]$

Examples: $\mathbb{CR}$-derivatives

Prove these from the definitions of the $\mathbb{R}$ and $\mathbb{R}^*$ derivatives

For the $\mathbb{R}-$ derivative, the function is partially differeentaited w.r.t z while keeping z^* constant, vice versa for the $\mathbb{R}^*-$ derivative.

$f(z, z^*)$	$\mathbb{R}-$der	$\mathbb{R}^*-$der
z	1	0
z^*	0	1
$\lvert z\rvert^2 = zz^*$	z^*	z
$z^2 z^*$	$2\lvert z\rvert^2$	z^2
e^z	e^z	0

The derivative of a cost function $\frac{1}{2}e(k)e^*(k)$ and CLMS

As $\mathbb{C}$-derivatives are not defined for real functions of complex variable

$$\mathbb{R}-\text{der:}\quad \frac{\partial}{\partial \mathbf{z}}=\frac{1}{2}\left[\frac{\partial}{\partial \mathbf{x}}-\jmath\frac{\partial}{\partial \mathbf{y}}\right] \qquad \mathbb{R}^*-\text{der:}\quad \frac{\partial}{\partial \mathbf{z}^*}=\frac{1}{2}\left[\frac{\partial}{\partial \mathbf{x}}+\jmath\frac{\partial}{\partial \mathbf{y}}\right]$$

and the gradient

$$\nabla_{\mathbf{w}}J=\frac{\partial J(e,e^*)}{\partial \mathbf{w}}=\left[\frac{\partial J(e,e^*)}{\partial w_1},\ldots,\frac{\partial J(e,e^*)}{\partial w_N}\right]^T=2\frac{\partial J}{\partial \mathbf{w}^*}=\underbrace{\frac{\partial J}{\partial \mathbf{w}^r}+\jmath\frac{\partial J}{\partial \mathbf{w}^i}}_{pseudogradient}$$

The standard Complex Least Mean Square (CLMS) (Widrow *et al.* 1975)

$$\begin{aligned} y(k) &= \mathbf{w}^H(k)\mathbf{x}(k) \\ e(k)=d(k)-y(k) &\qquad e^*(k)=d^*(k)-\mathbf{x}^H(k)\mathbf{w}(k) \\ &\text{and}\quad \nabla_{\mathbf{w}}J=\nabla_{\mathbf{w}^*}J \\ \mathbf{w}(k+1) &= \mathbf{w}(k)-\mu\frac{\partial e(k)e^*(k)}{\partial \mathbf{w}^*(k)}=\mathbf{w}(k)+\mu e^*(k)\mathbf{x}(k) \end{aligned}$$

Thus, no need for tedious computations – The CLMS is derived in one line.

Orthogonality principle & alternative forms for the CLMS

Consider the expected value for the CLMS update

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu E\big[e(k)\mathbf{x}^*(k)\big]$$

It has converged when $\mathbf{w}(k+1) = \mathbf{w}(k) = \mathbf{w}(\infty)$ and thus the weight update $\Delta\mathbf{w}(k) = \mu E[e(k)\mathbf{x}^*(k)] = \mathbf{0}$.

This is achieved for $E\big[d^*(k)\mathbf{x}(k)\big] - E\big[\mathbf{x}(k)\mathbf{x}^H(k)\big]\mathbf{w}(k) = \mathbf{0} \Leftrightarrow \mathbf{R}\mathbf{w}_o = \mathbf{p}$ that is, for the Wiener solution, $\mathbf{w}_o = \mathbf{R}^{-1}\mathbf{p}$.

The condition $E[e(k)\mathbf{x}^*(k)] = \mathbf{0}$ is called the "orthogonality condition" and states that the output error of the filter and the tap input vector are orthogonal ($e \perp \mathbf{x}$) when the filter has converged to the optimal solution.

The following formulations for the CLMS produce identical results:

$$y(k) = \mathbf{x}^T(k)\mathbf{w}(k) = \mathbf{w}^T(k)\mathbf{x}(k) \quad \rightarrow \quad \mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\mathbf{x}^*(k)$$

$$y(k) = \mathbf{w}^H(k)\mathbf{x}(k) = \mathbf{x}^T(k)\mathbf{w}^*(k) \quad \rightarrow \quad \mathbf{w}(k+1) = \mathbf{w}(k) + \mu e^*(k)\mathbf{x}(k)$$

$$y(k) = \mathbf{x}^H(k)\mathbf{w}(k) = \mathbf{w}^T(k)\mathbf{x}^*(k) \quad \rightarrow \quad \mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\mathbf{x}(k)$$

Types of convergence of CLMS

It is of greater interest, however, to analyse the evolution of the weights in time. As with any other estimation problem, we need to analyse the "bias" and "variance" of the estimator, that is:

- Convergence in the mean, to ascertain whether $\mathbf{w}(k) \to \mathbf{w}_o$ when $k \to \infty$;
- Convergence in the mean square, in order to establish whether the variance of the weight error vector $\mathbf{v}(k) = \mathbf{w}(k) - \mathbf{w}_o(k)$ approaches J_{min} as $k \to \infty$.

The analysis of convergence of linear adaptive filters is made mathematically tractable if we use so called **independence assumptions**, such that the filter coefficients are

- statistically independent of the data currently in filter memory, and
- $\{d(l), x(l)\}$ is independent of $\{d(k), x(k)\}$ for $k \neq l$.

Convergence of CLMS in the Mean

Assume $d(k) = \mathbf{x}^H(k)\mathbf{w}_o + q(k),\ q(k) \sim \mathcal{N}(0, \sigma_q^2)$. Then

$$\begin{aligned} e(k) &= \mathbf{x}^H(k)\mathbf{w}_o + q(k) - \mathbf{x}^H(k)\mathbf{w}(k) \\ \mathbf{w}(k+1) &= \mathbf{w}(k) + \mu\mathbf{x}(k)\mathbf{x}^H\mathbf{w}_o - \mu\mathbf{x}(k)\mathbf{x}^H(k)\mathbf{w}(k) + \mu q(k)\mathbf{x}(k) \end{aligned}$$

Subtract $\mathbf{w}_o$ from both sides, and define the weight error vector $\mathbf{v}(k) \stackrel{\text{def}}{=} \mathbf{w}(k) - \mathbf{w}_o$

$$\mathbf{v}(k+1) = \mathbf{v}(k) - \mu\mathbf{x}(k)\mathbf{x}^H(k)\mathbf{v}(k) + \mu q(k)\mathbf{x}(k)$$

Applying the statistical expectation operator $E\{\cdot\}$ gives the mean weight error recursion

$$E[\mathbf{v}(k+1)] = \big(\mathbf{I} - \mu E[\mathbf{x}(k)\mathbf{x}^H(k)]\big)E[\mathbf{v}(k)] + \mu \underbrace{E[q(k)\mathbf{x}(k)]}_{=\mathbf{0}}$$

$$\Longrightarrow\ E[\mathbf{v}(k+1)] = \big(\mathbf{I} - \mu\mathbf{R}\big)E[\mathbf{v}(k)]$$

Challenge: Unless the correlation matrix $\mathbf{R}$ is diagonal, there will be cross–coupling between the coefficients of the weight error vector.

Convergence of CLMS in the Mean (contd.)

Solution: Since the eigenvalue decomposition of $\mathbf{R} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^H$, rotating the weight error vector $\mathbf{v}(k)$ by the eigenvector matrix $\mathbf{Q}$, that is, $\mathbf{v}'(k) \stackrel{\text{def}}{=} \mathbf{Q}^H E\{\mathbf{v}(k)\}$, decouples the evolution of its coefficients.

Proof: Pre-multiply both sides of mean weight vector error recursion:

$$\mathbf{Q}^H E[\mathbf{v}(k+1)] = \mathbf{Q}^H \big(\mathbf{I} - \mu \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^H\big) E[\mathbf{v}(k)]$$

Since the eigenvector matrix is unitary, i.e. $\mathbf{Q}^H\mathbf{Q} = \mathbf{I}$, the **modes of convergence** are

$$\mathbf{v}'(k+1) = (\mathbf{I} - \mu\mathbf{\Lambda})\, \mathbf{v}'(k)$$

This recursion will be stable if the diagonal elements of $(\mathbf{I} - \mu\mathbf{\Lambda})$ satisfy

$$|1 - \mu\lambda_i| < 1, \quad \text{for } i = 1, \ldots, N \qquad \Longrightarrow\ 0 < \mu < \frac{2}{\lambda_{\max}} < \frac{2}{\text{tr}[\mathbf{R}]}$$

Since $\text{tr}[\mathbf{R}] = NE\{|x(k)|^2\}$, an easier to estimate bound $0 < \mu < \frac{2}{NE[|x(k)|^2|]}$.

Convergence of CLMS in the Mean Square

As the filter coefficients converge in the mean, they fluctuate around their optimum values $\mathbf{w}_o$. As a result, the mean square error $\xi(k) = E[|e(k)|^2]$ exceeds the minimum mean square error J_{min} by an amount referred to as the **excess mean square error**, denoted by $\xi_{EMSE}(k)$, that is

$$\xi(k) = J_{min} + \xi_{EMSE}(k) \quad \overset{J_{min}=\sigma_d^2}{\Longrightarrow} \quad \xi(k) = \sigma_q^2 + E\left[\mathbf{v}^H(k)\mathbf{R}\mathbf{v}(k)\right] = \sigma_q^2 + \text{tr}[\mathbf{R}\mathbf{K}(k)]$$

We have used the identity $E[\mathbf{v}^H(k)\mathbf{R}\mathbf{v}(k)] = \text{tr}[\mathbf{R}\mathbf{K}(k)] = \text{tr}[\mathbf{K}(k)\mathbf{R}]$, where $\mathbf{K}(k) = E[\mathbf{v}(k)\mathbf{v}^H(k)]$.

The excess mean square error depends on second order statistical properties of $d(k), \mathbf{x}(k), \mathbf{w}(k), e(k)$. **The plot showing time evolution of the mean square error is called the *learning curve*.**

For convergence in the **mean square** the misadjustment

$$\mathcal{M} = \frac{\xi_{EMSE}(\infty)}{\xi_{min}} = \frac{\xi_{EMSE}(\infty)}{\sigma_q^2} \approx \frac{1}{2}\mu\sigma_x^2 N$$

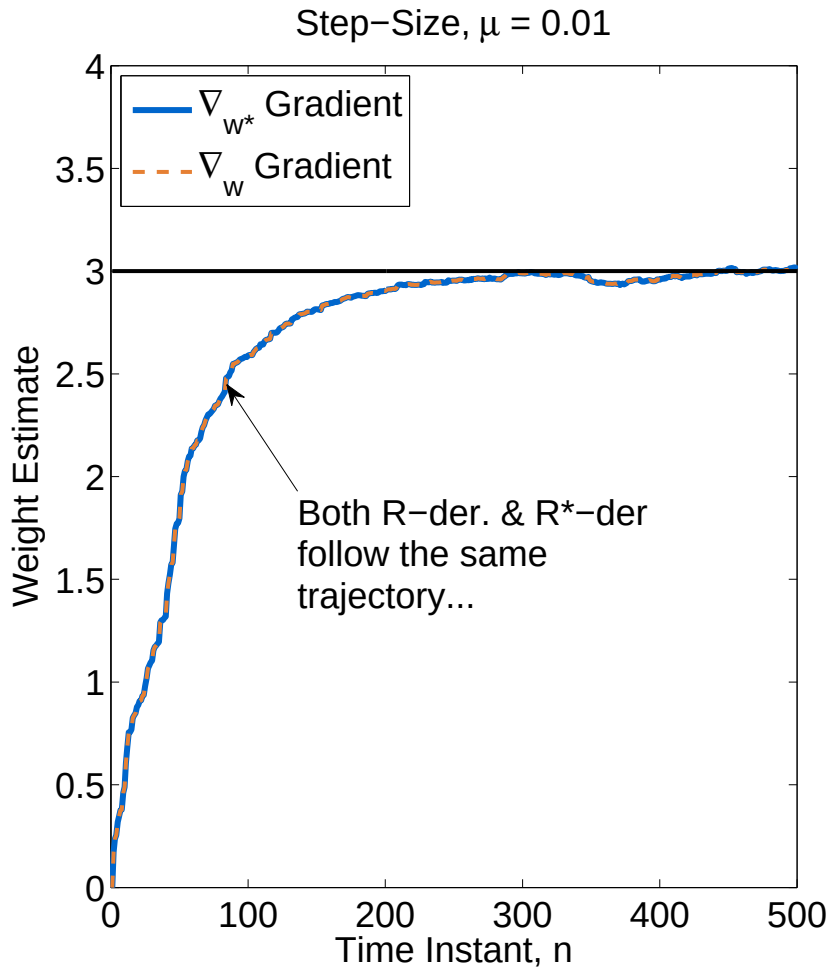
must be bounded and positive, that is, $(1 - \frac{1}{2}\mu\text{tr}[\mathbf{R}] > 0)$, and therefore

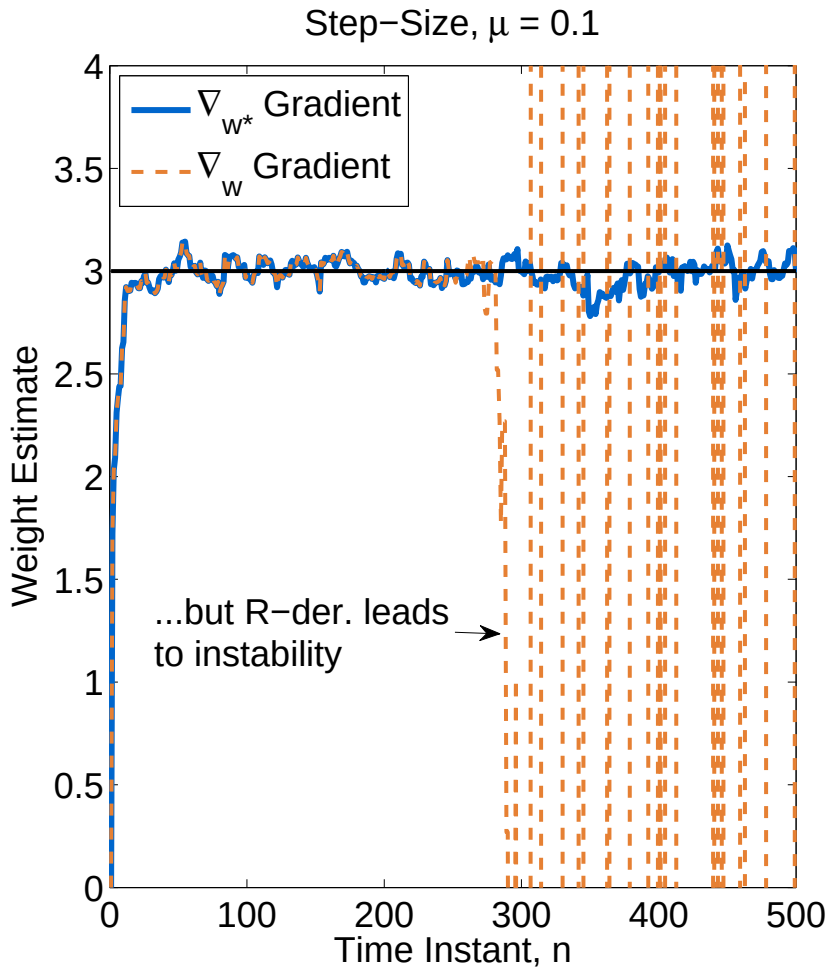
$$0 < \mu < 2/\text{tr}[\mathbf{R}] \quad \leftrightarrow \quad 0 < \mu < 2/(\sigma_{\text{x}}^2\text{N}).$$

Which derivative to we choose to compute the gradient?

$\mathbb{R}$-der vs. $\mathbb{R}^*$-der?

Simulation for the CLMS derived using $\mathbb{R}$-der. and $\mathbb{R}^*$-der. $(\mathbf{w}_o = 3)$





Complex Wiener filter

Notice that we have used: $(\mathbf{x}^H\mathbf{y})^* = \mathbf{y}^H\mathbf{x}$

The **Wiener solution** can be obtained by minimising

$$J(\mathbf{w}) = E\big[e(k)e^*(k)\big] = E\big[\big(d(k) - \mathbf{w}^H\mathbf{x}(k)\big)\big(d(k) - \mathbf{w}^H\mathbf{x}(k)\big)^*\big]$$

$$\begin{aligned} J(\mathbf{w}) &= E\big[|d(k)|^2\big] - \mathbf{w}^H E\big[\mathbf{x}(k)d^*(k)\big] - \mathbf{w}^T E\big[\mathbf{x}^*(k)d(k)\big] + \mathbf{w}^H E\big[\mathbf{x}(k)\mathbf{x}^H(k)\big]\mathbf{w} \\ &= \sigma_d^2 - \mathbf{w}^H\mathbf{p} - \mathbf{p}^H\mathbf{w} + \mathbf{w}^H\mathbf{R}\mathbf{w} \quad \rightarrow \quad \nabla_{\mathbf{w}^*}J(\mathbf{w}) = \mathbf{R}\mathbf{w} - \mathbf{p} = \mathbf{0} \end{aligned}$$

The optimum solution $\mathbf{w}_o$ and the minimum mean square error $J_{\min}$

$$\mathbf{w}_o = arg\min_{\mathbf{w}} J(\mathbf{w}) = \mathbf{R}^{-1}\mathbf{p} \qquad J_{min} = J(\mathbf{w}_o) = \sigma_d^2 - \mathbf{p}^H\mathbf{R}^{-1}\mathbf{p}$$

Also, $J(\mathbf{w})$ can be expressed as $J(\mathbf{w}) = J_{min} + (\mathbf{w} - \mathbf{w}_o)^H\,\mathbf{R}\,(\mathbf{w} - \mathbf{w}_o)$ by noticing that $d(k) = \mathbf{w}_o^H\mathbf{x}(k) + q(k)$ and $e(k) = (\mathbf{w}_o - \mathbf{w})^H\mathbf{x}_k + q(k)$, where $q(k) \sim \mathcal{N}(0, \sigma_q^2)$.

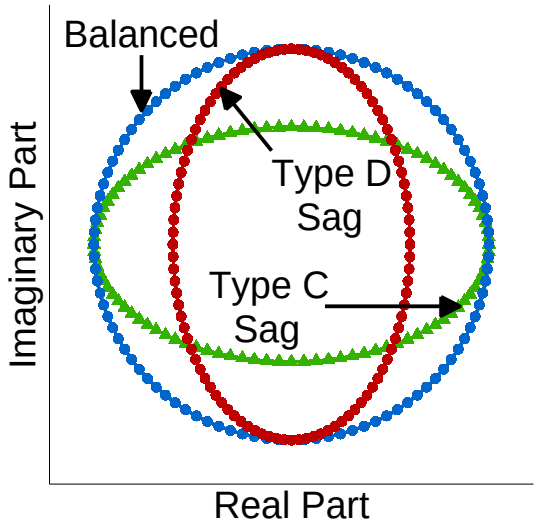
Notice that $J(\mathbf{w})$ is quadratic in $\mathbf{w}$ and has a global minimum for $\mathbf{w} = \mathbf{w}_o$, with $J_{min} = \sigma_q^2$.

Does Circularity Influence Estimation in $\mathbb{C}$?

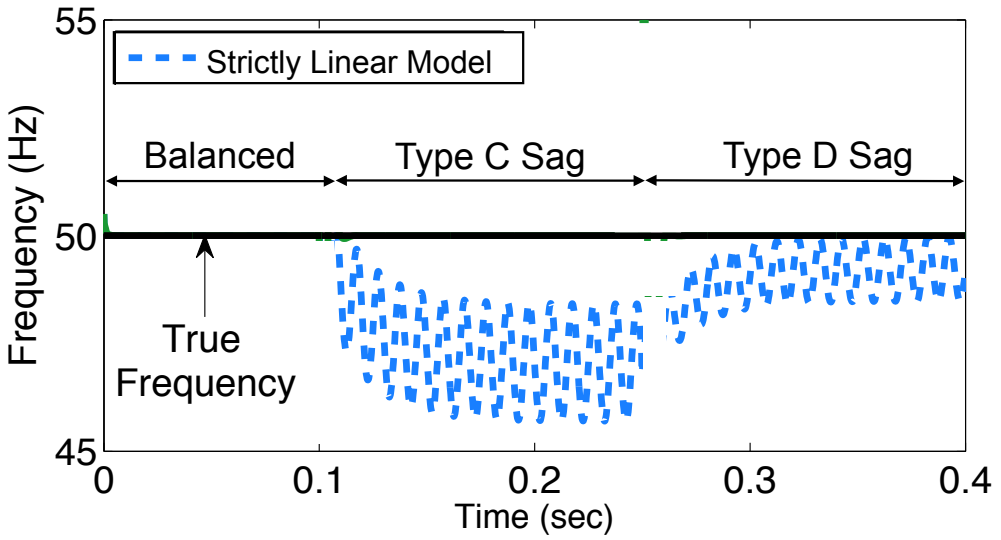
Voltage Sag: A magnitude and/or phase imbalance

- For balanced systems, $v(k) = A(k)e^{\jmath\omega k\Delta T} \rightarrow$ circular trajectory.
- Unbalanced systems, $v(k) = A(k)e^{\jmath\omega k\Delta T} + B(k)\mathbf{e}^{-\jmath\omega\mathbf{k}\boldsymbol{\Delta}\mathbf{T}}$ are influenced by the "conjugate" component.
- We need the complex conjugate when the modelling the signal.

Circularity Diagram



Strictly linear model yields biased estimates when system is unbalanced



What are we doing wrong $\hookrightarrow$ the Widely Linear Model

Consider the MSE estimator of a signal y in terms of another observation x

$$\hat{y} = E[y|x]$$

For zero mean, jointly normal y and x, the solution is

$$\hat{y} = \mathbf{h}^T \mathbf{x}$$

In standard MSE in the complex domain $\hat{y} = \mathbf{h}^H \mathbf{x}$, however

$$\hat{y}_r = E[y_r|x_r, x_i] \quad \& \quad \hat{y}_i = E[y_i|x_r, x_i]$$

$$thus \qquad \hat{y} = E[y_r|x_r, x_i] + \jmath E[y_i|x_r, x_i]$$

Upon employing the identities $x_r = (x + x^*)/2$ and $x_i = (x - x^*)/2\jmath$

$$\hat{y} = E[y_r|x, x^*] + \jmath E[y_i|x, x^*]$$

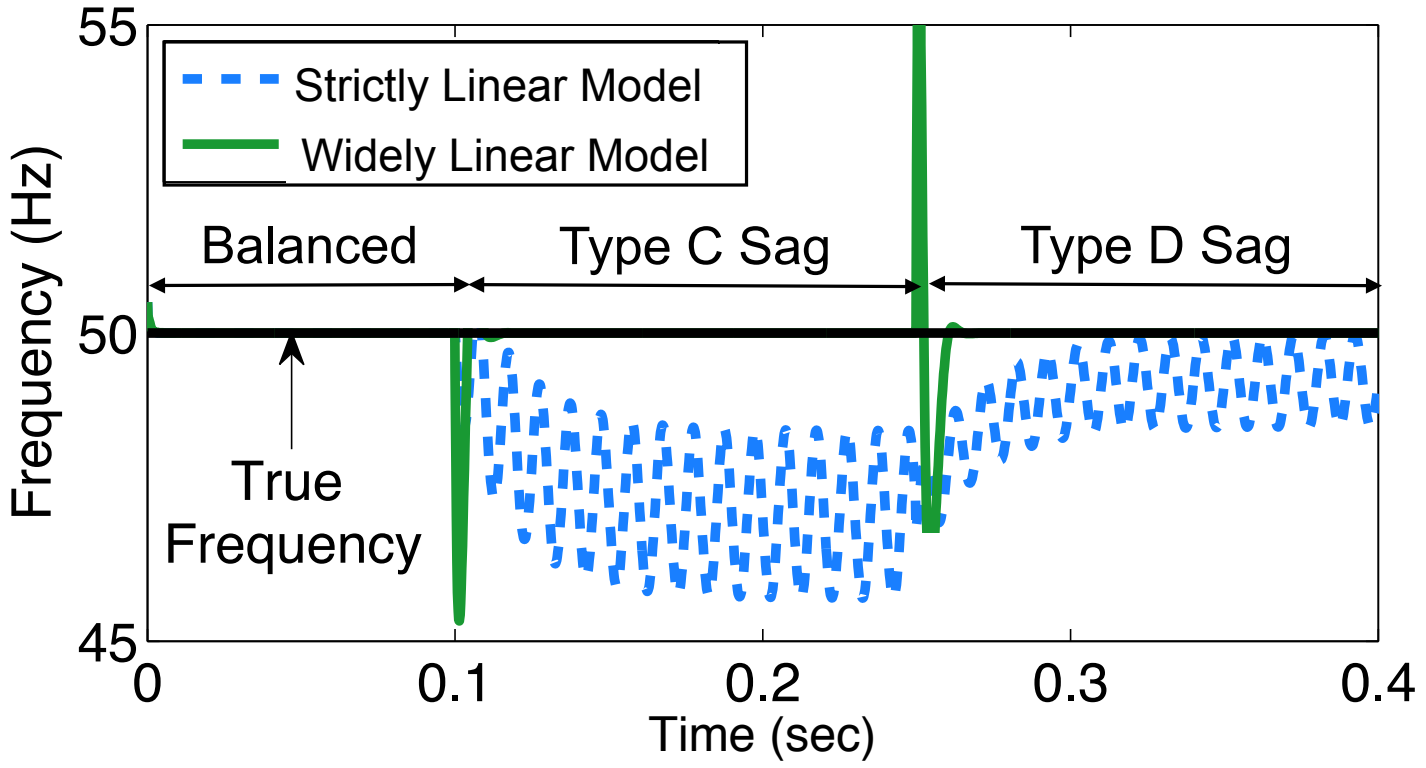
and thus arrive at the **widely linear** estimator for general complex signals

$$y = \mathbf{h}^T \mathbf{x} + \mathbf{g}^T \mathbf{x}^*$$

We can now process general (noncircular) complex signals!

Using the widely linear model for frequency estimation

The widely linear model is able to estimate the frequency for both **circular** (balanced) and **noncircular** (unbalanced) voltages.



Widely linear autoregressive modelling in $\mathbb{C}$

Standard AR model of order n is given by

$$z(k) = a_1 z(k-1) + \cdots + a_n z(k-n) + q(k) = \mathbf{a}^T \mathbf{z}(k) + q(k),$$

Using the Yule-Walker equations the AR coefficients are found from

$$\mathbf{a}^* = \mathcal{C}^{-1}\mathbf{c}$$

$$\begin{bmatrix} a_1^* \\ a_2^* \\ \vdots \\ a_n^* \end{bmatrix} = \begin{bmatrix} c(0) & c^*(1) & \dots & c^*(n-1) \\ c(1) & c(0) & \dots & c^*(n-2) \\ \vdots & \vdots & \ddots & \vdots \\ c(n-1) & c(n-2) & \dots & c(0) \end{bmatrix}^{-1} \begin{bmatrix} c(1) \\ c(2) \\ \vdots \\ c(n) \end{bmatrix}$$

where $\mathbf{c} = [c(1), c(2), \dots, c(n)]^T$ is the time shifted correlation vector.

Widely linear model

$$y(k) = \mathbf{h}^T(k)\mathbf{x}(k) + \mathbf{g}^T(k)\mathbf{x}^*(k) + q(k)$$

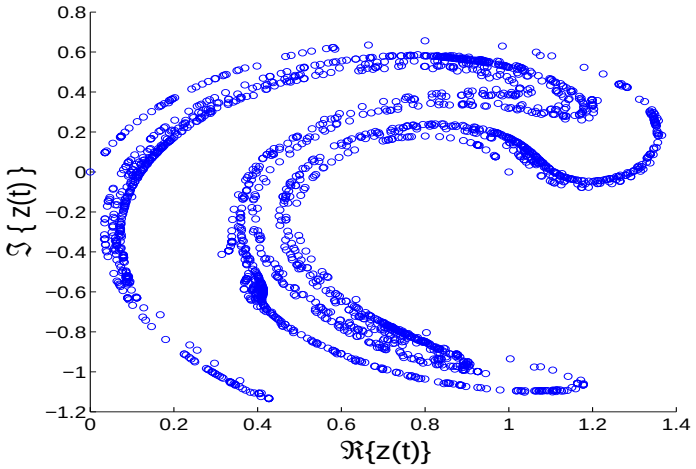
Widely linear normal equations

$$\begin{bmatrix} \mathbf{h}^* \\ \mathbf{g}^* \end{bmatrix} = \begin{bmatrix} \mathcal{C} & \mathcal{P} \\ \mathcal{P}^* & \mathcal{C}^* \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{c} \\ \mathbf{p}^* \end{bmatrix}$$

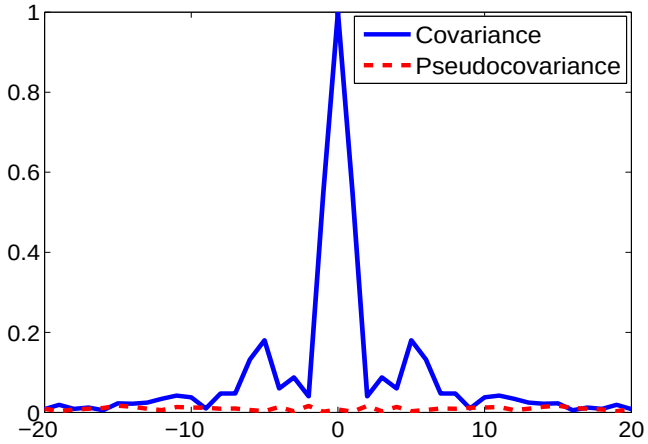
where $\mathbf{h}$ and $\mathbf{g}$ are coefficient vectors and $\mathbf{x}$ the regressor vector.

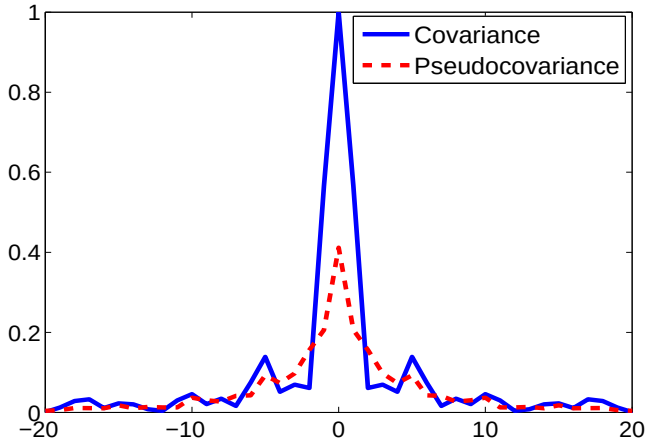
This is a rigorous way to model general complex signals!

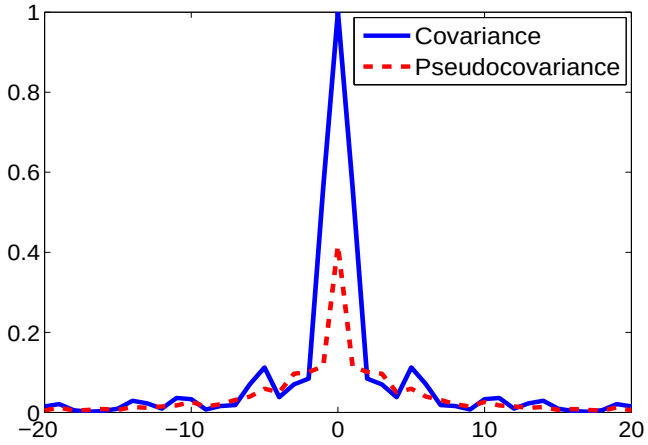
Circularity for Ikeda map



AR model of Ikeda signal







Covariances: Original Ikeda

Widely linear AR of Ikeda

The Augmented (widely linear) CLMS (ACLMS)

Widely linear model $y(k) = \mathbf{h}^{\mathsf{T}}(k)\mathbf{z}(k) + \mathbf{g}^{\mathsf{T}}(k)\mathbf{z}^*(k)$

$$\mathbf{h}(k+1) = \mathbf{h}(k) - \mu\nabla_{\mathbf{h}^*}J \quad \Rightarrow \quad \nabla_{\mathbf{h}^*}J = -e(k)\mathbf{x}^*(k)$$
$$\mathbf{g}(k+1) = \mathbf{g}(k) - \mu\nabla_{\mathbf{g}^*}J \quad \Rightarrow \quad \nabla_{\mathbf{g}^*}J = -e(k)\mathbf{x}(k)$$

Therefore, the ACLMS update

$$\begin{aligned}\mathbf{h}(k+1) &= \mathbf{h}(k) + \mu e(k)\mathbf{x}^*(k)\\ \mathbf{g}(k+1) &= \mathbf{g}(k) + \mu e(k)\mathbf{x}(k)\end{aligned}$$

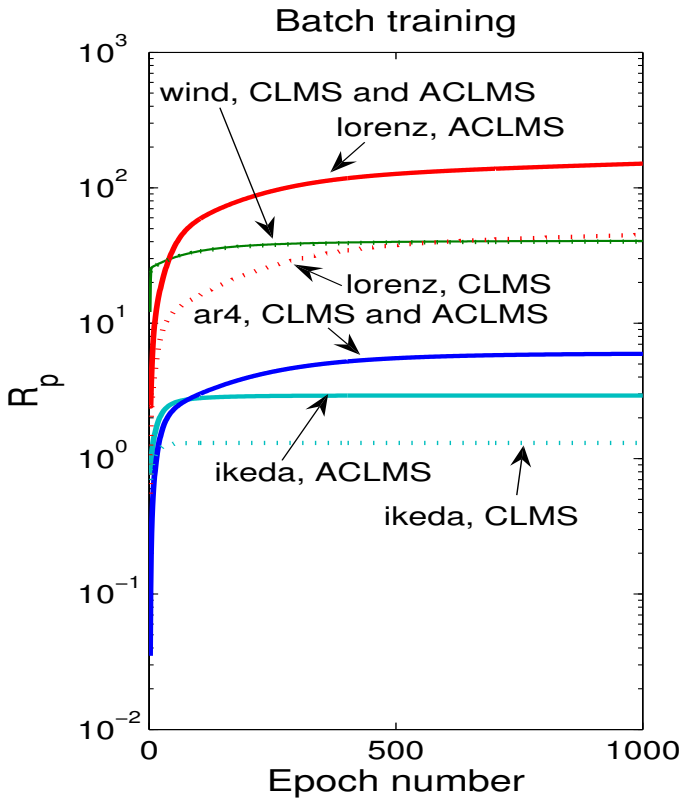
or in a more compact form (using augmented input and weight vectors)

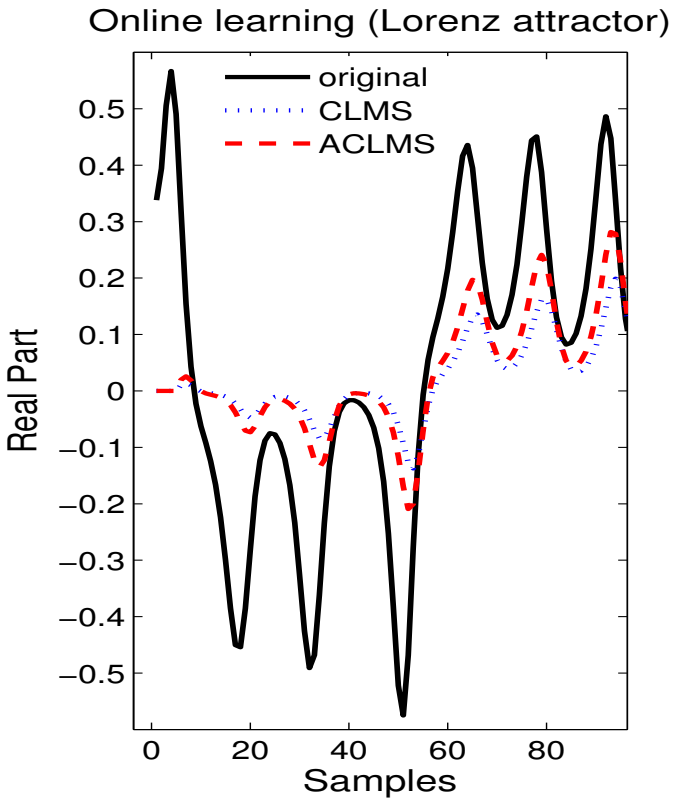
$$\mathbf{w}^a(k+1) = \mathbf{w}^a(k) + \eta e^a(k)\mathbf{x}^{a^*}(k)$$

where $\eta = \mu_h = \mu_g$, $\mathbf{w}^a(k) = [\mathbf{h}^T(k),\ \mathbf{g}^T(k)]^T$, $\mathbf{x}^a(k) = [\mathbf{x}^T(k),\ \mathbf{x}^H(k)]^T$, $e^a(k) = d(k) - \mathbf{x}^{a^T}(k)\mathbf{w}^a(k)$

Performance of the ACLMS

Evaluated for both second order circular (proper) and improper signals.

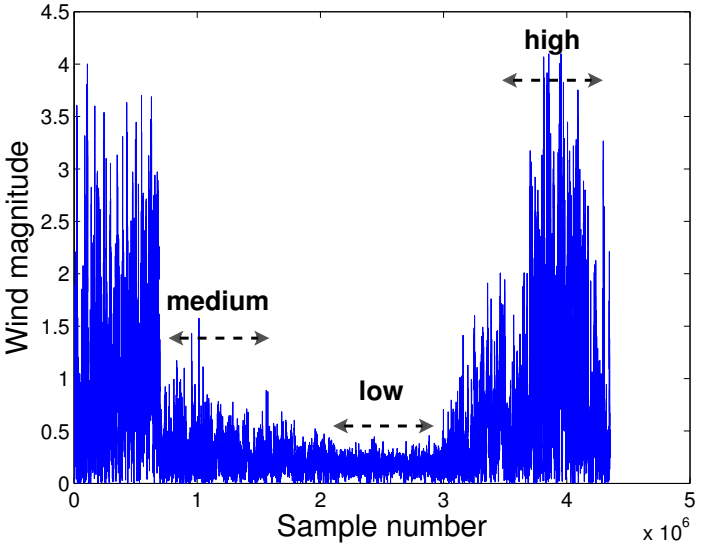




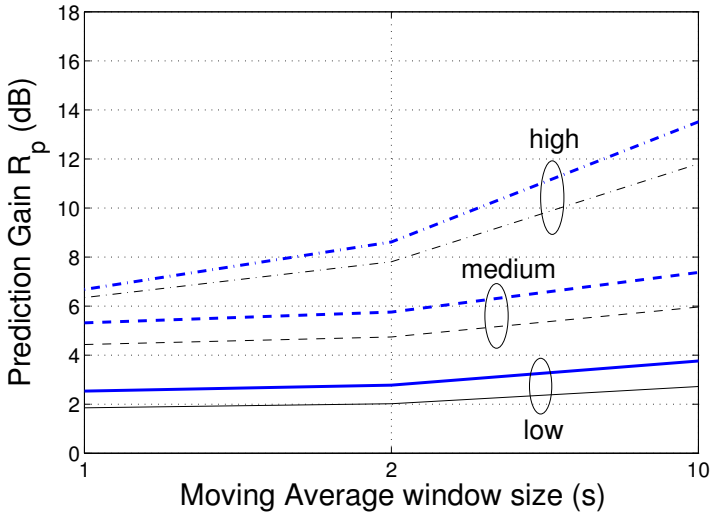
The ACLMS outperforms CLMS for second order noncircular signals.

Wind modelling: Performance vs dynamics vs circularity

Data recorded in an urban environment over one day



(a) Modulus of complex windover one day



(b) CLMS vs ACLMS for different wind regimes. CLMS - black, ACLMS - blue

Different wind regimes $\rightsquigarrow$ different dynamics,

$$v(k) = |v(k)|e^{\jmath\Phi(k)},\ |v| \text{ - speed, } \Phi \text{ - direction}$$

Different dynamics $\rightsquigarrow$ different circularity properties $\rightsquigarrow$ impact of ACLMS

Part 2: Applications of Complex-Valued Adaptive Filters

Application: Circularity Tracker

Relationship between a complex variable and its conjugate

Consider the problem of using a zero-mean r.v. $z \in \mathbb{C}$ to estimate its complex conjugate, that is

$$\hat{z}^* = w^* z$$

find an estimate of w that minimizes

$$J_{\text{MSE}} = E\{|e|^2\} = E\{|z^* - \hat{z}^*|^2\}$$

The Wiener solution

$$w_{\text{opt}} = c^{-1} r = \frac{E\{z^2\}}{E\{|z|^2\}} = \frac{p}{c}$$

$\rightarrow$ Circularity Quotient!!!

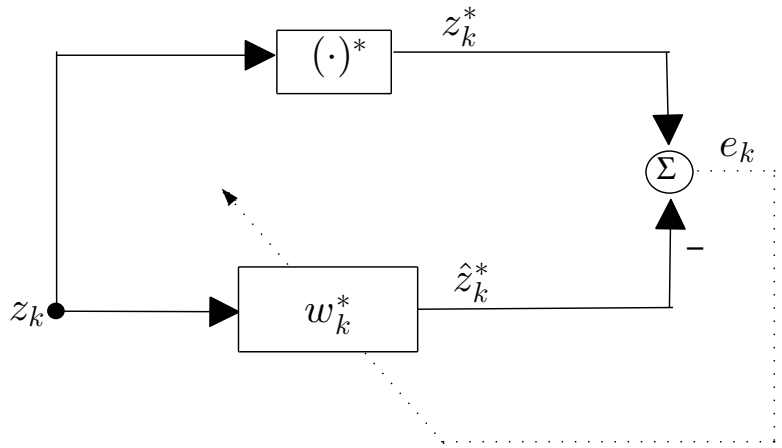
Real time tracker of complex circularity

Idea: Use an **adaptive filter** to estimate the complex conjugate of a signal using the original signal as the input.

Since the complex least mean square (CLMS) [Widrow 1975] estimates the Wiener solution, we can configure it with input signal z_k and the desired signal z_k^*.

The one-tap filter weight is the estimate of the circularity quotient ρ.

Adaptive filtering configuration



CLMS algorithm

$$\hat{z}_k^* = w_k^* z_k$$

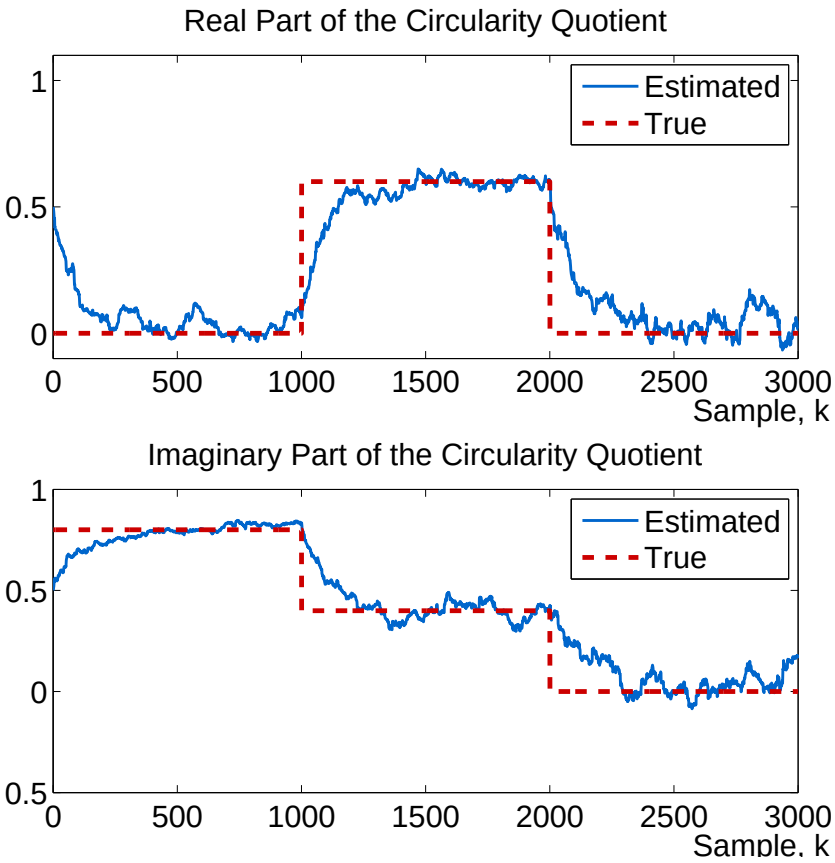
$$e_k = z_k^* - \hat{z}_k^*$$

$$w_{k+1} = w_k + \mu e_k^* z_k$$

Real time tracker of complex circularity

Simulations on synthetic white Gaussian data

The real and imaginary parts of the evolution of the CLMS weights when tracking circularity.



Synthetic signal was generated by concatenating three segments of zero-mean white Gaussian signals, $z_{i,k}$, with different properties, where

$$z_{i,k} = x_{i,k} + jy_{i,k} \quad z_i \sim \mathcal{N}(0, c, p_i)$$
$$i = \{1, 2, 3\}$$

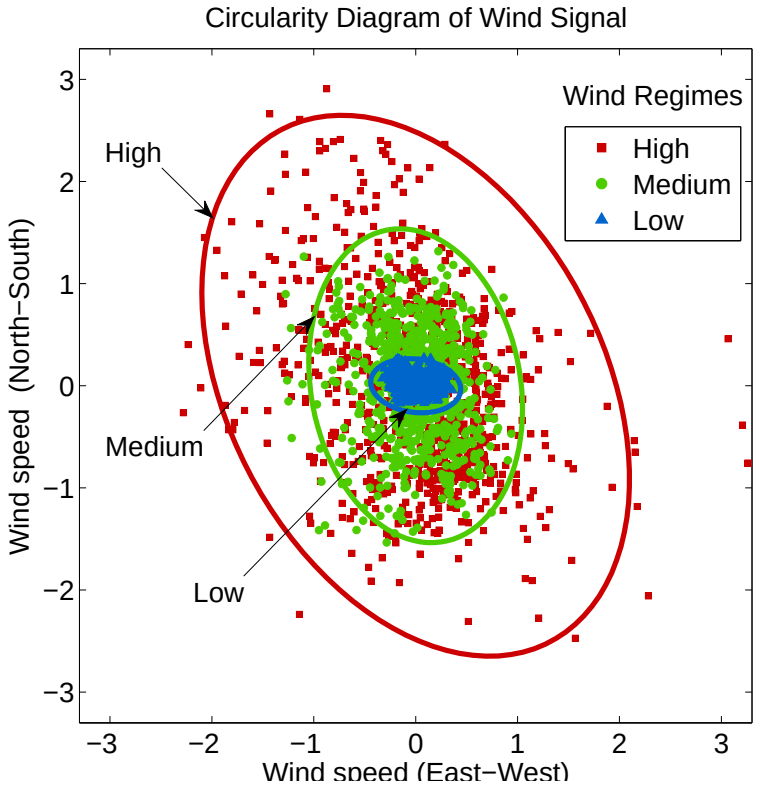
Each segment:

Sample, k	c	p_i	$\lvert\rho_i\rvert$
$1 - 1000$	1	$0.8j$	0.8
$1001 - 2000$	1	$0.6 + 0.4j$	0.72
$2001 - 3000$	1	0	0

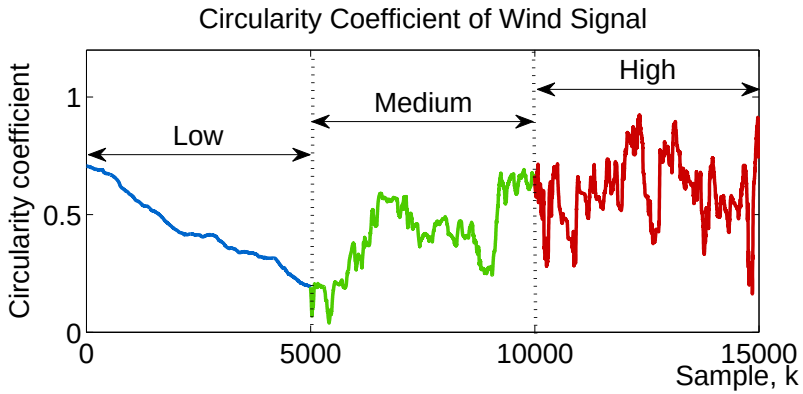
Real time tracker of complex circularity

Simulations on wind sensor data

The circularity diagram of wind speeds in the low, medium and high dynamic regimes.



The estimate of circularity coefficient for wind signals in the low, medium and high dynamic regimes using the proposed algorithm.

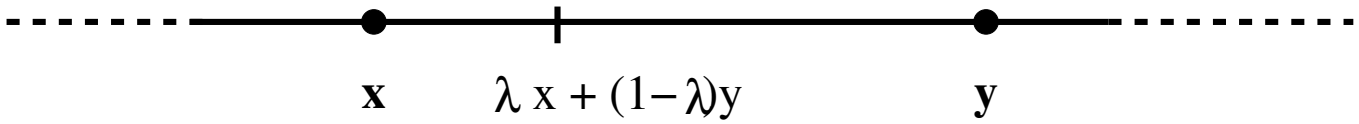


Wind signal modelled as wind a complex number

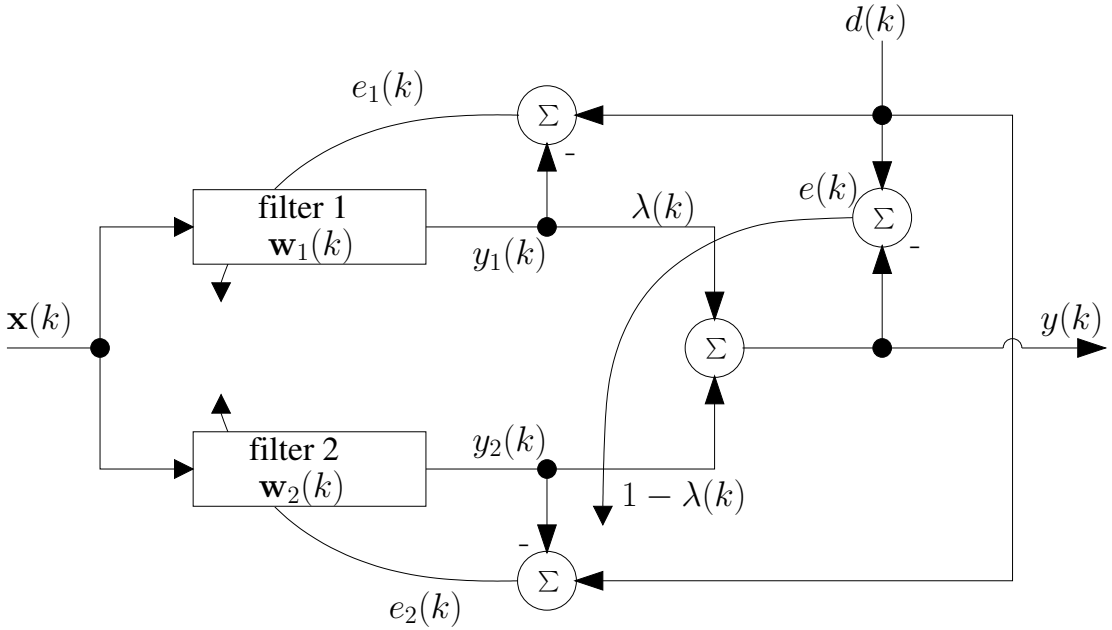
$$s = s_{\mathrm{E}} + j s_{\mathrm{N}}$$

Circular vs. Noncircular: That is the question
Best of both worlds $\hookrightarrow$ a Hybrid Filter

Virtues of Convex Combination ($\lambda \in [0, 1]$)



Convexity $\Rightarrow$ existence and uniqueness of solution ☺



Let Filter1 be trained by CLMS and Filter2 by ACLMS

Adaptation of the mixing parameter λ

To preserve their inherent characteristics, subfilters, $Filter1$ and $Filter2$ updated based on their own errors:

- Linear $e_{clms}(k)$
- Widely linear $e_{aclms}(k)$

The convex mixing parameter λ is updated based on based on

$$J(k) = \frac{1}{2}e(k)e^*(k) = \frac{1}{2}|e(k)|^2 \quad \rightsquigarrow \quad \nabla_\lambda J(k)_{\lambda=\lambda(k)} = e(k)\frac{\partial e^*(k)}{\partial \lambda(k)} + e^*(k)\frac{\partial e(k)}{\partial \lambda(k)}$$

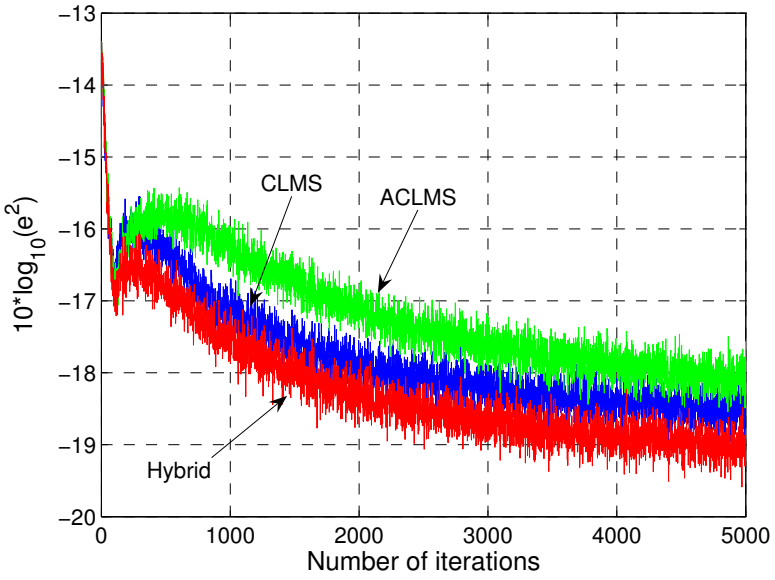
and the stochastic gradient based update of λ becomes

$$\begin{aligned} \lambda(k+1) &= \lambda(k) + \mu_\lambda \Big[e(k)\big(y_{aclms}(k) - y_{clms}(k)\big)^* \\ &+ e^*(k)\big(y_{clms}(k) - y_{aclms}(k)\big) \Big] \end{aligned}$$

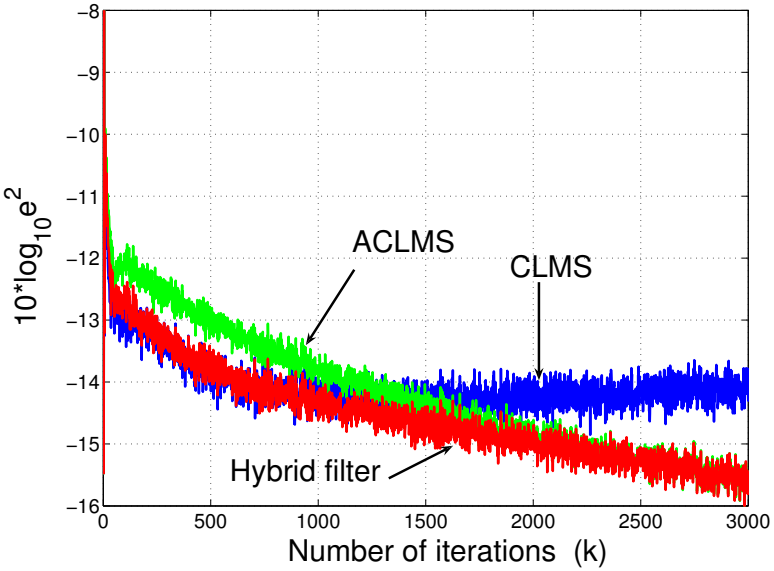
We must ensure that the value of $\lambda(k)$ belongs to $0 \leq \lambda(k) \leq 1$.

The hybrid CLMS $\leftrightarrow$ ACLMS filter (prediction setting)

Left: circular AR(4) process



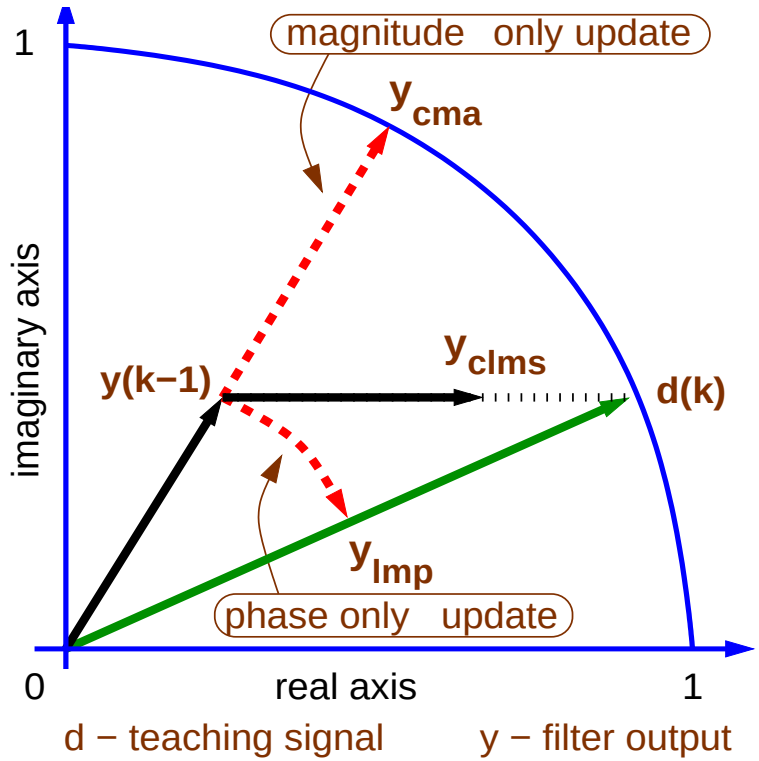
Right: Noncircular Ikeda map



- The CLMS has half the number of parameters of ALMS
- Hence, it initially converges faster for all test signals

- Both filters perform similarly for proper data in terms of the steady state
- ACLMS has superior steady state properties for the improper Ikeda map
- Hybrid filter: both fast covergence and excellent steady state properties!

A continuum between phase-only and magnitude-only cost functions (when does phase error matter?)



So, when does the (complex) phase error matter to the overall performance?

- **Answer #1:** Constant Modulus Channel Estimator (CMCE) [Rupp 1998] can estimate time-varying channels better *if phase error is ignored*.
- **Answer #2:** Least Mean Phase (LMP) [Tarighat/Sayed 2004] can estimate complex symbols better if *phase error term is added to update*.
- **Answer #3:** Least Mean Magnitude Phase (LMMP) [Douglas/Mandic ICASSP 2011] employs a combined magnitude- and phase-based criterion *best performance with stable behavior*.

Decomposing the squared error cost function

Decompose the well-known squared error cost function:

$$\begin{aligned} J_{\text{LMS}} &= |e(k)|^2 = |d(k) - y(k)|^2 \\ &= |d(k)|^2 + |y(k)|^2 - [d(k)y^*(k) + d(k)y^*(k)] \\ &= (|d(k)| - |y(k)|)^2 + 2|d(k)||y(k)| - [d(k)y^*(k) + d(k)y^*(k)] \\ &= \underbrace{(|d(k)| - |y(k)|)^2}_{\text{Magnitude Error}} + \underbrace{2|d(k)||y(k)|\,[1 - \cos(\theta_d - \theta_y)]}_{\text{Phase Error}} \end{aligned}$$

The Least Mean-Magnitude Phase (LMMP) algorithm can be derived as

$$\begin{aligned} \mathbf{w}(k+1) &= \mathbf{w}(k) - \mu_m \nabla_{\mathbf{w}^*} J_{\text{Mag}} - \mu_p \nabla_{\mathbf{w}^*} J_{\text{Phase}} \\ &= \mathbf{w}(k) + \mu_m \left(|d(k)| - |y(k)|\right) \frac{y_k}{|y_k|} \mathbf{x}_k^* + \mu_p \left(d(k) - |d(k)| \frac{y_k}{|y_k|} \right) \mathbf{x}_k^* \end{aligned}$$

If $\mu_p = \mu_m$, the LMMP simplifies into the CLMS!

Duality between bivariate real and complex filters

Desired signal: $\hat{d}(k) = \hat{d}_r(k) + \jmath\hat{d}_i(k)$, **Input:** $\mathbf{x}(k) = \mathbf{x}_r(k) + \jmath\mathbf{x}_i(k)$,

The bivariate (dual channel) real filter:

$$\begin{bmatrix}\hat{d}_r(k)\\ \hat{d}_i(k)\end{bmatrix} = \begin{bmatrix}\mathbf{a}^T(k) & \mathbf{b}^T(k)\\ \mathbf{c}^T(k) & \mathbf{d}^T(k)\end{bmatrix}\begin{bmatrix}\mathbf{x}_r(k)\\ \mathbf{x}_i(k)\end{bmatrix}$$

The strictly linear model $\hat{d}(k) = \mathbf{w}^H(k)\mathbf{x}(k)$ $(\mathbf{w}(k) = \mathbf{w}_r(k) + \jmath\mathbf{w}_i(k))$ in the CLMS is highly restrictive since it imposes the condition

$$\mathbf{a}(k) = \mathbf{d}(k) = \mathbf{w}_r(k) \quad \text{and} \quad \mathbf{b}(k) = -\mathbf{c}(k) = \mathbf{w}_i(k)$$

Augmented CLMS: $\hat{d}(k) = \mathbf{h}^H(k)\mathbf{x}(k) + \mathbf{g}^H(k)\mathbf{x}^*(k)$

$$\begin{bmatrix}\hat{d}_r(k)\\ \hat{d}_i(k)\end{bmatrix} = \begin{bmatrix}(\mathbf{h}_r(k) + \mathbf{g}_r(k))^T & (\mathbf{h}_i(k) - \mathbf{g}_i(k))^T\\ -(\mathbf{h}_i(k) + \mathbf{g}_i(k))^T & (\mathbf{h}_r(k) - \mathbf{g}_r(k))^T\end{bmatrix}\begin{bmatrix}\mathbf{x}_r(k)\\ \mathbf{x}_i(k)\end{bmatrix}$$

Sufficient degrees of freedom to model general complex signals!

The bound on the step-size which preserves convergence: $0 < \mu < \frac{2}{\lambda_{max}}$

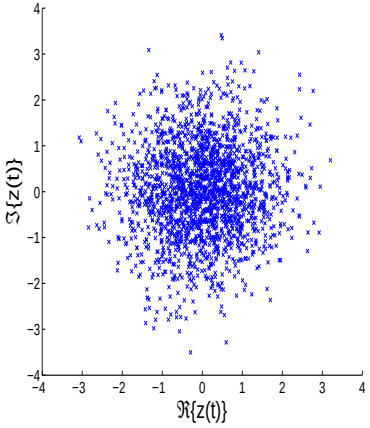
The bivariate and augmented complex correlation matrices related as

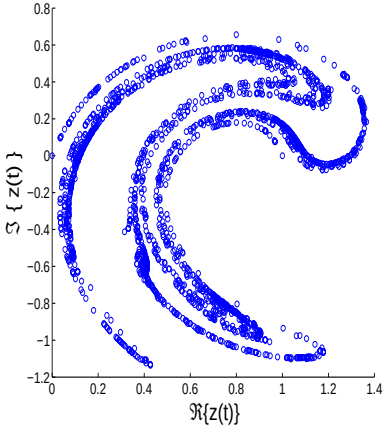
$$\mathcal{C}^a = E[\mathbf{z}^a\mathbf{z}^{aH}] = E\left[\mathbf{A}\boldsymbol{\omega}\boldsymbol{\omega}^T\mathbf{A}^H\right] = \mathbf{A}\mathbf{W}\mathbf{A}^H$$

It then follows that $\lambda^a = 2\lambda^\omega$.

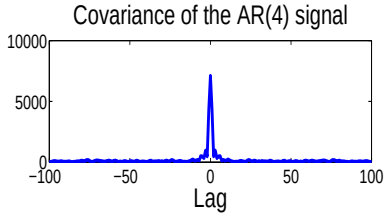
$\Rightarrow$ ACLMS and DCRLMS converge at the same speed when $\mu_r = 2\mu_{aclms}$.

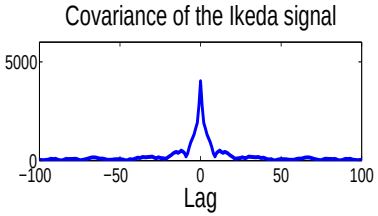
Duality: Simulations

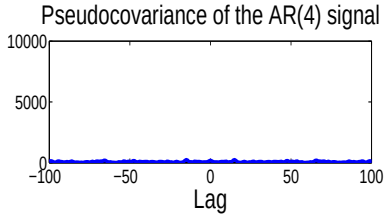


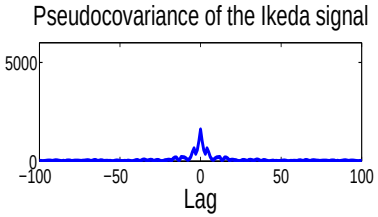


AR(4) and Ikeda signal









Covariances and pseudocovariances

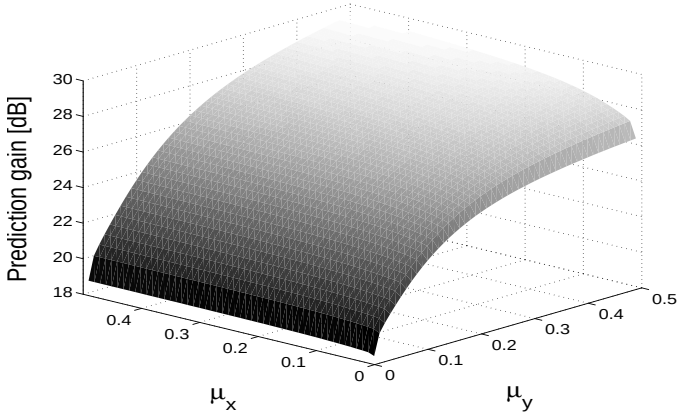
$$R_p = 10 \log \frac{\sigma_z^2}{\sigma_e^2}$$

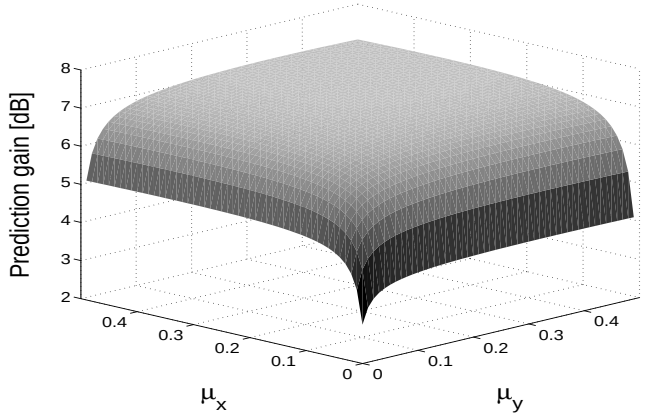
Algorithm	AR4	Ikeda	Wind
R_p for DCRLMS	5.8423	3.9733	13.2604
R_p for CLMS	6.6380	2.4278	14.2941
R_p for ACLMS	6.6096	4.0330	14.8926
R_p for DCRLMS (double μ)	6.6096	4.0330	14.8926

Duality simulations: Dependence on the parameters

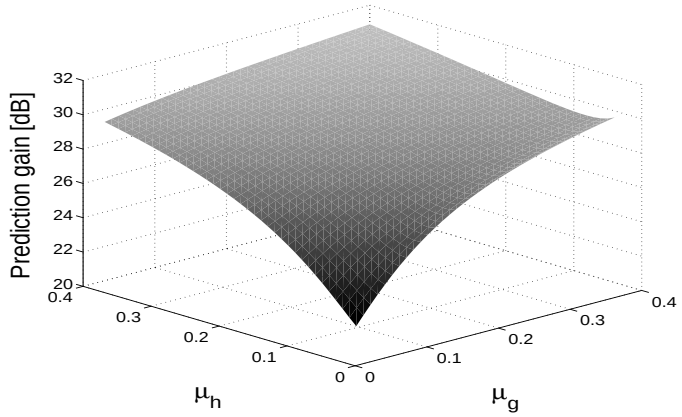
(observe the possibility for better tuning using the complex ACLMS)

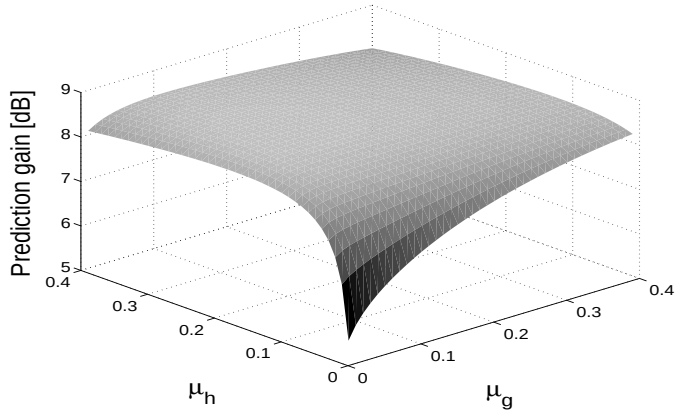
DCRLMS Lorenz (left) and wind signal (right)



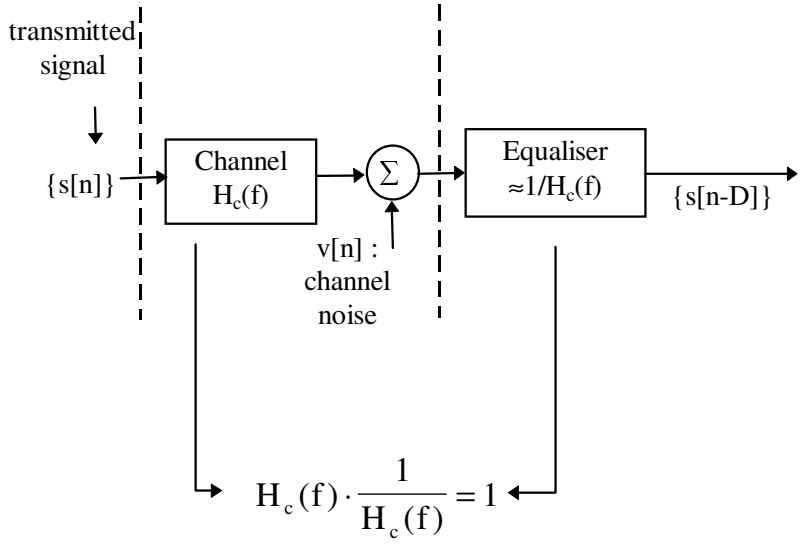


ACLMS Lorenz (left) and wind signal (right)





Example: Channel equalisation in digital communications



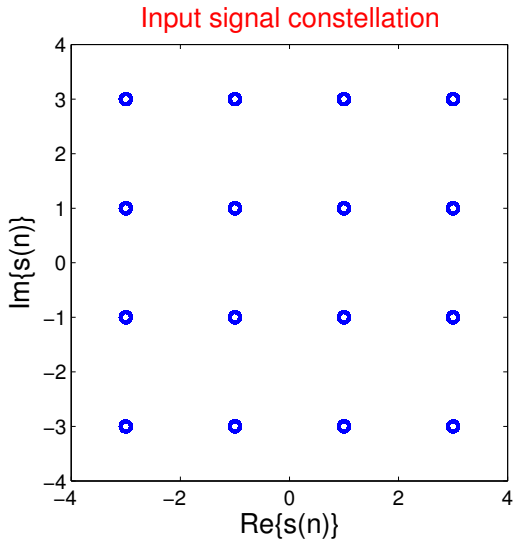
☐ Channel equalization is a simple way of mitigating the detrimental effects caused by a frequency-selective and/or dispersive communication link between sender and receiver.

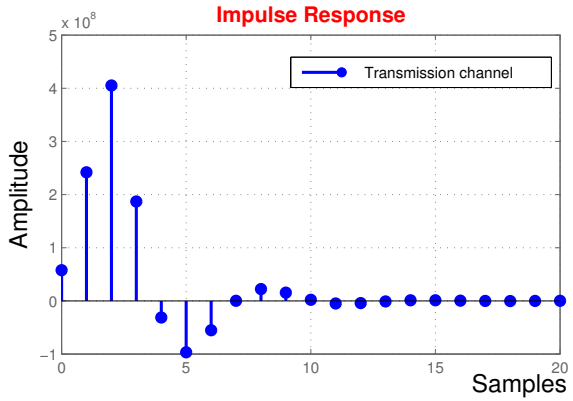
☐ During the training phase of channel equalization, a digital signal $s[n]$ that is known to both the transmitter and receiver is sent by the transmitter to the receiver

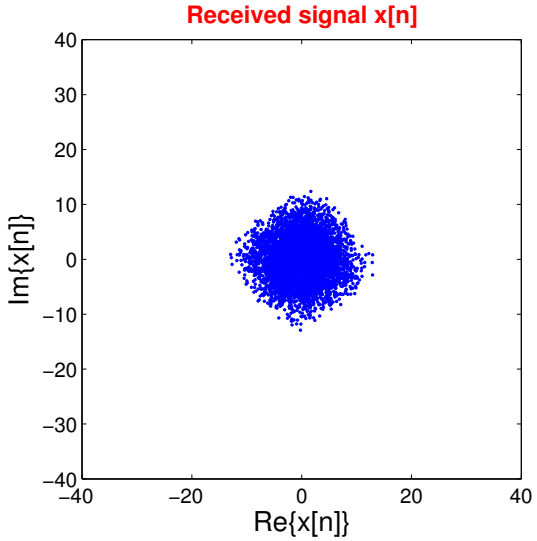
☐ The received signal $x[n]$ contains two signals: the signal $s[n]$ filtered by the channel impulse response, and an unknown broadband noise signal $v[n]$

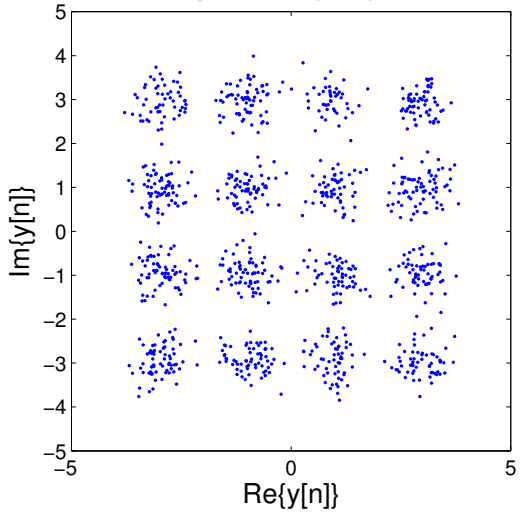
☐ The goal is to filter $x[n]$ to remove the inter-symbol interference (ISI) caused by the dispersive channel and to minimize the effect of the additive noise $v[n]$

Example: Digital communications ↬ continued









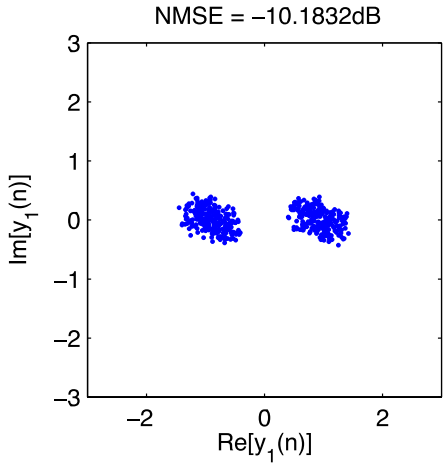
Beamforming example

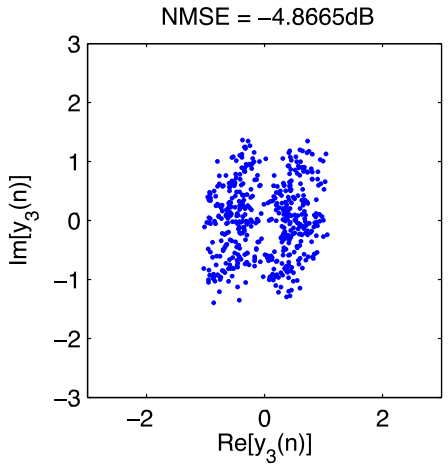
- Beamforming Model: ULA, $\lambda/2$ spacing
- Sources: BPSK, BPSK, QPSK, QPSK
- Angle of Arrival: $-45^\circ, 8^\circ, -13^\circ, 30^\circ$
- Number of Antenna Elements: 3
- Algorthims: ACLMS and CCLMS
- Desired Signal: $d(k) = s_p^*(k)$, $1 \leq p \leq 4$
- Step Size: $\mu = 0.0001$.

Compare: Convergence Rates, Steady State MSE

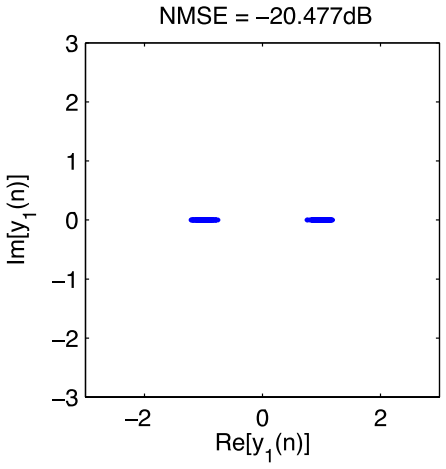
Output signal constellations

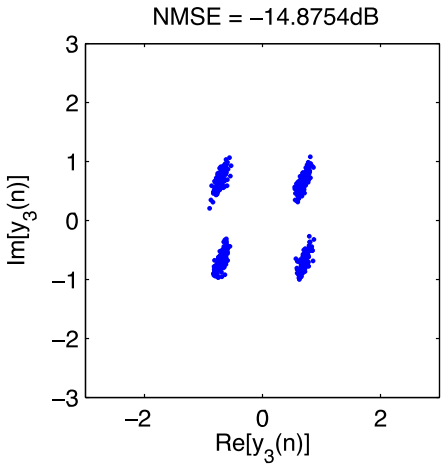
CCLMS





ACLMS



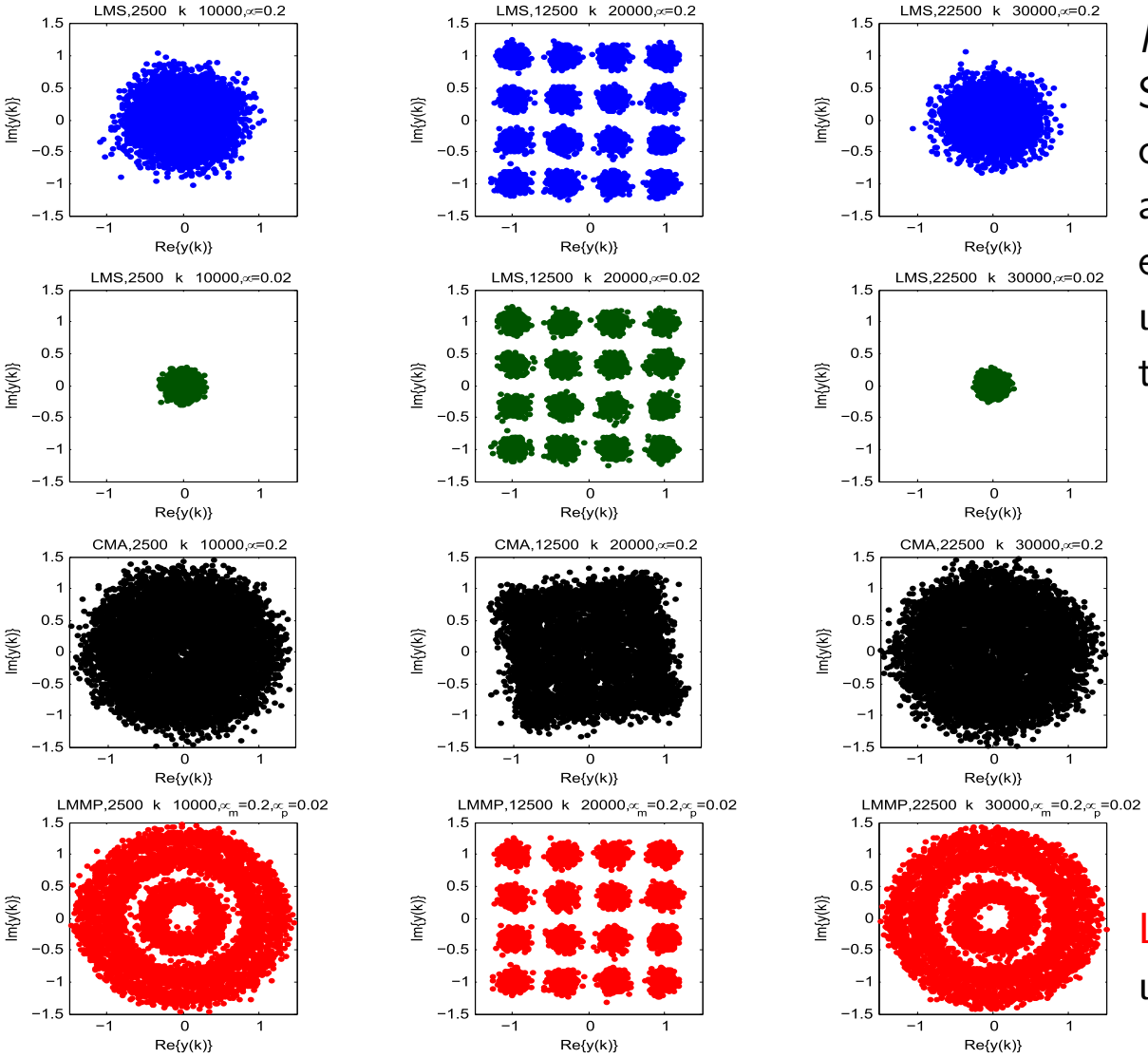


ACLMS has lower MSE because of non-circular binary sources

Signal constellations: channel equalization

Channel: 3-tap with frequency offset Source phases: time-varying

Performance meas.: average inter-symbol interfer., ISI (not dependent on phase)



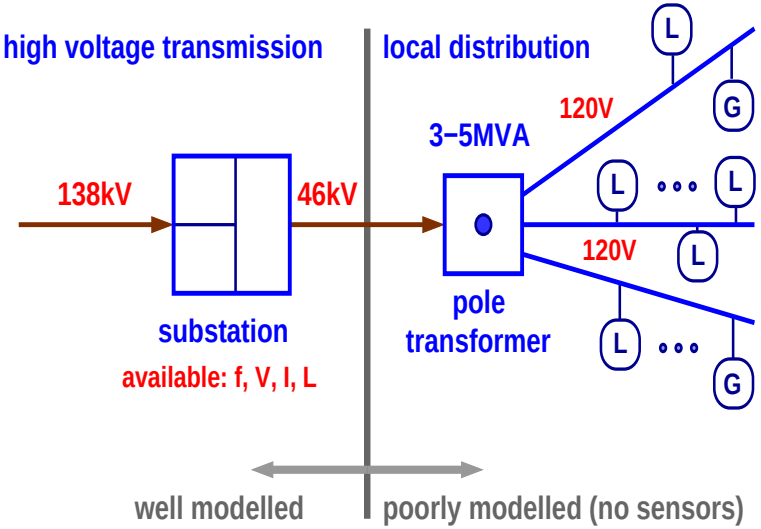
Moral:
Signal phase uncertainty can harm channel amplitude and phase estimation performance, unless it is mitigated within the algorithm itself.

LMMP in red addresses this uncertainty explicitly.

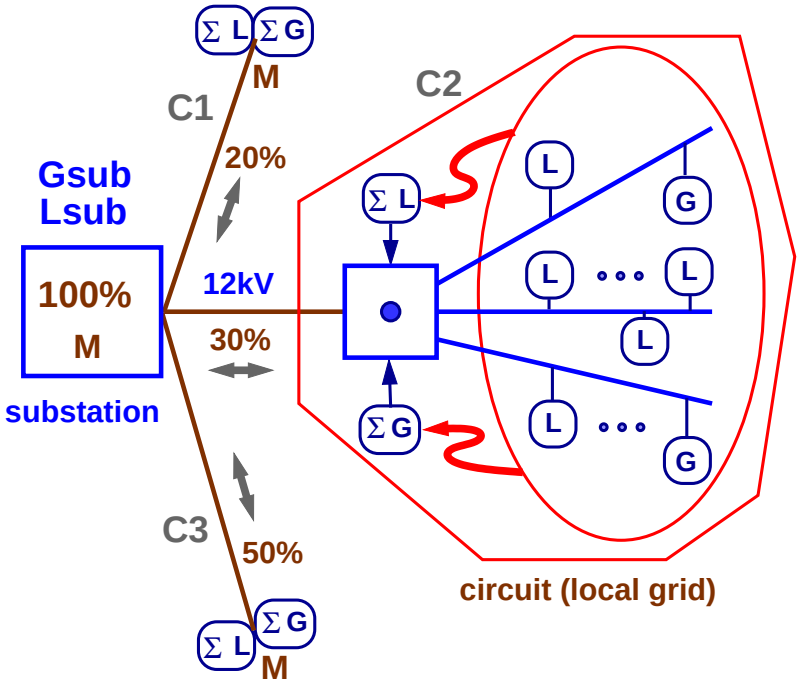
Case Study: Frequency estimation in smart grid

Sources of frequency deviation

Transmission: well-modelled distribution side is not



Block diagram of power grid



Nodal estimation

- Dual character of load/supply $\rightsquigarrow$ $f+$ for $G > L$ and $f-$ for $G < L$
- Harmonics and freq. drifts from loads with nonlinear $V - I$ properties
- Transient stability issues cause inaccurate frequency estimates, also switching on/off the shunt capacitors in reactive power compensation

The three-phase power system and $\alpha\beta$ transformation

Three-phase system where $V_a(k), V_b(k), V_c(k)$ are the peak values.

$$v_{\mathrm{a}}(k) = V_a(k)\cos(\omega k \triangle \mathrm{T} + \phi)$$

$$v_{\mathrm{b}}(k) = V_b(k)\cos(\omega k \triangle \mathrm{T} + \phi - \frac{2\pi}{3})$$

$$v_{\mathrm{c}}(k) = V_c(k)\cos(\omega k \triangle \mathrm{T} + \phi + \frac{2\pi}{3})$$

The $\alpha\beta$ transform - a complex signal which carries the same information

$$\begin{bmatrix} v_0 \\ v_\alpha(k) \\ v_\beta(k) \end{bmatrix} = \sqrt{\frac{2}{3}} \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} v_{\mathrm{a}}(k) \\ v_{\mathrm{b}}(k) \\ v_{\mathrm{c}}(k) \end{bmatrix}$$

For **balanced systems** $v_0 = 0$, and thus the complex Clarke's voltage

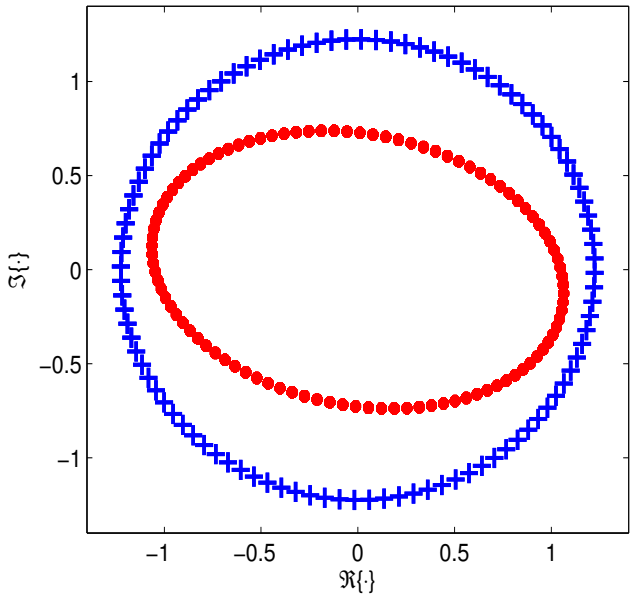
$$v(k) = v_\alpha(k) + \jmath v_\beta(k) = A(k)e^{\jmath(\omega k \Delta T + \phi)} + B(k)e^{-\jmath(\omega k \Delta T + \phi)}$$

$$A(k) = \frac{\sqrt{6}(V_a(k)+V_b(k)+V_c(k))}{6} \quad \text{and} \quad B(k) = \frac{\sqrt{6}(2V_a(k)-V_b(k)-V_c(k))}{12} - \frac{\sqrt{2}(V_b(k)-V_c(k))}{4}\jmath$$

Clearly, unbalanced systems are noncircular!

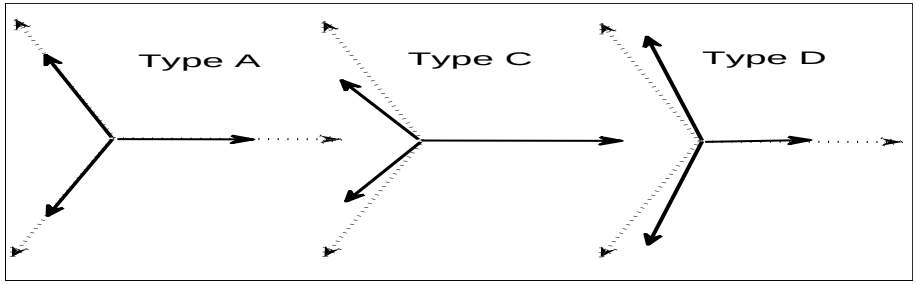
Noncircularity in unbalanced voltage conditions

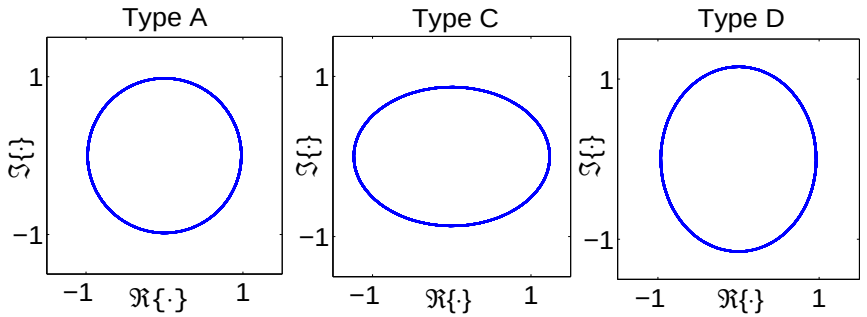
- **Balanced system:** $V_a(k) = V_b(k) = V_c(k)$, $A(k) = \text{const}$, $B(k) = 0$, and $v(k)$ is on a circle
- **Unbalanced system:** $V_a(k)$, $V_b(k)$, $V_c(k)$ are not identical
  - $A(k)$ is no longer constant, $B(k) \neq 0$
  - $v(k)$ is not on a circle $\rightarrow$ **a degree of noncircularity**



balanced and unbalanced system

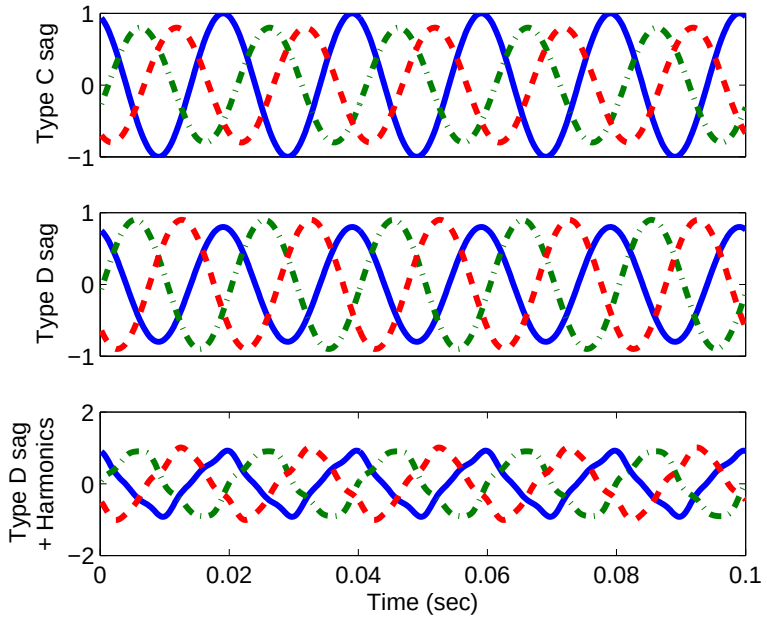
Some types of voltage sags

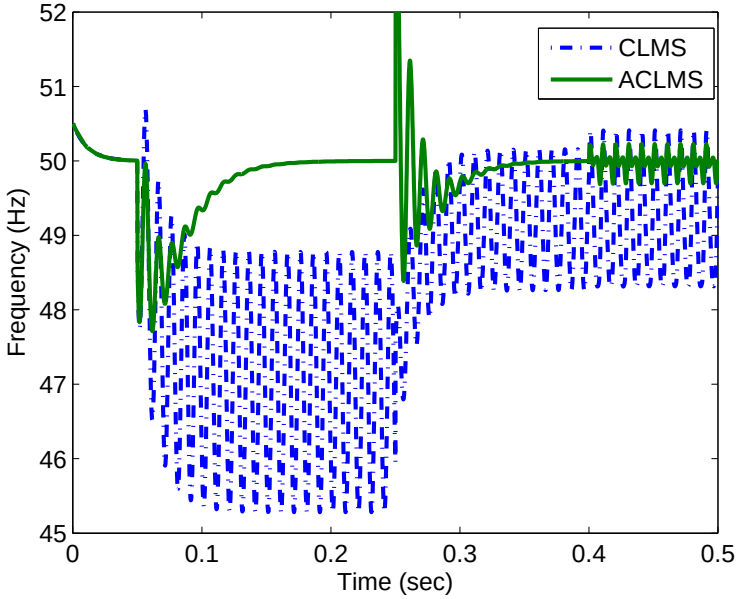




Their circularity diagrams

Simulations: Several successive sags



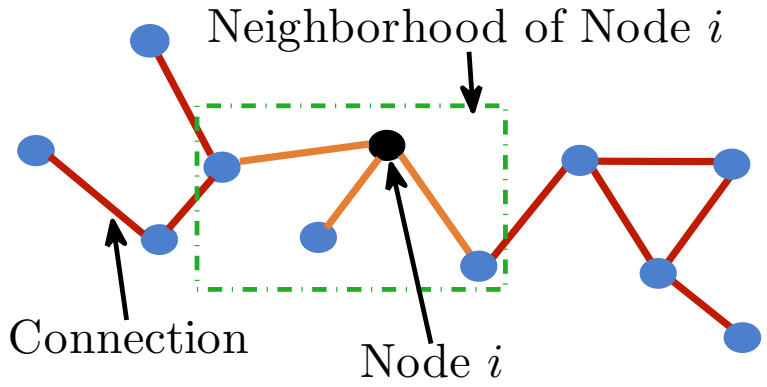


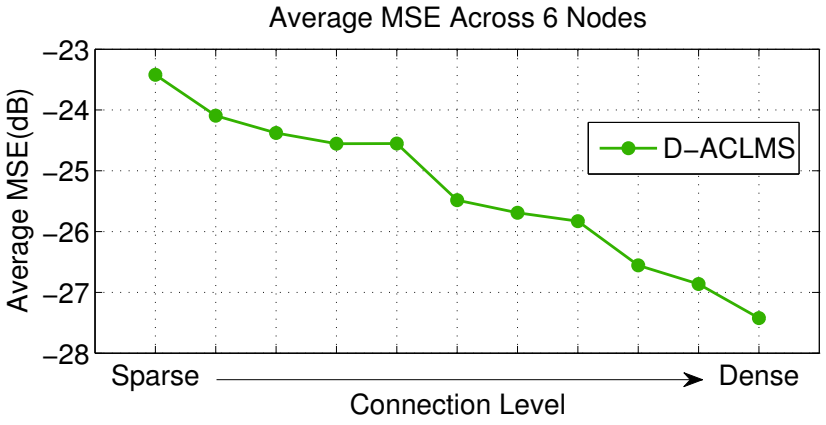
Phase voltages for different sags Linear and widely linear frequency estimation

- Initially, the power system (50Hz) was operating normally and both CLMS and ACLMS converged to 50Hz
- The widely linear ACLMS had advantage in the subsequent type C and D sags and under harmonics

Introduction to Distributed Adaptive Filters

Multiple adaptive filters collaborating in a network





The *Diffusion* ACLMS at node i

Input: $\mathbf{z}_i(k) = [\mathbf{x}_i^T(k), \mathbf{x}_i^H(k)]^T$,

Weights: $\mathbf{w}_i(k) = [\mathbf{h}_i^T(k), \mathbf{g}_i^T(k)]^T$

$$y_i(k) = \mathbf{w}_i^H(k)\mathbf{z}_i(k)$$

$$e_i(k) = d_i(k) - y_i(k)$$

$$\boldsymbol{\psi}_i(k+1) = \mathbf{w}_i(k) + \mu_i e_i^*(k)\mathbf{z}_i(k)$$

$$\mathbf{w}_i(k+1) = \sum_{\ell=1}^{N} a_{\ell i}\boldsymbol{\psi}_\ell(k+1)$$

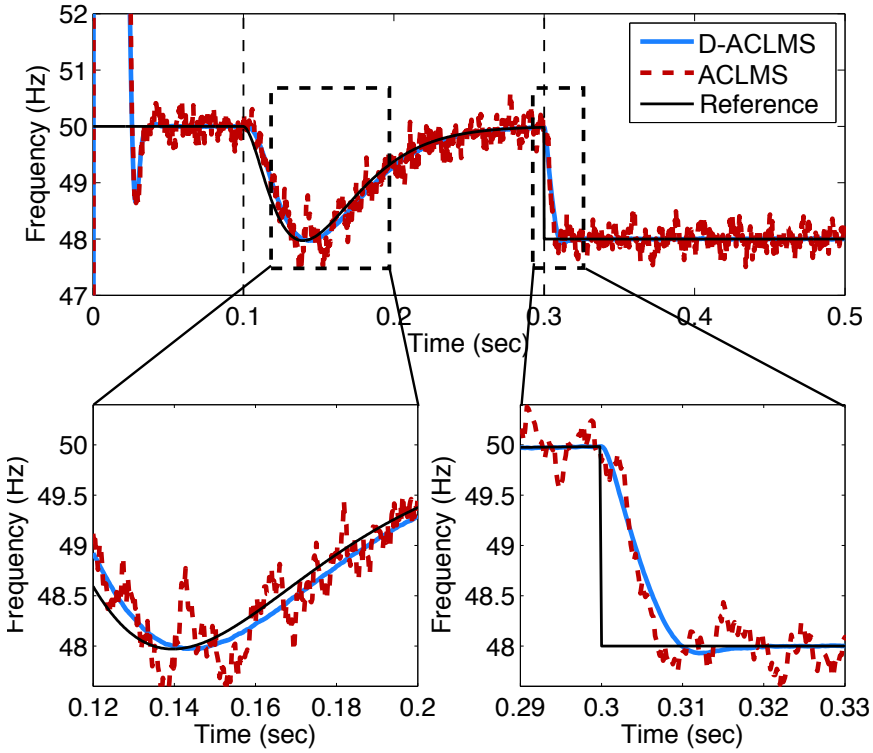
The weighting coefficients satisfy

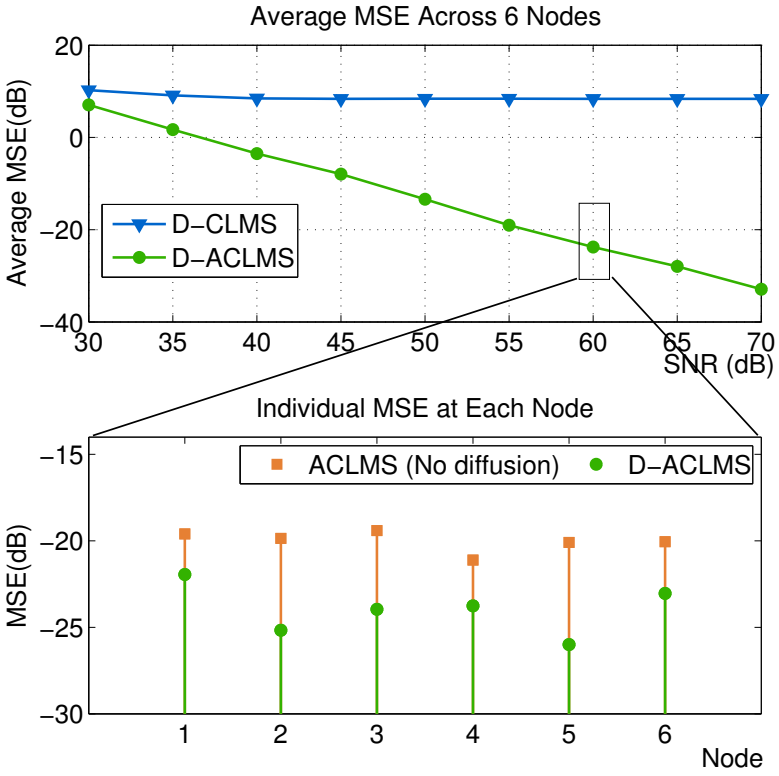
$$\sum_{\ell=1}^{N} a_{\ell i} = 1 \implies a_{ii} = 1 - \sum_{\ell \neq i} a_{\ell i}$$

Performance of the Diffusion-CLMS and Diffusion-ACLMS in the smart grid

Frequency estimate at a randomly selected node with multiple frequency events between 0.1 s and 0.5 s

The average MSE of the frequency estimate of D-ACLMS and D-CLMS under a Type D sag.



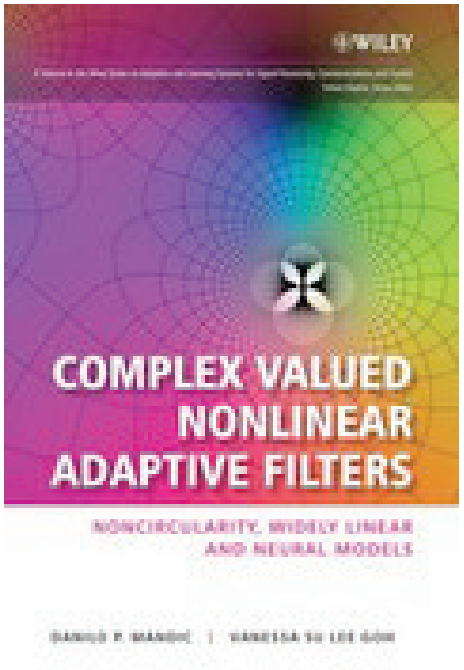


Conclusions

- Adaptive processing of noncircular complex signals
- These arise e.g. due to the nonlinearity of transievers (I/Q imbalance mitigation), multipath, in some modulation schemes (BPSK, GSMK, PAM, offset-QPSK) widely used in practical communications systems
- Standard solutions assume second order circularity of signal distributions and are inadequate when the signals are observed through nonlinear sensors, mixtures of sources, or noise model which is not doubly white
- This is achieved based on *augmented complex statistics* and *widely linear modelling*
- The complex LMS (CLMS) and augmented CLMS (ACLMS) introduced
- Convergence of CLMS and ACLMS – from the booklet provided (Chapter 6)
- This promises enhanced practical solutions in a variety of applications (interference supression, DoA estimation, blind estimation)

A comprehensive account of widely linear modelling

D.Mandic and V. Goh, "Complex Valued Nonlinear Adaptive FIlters: Noncircularity, Widely Linear and Neural Models", Wiley 2009.



- Unified approach to the design of complex valued adaptive filters and neural networks
- Augmented learning algorithms based on widely linear models
- Suitable for processing both second order circular (proper) and noncircular (improper) complex signals
- ACLMS, augmented Kalman filters, augmented CRTRL, linear and nonlinar IIR filters
- Adaptive stepsizes, dynamical range reduction, collaborative adaptive filters, statitical tests for the validity of complex representations

Some back–up material

Appendix: The RLS algorithms and its variants

From $\mathbf{w}(n+1) = \mathbf{R}^{-1}(n+1)\mathbf{p}(n+1)$, we have

$$\mathbf{w}(n+1) = \left[\mathbf{R}^{-1}(k) - \frac{\mathbf{R}^{-1}(k)\mathbf{x}(k+1)\mathbf{x}^T(k+1)\mathbf{R}^{-1}(k)}{\mathbf{x}^T(k+1)\mathbf{R}^{-1}(k)\mathbf{x}(k+1)+1}\right]\left[\mathbf{p}(n) + d(n)\mathbf{x}(n)\right]$$

After some grouping of the terms above, we arrive at

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \frac{\mathbf{R}^{-1}(n)}{1+\mathbf{x}^T(n)\mathbf{R}^{-1}(n)\mathbf{x}(n)}\, e(n)\mathbf{x}(n)$$

- the term $\mathbf{R}^{-1}$ "filters" the direction and length of the data vector $\mathbf{x}$
- The denominator $1+\mathbf{x}^T(n)\mathbf{R}^{-1}(n)$ is a measure of input signal power, normalised by $\mathbf{R}^{-1}$.
- This normalisation makes the power proportional to the data length N and not to the actual signal level $\leadsto$ it also decorrelates the data.

Related algorithms:

On the next slide.

Appendix: The RLS algorithms and its variants

Accelerated LMS:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu e(n)\mathbf{C}(n)\mathbf{x}(n), \quad \mathbf{C} \text{ a chosen matrix}$$

Normalised LMS:

$$\mu(n) = \frac{\mu}{\varepsilon + \mathbf{x}^T(n)\mathbf{x}(n)}, \quad 0 < \mu < 2$$

Exponentially weighted RLS (RLS with the forgetting factor)

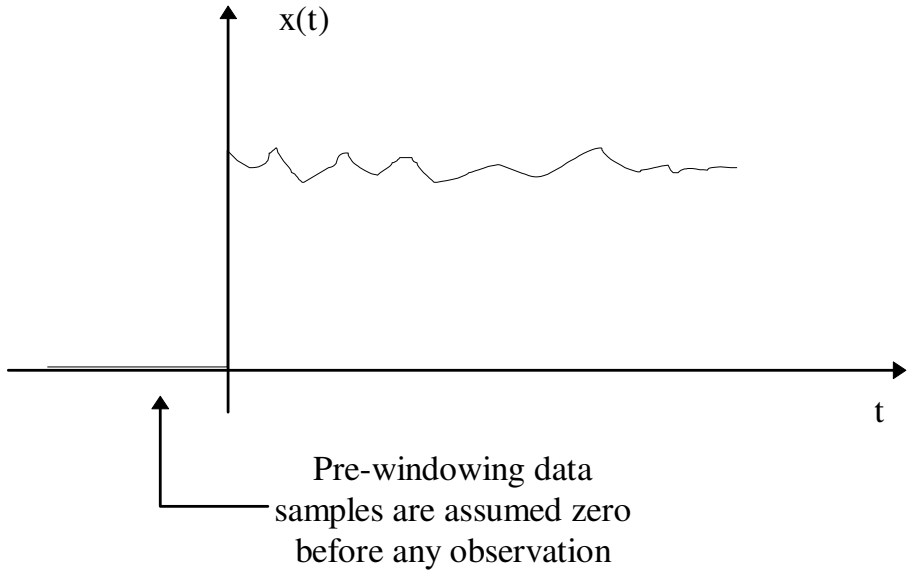
$$J(n) = \sum_{i=0}^{n} \lambda^{n-i} \left| d(i) - y(i) \right|^2, \quad \lambda \text{ is a forgetting factor}$$

- Forgetting factor $\varepsilon \in (0, 1]$, but typically > 0.95
- The forgetting factor introduces an effective window length of $\frac{1}{1-\varepsilon}$

Appendix: Pre–windowing

- To initialise RLS we need to make some assumptions about the input data

- The most common one is that of prewindowing, that is $\forall k < 0, x(k) = 0$



Appendix: Complete RLS algorithm

- Initialisation $\mathbf{w}(0) = \mathbf{0}$, $\mathbf{P}(0) = \delta\mathbf{I}$, $\mathbf{P}(k) = \mathbf{R}^{-1}$

- For each k

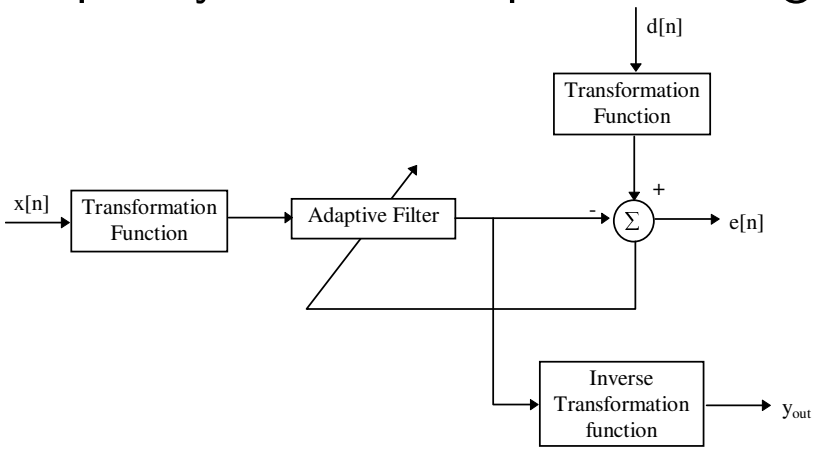
$$\begin{aligned}
\gamma(k+1) &= \mathbf{x}^T(k+1)\mathbf{P}(k)\mathbf{x}(k+1) \\
\mathbf{k}(k+1) &= \frac{\mathbf{P}(k)\mathbf{x}(k+1)}{1+\gamma(k+1)} \\
\mathbf{e}(k+1) &= \mathbf{d}(k+1) - \mathbf{x}^T(k+1)\mathbf{w}(k+1) \\
\mathbf{w}(k+1) &= \mathbf{w}(k)+\mathbf{k}(k+1)\mathbf{e}(k+1) = \mathbf{w}(k)+\mathbf{R}^{-1}(k+1)\mathbf{x}(k+1)\mathbf{e}(k+1) \\
\mathbf{P}(k+1) &= \mathbf{P}(k) - \mathbf{k}(k+1)\mathbf{x}^T(k)\mathbf{P}(k)
\end{aligned}$$

- Computational complexity of RLS is an $\mathcal{O}(p^2)$ in terms of multiplications and additions as compared to LMS and NLMS which are $\mathcal{O}(2N)$

Appendix: Transform domain signal processing

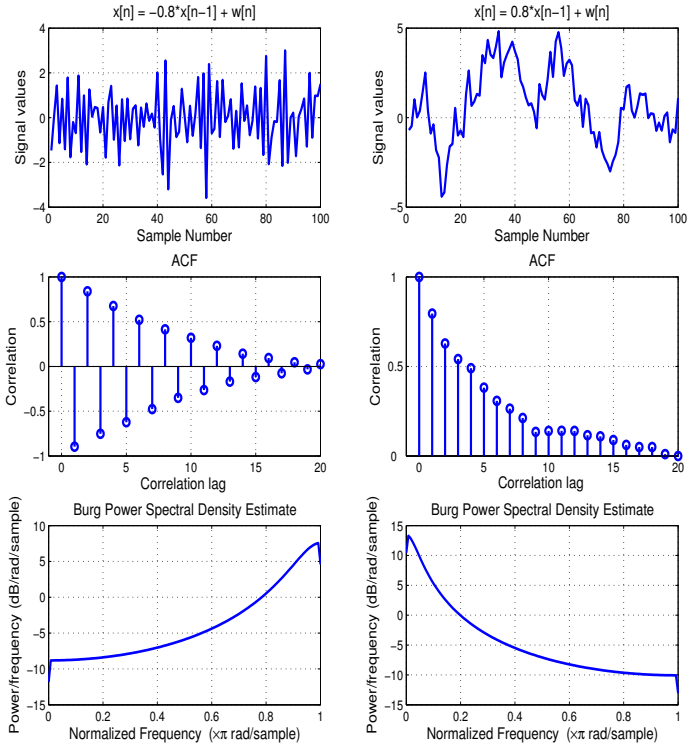
Usually complex valued (e.g. Fourier Domain)

Frequency domain adaptive filtering



A transform can be applied to the inputs of an adaptive filter in order to maximise the performance of the LMS algorithm, i.e. a white input with equal eigenvalues.

Various techniques can be used such as Lattice Filters, the FFT which requires the Complex LMS algorithm, the Discrete Cosine or Wavelet transforms, or sub-band filters.



$a < 0 \rightarrow$ highpass, $a > 0 \rightarrow$ lowpass

$$x(k) = a_1 x(k-1) + w(k)$$

Highpass signal: fast changing in time $\not\rightarrow$ however, smooth spectrum

Appendix: The CLMS algorithm

Step-by-step derivation

Consider a complex valued FIR filter.

The weight update equation for a real valued filter is

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \eta e(k)\mathbf{x}(k)$$

Now, the weights and errors and teaching signal and input are complex valued.

Hence

$$\begin{aligned} e(k) &= e_r(k) + \jmath e_i(k) \\ d(k) &= d_r(k) + \jmath d_i(k) \\ \mathbf{x}(k) &= \mathbf{x}_r(k) + \jmath \mathbf{x}_i(k) \\ \mathbf{w}(k) &= \mathbf{w}_r(k) + \jmath\ \mathbf{w}_i(k) \\ y(k) &= \mathbf{x}^T(k)\mathbf{w}(k) \end{aligned}$$

Appendix: The CLMS cost function

The complex LMS should simultaneously adapt the real and imaginary part, minimising in some sense both $e_r(k)$ and $e_i(k)$, with respect to the average total error power, given by

$$E\left[e(k)e^*(k)\right] = \frac{1}{2}E\left[e_r^2(k) + e_i^2(k)\right] = \frac{1}{2}E\left[e_r^2(k)\right] + \frac{1}{2}E\left[e_i^2(k)\right]$$

Since the two components of the error are in quadrature relative to each other, they cannot be minimised independently.

The derivation of the complex LMS is similar to the derivation fo the original LMS, except that the rules of complex algebra must be observed.

Notice that $\left(\mathbf{x}^T(k)\mathbf{w}(k)\right)^* = \left(\mathbf{x}^*(k)\right)^T \mathbf{w}^*(k)$, e.g.

$x \times w = [(x_r + \jmath x_i)(w_r + \jmath w_i)] = [x_r w_r - x_i w_i + \jmath(x_r w_i + x_i w_r)]$ After conjugation we have $[x_r w_r - x_i w_i - \jmath(x_r w_i + x_i w_r)]$. On the other hand $x^* w^* = [(x_r - \jmath x_i)(w_r - \jmath w_i] = [x_r w_r - x_i w_i - \jmath(x_r w_i + x_i w_r)]$,

which is the same as when we conjugate the whole expression.

Appendix: The CLMS Derivation

$$e^*(k) = d^*(k) - \left(\mathbf{x}^T(k)\right)^* \mathbf{w}^*(k)$$

Therefore, for a GD adaptation we have

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \frac{1}{2}\eta \nabla_{\mathbf{w}} \left(e(k)e^*(k)\right)$$

where $$\nabla(e(k)e^*(k)) = \nabla_r(e(k)e^*(k)) + \jmath\nabla_i(e(k)e^*(k)).$$

Now,

$$\nabla_r(e(k)e^*(k)) = \begin{bmatrix} \frac{\partial(e(k)e^*(k))}{\partial w_{1r}} \\ \frac{\partial(e(k)e^*(k))}{\partial w_{2r}} \\ \vdots \\ \frac{\partial(e(k)e^*(k))}{\partial w_{Nr}} \end{bmatrix} = e(k)\nabla_r(e^*(k)) + e^*(k)\nabla_r(e(k))$$

Appendix: The CLMS derivation – contd.

Notice that (in a simplified way)

$$\begin{aligned} e(k) &= d(k) - x(k)w(k) = d(k) - \\ &- [x_r(k)w_r(k) - x_i(k)w_i(k) + \jmath(x_r(k)w_i(k) + x_i(k)w_r(k))] \\ e^*(k) &= d(k) - [x_r(k)w_r(k) + x_i(k)w_i(k) - \jmath(x_r(k)w_i(k) + x_i(k)w_r(k))] \end{aligned}$$

The partial derivatives wrt to w_r and w_i are

$$\begin{aligned} \frac{\partial e(k)}{\partial w_r(k)} &= -[x_r(k) + \jmath x_i(k)] = -\mathbf{x}(k) \\ \frac{\partial e(k)}{\partial w_i(k)} &= -[-x_i(k) + \jmath x_r(k)] = -\jmath\mathbf{x}(k) \\ \frac{\partial e^*(k)}{\partial w_r(k)} &= -[x_r(k) - \jmath x_i(k)] = -\mathbf{x}^*(k) \\ \frac{\partial e^*(k)}{\partial w_i(k)} &= -[x_i(k) - \jmath x_r(k)] = -[-\jmath\mathbf{x}^*(k)] \end{aligned}$$

Appendix: The CLMS Derivation – Complex Gradients

The instantaneous gradient with respect to its real and imaginary component becomes

$$\begin{aligned}
\nabla_r(e(k)e^*(k)) &= e(k)\nabla_r(e^*(k)) + e^*(k)\nabla_r(e(k)) \\
&= e(k)(-\mathbf{x}^*(k)) + e^*(k)(-\mathbf{x}(k)) \\
\nabla_i(e(k)e^*(k)) &= e(k)\nabla_i(e^*(k)) + e^*(k)\nabla_i(e(k)) \\
&= e(k)\left(\jmath\mathbf{x}^*(k)\right) + e^*(k)\left(-\jmath\mathbf{x}(k)\right)
\end{aligned}$$

Now, applying the method of steepest descent, to the real and imaginary part of the weights we have

$$\begin{aligned}
\mathbf{w}_r(k+1) &= \mathbf{w}_r(k) - \frac{1}{2}\eta\nabla_r\left(e(k)e^*(k)\right) \\
\mathbf{w}_i(k+1) &= \mathbf{w}_i(k) - \frac{1}{2}\eta\nabla_i\left(e(k)e^*(k)\right)
\end{aligned}$$

Appendix: The CLMS Derivation – Filter Update

Having in mind that $\mathbf{w}(k+1) = \mathbf{w}_r(k+1) + \jmath\mathbf{w}_i(k+1)$, we have

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \frac{1}{2}\eta\left[\nabla_r\left(e(k)e^*(k)\right) + \jmath\nabla_i\left(e(k)e^*(k)\right)\right]$$

If the gradients are now substituted in the above equation, we have

$$\begin{aligned}
\nabla_r\left(e(k)e^*(k)\right) + \jmath\nabla_i\left(e(k)e^*(k)\right) &= \\
e(k)(-\mathbf{x}^*(k)) + e^*(k)(-\mathbf{x}(k)) + \jmath\left[e(k)\left(\jmath\mathbf{x}^*(k)\right) + e^*(k)\left(-\jmath\mathbf{x}(k)\right)\right] &= \\
-e_r\mathbf{x}^* - \jmath e_i\mathbf{x}^* - e_r\mathbf{x} + \jmath e_i\mathbf{x} - e_r\mathbf{x}^* - \jmath e_i\mathbf{x}^* + e_r\mathbf{x} - \jmath e_i\mathbf{x} &= \\
-2e_r\mathbf{x}^* - 2\jmath e_i\mathbf{x}^* = -2e(k)\mathbf{x}^*(k)&
\end{aligned}$$

Therefore, the complex form of the LMS algorithm is given by

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \eta e(k)\mathbf{x}^*(k)$$

Notes

○