The Complex Gaussian Kernel LMS algorithm

A unified framework for complex signal processing in RKHS

P. Bouboulis[1] S. Theodoridis[1]

[1]Department of Informatics and Telecommunications
University of Athens Greece

16-09-2010

Outline

Outline

Processing in RKHS

Processing in Reproducing Kernel Hilbert Spaces is gaining in popularity within the Machine Learning and Signal Processing Communities:

Processing in RKHS

Processing in Reproducing Kernel Hilbert Spaces is gaining in popularity within the Machine Learning and Signal Processing Communities:

Basic Steps:

Processing in RKHS

Processing in Reproducing Kernel Hilbert Spaces is gaining in popularity within the Machine Learning and Signal Processing Communities:

Basic Steps:

1. Map the finite dimensionality input data from the input space F into a higher dimensionality RKHS $\mathcal{H}$.

Processing in RKHS

Processing in Reproducing Kernel Hilbert Spaces is gaining in popularity within the Machine Learning and Signal Processing Communities:

Basic Steps:

1. Map the finite dimensionality input data from the input space F into a higher dimensionality RKHS $\mathcal{H}$.
2. Perform a linear processing (e.g., adaptive filtering) on the mapped data in $\mathcal{H}$.

Processing in RKHS

Processing in Reproducing Kernel Hilbert Spaces is gaining in popularity within the Machine Learning and Signal Processing Communities:

Basic Steps:

1. Map the finite dimensionality input data from the input space F into a higher dimensionality RKHS $\mathcal{H}$.
2. Perform a linear processing (e.g., adaptive filtering) on the mapped data in $\mathcal{H}$.

This procedure is equivalent with a non linear processing in F.

Reproducing Kernel Hilbert Spaces.

Consider a linear class $\mathcal{H}$ of real valued functions f defined on a set X (in particular $\mathcal{H}$ is a Hilbert space), for which there exists a function $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ with the following two properties:

Reproducing Kernel Hilbert Spaces.

Consider a linear class $\mathcal{H}$ of real valued functions f defined on a set X (in particular $\mathcal{H}$ is a Hilbert space), for which there exists a function $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ with the following two properties:

1. For every $x \in \mathcal{X}$, $\kappa(x, \cdot)$ belongs to $\mathcal{H}$.

Reproducing Kernel Hilbert Spaces.

Consider a linear class $\mathcal{H}$ of real valued functions f defined on a set X (in particular $\mathcal{H}$ is a Hilbert space), for which there exists a function $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ with the following two properties:

1. For every $x \in \mathcal{X}$, $\kappa(x, \cdot)$ belongs to $\mathcal{H}$.
2. κ has the so called reproducing property, i.e.,

$$f(x) = \langle f, \kappa(x, \cdot) \rangle_{\mathcal{H}}, \text{ for all } f \in \mathcal{H},\ x \in \mathcal{X}. \tag{1}$$

Kernel Trick

In particular, If

Kernel Trick

In particular, If
$\mathcal{X} \ni x \rightarrow \Phi(x) := \kappa(x, \cdot) \in \mathcal{H}$
$\mathcal{X} \ni y \rightarrow \Phi(y) := \kappa(y, \cdot) \in \mathcal{H}$,
then the inner product in $\mathcal{H}$ is given as a function computed on $\mathcal{X}$:

$$\kappa(x, y) = \langle \kappa(x, \cdot), \kappa(y, \cdot) \rangle_{\mathcal{H}} \qquad \text{kernel trick}$$

Advantages

Advantages of kernel-based signal processing:

Advantages

Advantages of kernel-based signal processing:

- The original nonlinear task is transformed into a linear one.

Advantages

Advantages of kernel-based signal processing:

- The original nonlinear task is transformed into a linear one.
- Different types of nonlinearities can be treated in a unified way.

Developing Algorithms in RKHS

- The black box approach.

Developing Algorithms in RKHS

- The black box approach.
  - Develop the Algorithm in $\mathcal{X}$.

Developing Algorithms in RKHS

- The black box approach.
  - Develop the Algorithm in $\mathcal{X}$.
  - Express it, **if possible**, in inner products.

Developing Algorithms in RKHS

- The black box approach.
  - Develop the Algorithm in $\mathcal{X}$.
  - Express it, **if possible**, in inner products.
  - Replace inner products with kernel evaluations according to the kernel trick.

Developing Algorithms in RKHS

- The black box approach.
  - Develop the Algorithm in $\mathcal{X}$.
  - Express it, **if possible**, in inner products.
  - Replace inner products with kernel evaluations according to the kernel trick.
- Work directly in the RKHS, assuming that the data have been mapped and live in the RKHS $\mathcal{H}$, i.e.,

$$\mathcal{X} \ni \boldsymbol{x} \to \Phi(\boldsymbol{x}) := \kappa(\boldsymbol{x}, \cdot) \in \mathcal{H}.$$

The problem

The major task of this research:

The problem

The major task of this research:

Development of a unified framework for complex valued signal processing in RKHS.

Outline

1. Signal Processing with Kernels
   - Preliminaries
   - Kernel LMS
2. Wirtinger's Calculus
   - The Complex Case: Wirtinger's Calculus
   - Wirtinger's Calculus in complex RKHS
3. Complex Gaussian Kernel LMS
   - Formulation
   - Sparsification
   - Experiments

LMS

Consider the sequence of examples
$(\boldsymbol{x}(1), d(1)), (\boldsymbol{x}(2), d(2)), \ldots, (\boldsymbol{x}(N), d(N))$:

LMS

Consider the sequence of examples
$(\boldsymbol{x}(1), d(1)), (\boldsymbol{x}(2), d(2)), \dots, (\boldsymbol{x}(N), d(N))$:

- In a typical LMS filter the goal is to learn a linear input output mapping $f : X \to \mathbb{R} : f(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x}$, so that to minimize the square error $E[|d(n) - \boldsymbol{w}^T \boldsymbol{x}(n)|^2]$.

LMS

Consider the sequence of examples $(\boldsymbol{x}(1), d(1)), (\boldsymbol{x}(2), d(2)), \ldots, (\boldsymbol{x}(N), d(N))$:

- In a typical LMS filter the goal is to learn a linear input output mapping $f : X \to \mathbb{R} : f(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x}$, so that to minimize the square error $E[|d(n) - \boldsymbol{w}^T \boldsymbol{x}(n)|^2]$.
- Using the derivative of the cost, the gradient descent update rule becomes: $\boldsymbol{w}(n) = \boldsymbol{w}(n-1) + \mu e(n) \boldsymbol{x}(n)$.

LMS

Consider the sequence of examples
$(\boldsymbol{x}(1), d(1)), (\boldsymbol{x}(2), d(2)), \ldots, (\boldsymbol{x}(N), d(N))$:

- In a typical LMS filter the goal is to learn a linear input output mapping $f : X \to \mathbb{R} : f(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x}$, so that to minimize the square error $E[|d(n) - \boldsymbol{w}^T \boldsymbol{x}(n)|^2]$.
- Using the derivative of the cost, the gradient descent update rule becomes: $\boldsymbol{w}(n) = \boldsymbol{w}(n-1) + \mu e(n)\boldsymbol{x}(n)$.
- The desired output becomes $\hat{d}(n) = \boldsymbol{w}(n-1)^T \boldsymbol{x}(n) = \mu \sum_{k=1}^{n-1} e(k) \boldsymbol{x}(k)^T \boldsymbol{x}(n)$.

Kernel LMS

- In Kernel LMS, firstly we transform the input space to a RKHS $\mathcal{H}$ to obtain the sequence: $(\Phi(\boldsymbol{x}(1)), d(1)), (\Phi(\boldsymbol{x}(2)), d(2)), \dots, (\Phi(\boldsymbol{x}(N)), d(N))$.

Kernel LMS

- In Kernel LMS, firstly we transform the input space to a RKHS $\mathcal{H}$ to obtain the sequence: $(\Phi(\boldsymbol{x}(1)), d(1)), (\Phi(\boldsymbol{x}(2)), d(2)), \dots, (\Phi(\boldsymbol{x}(N)), d(N))$.
- We apply the LMS procedure to the sequence of examples minimizing the cost function $E[|d(n) - \langle \Phi(\boldsymbol{x}(n)), \boldsymbol{w} \rangle_{\mathcal{H}}|^2]$, where now $\boldsymbol{w} \in \mathcal{H}$.

Kernel LMS

- In Kernel LMS, firstly we transform the input space to a RKHS $\mathcal{H}$ to obtain the sequence: $(\Phi(\boldsymbol{x}(1)), d(1)), (\Phi(\boldsymbol{x}(2)), d(2)), \ldots, (\Phi(\boldsymbol{x}(N)), d(N))$.
- We apply the LMS procedure to the sequence of examples minimizing the cost function $E[|d(n) - \langle\Phi(\boldsymbol{x}(n)), \boldsymbol{w}\rangle_{\mathcal{H}}|^2]$, where now $\boldsymbol{w} \in \mathcal{H}$.
- Using the derivative in the RKHS the update rule for the KLMS becomes: $\boldsymbol{w}(n) = \boldsymbol{w}(n-1) + \mu e(n)\Phi(\boldsymbol{x}(n))$.

Kernel LMS

- In Kernel LMS, firstly we transform the input space to a RKHS $\mathcal{H}$ to obtain the sequence: $(\Phi(\boldsymbol{x}(1)), d(1)), (\Phi(\boldsymbol{x}(2)), d(2)), \ldots, (\Phi(\boldsymbol{x}(N)), d(N))$.
- We apply the LMS procedure to the sequence of examples minimizing the cost function $E[|d(n) - \langle \Phi(\boldsymbol{x}(n)), \boldsymbol{w} \rangle_{\mathcal{H}} |^2]$, where now $\boldsymbol{w} \in \mathcal{H}$.
- Using the derivative in the RKHS the update rule for the KLMS becomes: $\boldsymbol{w}(n) = \boldsymbol{w}(n-1) + \mu e(n) \Phi(\boldsymbol{x}(n))$.
- The filter output of the KLMS is: $\hat{d}(n) = \langle \boldsymbol{x}(n), \boldsymbol{w}(n-1) \rangle_{\mathcal{H}} = \mu \sum_{k=1}^{n-1} e(k) \, \kappa(\boldsymbol{x}(k), \boldsymbol{x}(n))$.

Remark

Since he RKHS $\mathcal{H}$ can be an infinite dimensional space, the derivative has to be considered in the Fréchet generalized notion:

Remark

Since he RKHS $\mathcal{H}$ can be an infinite dimensional space, the derivative has to be considered in the Fréchet generalized notion:
An operator $T : \mathcal{H} \to F$ is said to be Fréchet differentiable at f_0, if there exists $u \in \mathcal{H}$ such that the limit

$$\lim_{\|h\|_{\mathcal{H}}} \frac{T(f_0 + h) - T(f_0) - \langle u, h \rangle_{\mathcal{H}}}{\|h\|_{\mathcal{H}}} = 0.$$

Outline

Complex and real derivatives

Consider a complex function
$f : \mathbb{C} \to \mathbb{C} : f(z) = f(x + iy) = f_r(z) + if_i(z)$.

Complex and real derivatives

Consider a complex function
$f : \mathbb{C} \to \mathbb{C} : f(z) = f(x + iy) = f_r(z) + i f_i(z)$.

We will say that f is differentiable in the complex sense at c (or that it has complex derivative at c), iff the limit

$$\lim_{z \to c} \frac{f(z) - f(c)}{z - c}$$

exists.

Remarks

- Complex differentiability is a very strict notion.

Remarks

- Complex differentiability is a very strict notion.
- In complex signal processing we often encounter functions (e.g., the cost functions, which are defined in $\mathbb{R}$) that ARE NOT complex differentiable.

Remarks

- Complex differentiability is a very strict notion.
- In complex signal processing we often encounter functions (e.g., the cost functions, which are defined in $\mathbb{R}$) that ARE NOT complex differentiable.
- Example: $f(z) = |z|^2 = zz^*$.

Remarks

- Complex differentiability is a very strict notion.
- In complex signal processing we often encounter functions (e.g., the cost functions, which are defined in $\mathbb{R}$) that ARE NOT complex differentiable.
- Example: $f(z) = |z|^2 = zz^*$.
- In these cases one has to express the cost function in terms of its real part f_r and its imaginary part f_i, and use real derivation with respect to f_r, f_i.

Wirtinger's Approach

- This approach leads usually to cumbersome and tedious calculations.

Wirtinger's Approach

- This approach leads usually to cumbersome and tedious calculations.
- Wirtinger's Calculus provides an alternative equivalent formulation.

Wirtinger's Approach

- This approach leads usually to cumbersome and tedious calculations.
- Wirtinger's Calculus provides an alternative equivalent formulation.
- It is based on simple rules and principles.

Wirtinger's Approach

- This approach leads usually to cumbersome and tedious calculations.
- Wirtinger's Calculus provides an alternative equivalent formulation.
- It is based on simple rules and principles.
- These rules bear a great resemblance to the rules of the standard complex derivative.

Wirtinger's Approach

Wirtinger's Calculus considers two forms of derivatives:

Wirtinger's Approach

Wirtinger's Calculus considers two forms of derivatives:

- The $\mathbb{R}$-derivative:
  $$\frac{\partial f}{\partial z} = \frac{1}{2}\left(\frac{\partial f_r}{\partial x} + \frac{\partial f_i}{\partial y}\right) + \frac{i}{2}\left(\frac{\partial f_i}{\partial x} - \frac{\partial f_r}{\partial y}\right),$$

Wirtinger's Approach

Wirtinger's Calculus considers two forms of derivatives:

- The $\mathbb{R}$-derivative:
  $\frac{\partial f}{\partial z} = \frac{1}{2}\left(\frac{\partial f_r}{\partial x} + \frac{\partial f_i}{\partial y}\right) + \frac{i}{2}\left(\frac{\partial f_i}{\partial x} - \frac{\partial f_r}{\partial y}\right)$,
- The conjugate $\mathbb{R}$-derivative:
  $\frac{\partial f}{\partial z^*} = \frac{1}{2}\left(\frac{\partial f_r}{\partial x} - \frac{\partial f_i}{\partial y}\right) + \frac{i}{2}\left(\frac{\partial f_i}{\partial x} + \frac{\partial f_r}{\partial y}\right)$.

Simple Rules

- Wirtinger's rules are based on the fact that any complex functions, which it is differentiable in the real sense, can be written in the form $f(z, z^*)$.

Simple Rules

- Wirtinger's rules are based on the fact that any complex functions, which it is differentiable in the real sense, can be written in the form $f(z, z^*)$.
- It can be proved that $\frac{\partial f}{\partial z}$ can be easily evaluated as the standard complex derivative taken with respect to z (thus treating z^* as a constant).

Simple Rules

- Wirtinger's rules are based on the fact that any complex functions, which it is differentiable in the real sense, can be written in the form $f(z, z^*)$.
- It can be proved that $\frac{\partial f}{\partial z}$ can be easily evaluated as the standard complex derivative taken with respect to z (thus treating z^* as a constant).
- Similarly $\frac{\partial f}{\partial z^*}$ can be easily evaluated as the standard complex derivative taken with respect to z^* (thus treating z as a constant).

Outline

Extension of Wirtinger's Calculus

- Wirtinger's Calculus can be easily extended to any finite dimensional complex space (i.e., $\mathbb{C}^{\nu}$).

Extension of Wirtinger's Calculus

- Wirtinger's Calculus can be easily extended to any finite dimensional complex space (i.e., $\mathbb{C}^\nu$).
- The main rules and principles are similar.

Extension of Wirtinger's Calculus

- Wirtinger's Calculus can be easily extended to any finite dimensional complex space (i.e., $\mathbb{C}^\nu$).
- The main rules and principles are similar.
- In order to extend it to a complex RKHS (where the dimensionality can be infinite), we need to employ the notion of Fréchet differentiability.

Wirtinger's derivatives in RKHS

- Consider a complex RKHS $\mathbb{H}$ and a complex operator $\boldsymbol{T} = T_r + iT_i$, where T_r, T_i are defined on $\mathbb{H}$.

Wirtinger's derivatives in RKHS

- Consider a complex RKHS $\mathbb{H}$ and a complex operator $\boldsymbol{T} = T_r + iT_i$, where T_r, T_i are defined on $\mathbb{H}$.
- Let $\nabla_r T_r$, $\nabla_r T_i$, $\nabla_i T_r$, $\nabla_i T_i$ be the respective Fréchet derivatives (gradients).

Wirtinger's derivatives in RKHS

- Consider a complex RKHS $\mathbb{H}$ and a complex operator $\boldsymbol{T} = T_r + iT_i$, where T_r, T_i are defined on $\mathbb{H}$.
- Let $\nabla_r T_r$, $\nabla_r T_i$, $\nabla_i T_r$, $\nabla_i T_i$ be the respective Fréchet derivatives (gradients).
- We can define the respective $\mathbb{R}$-derivative and conjugate $\mathbb{R}$-derivative of $\boldsymbol{T}$ as follows:

Wirtinger's derivatives in RKHS

- Consider a complex RKHS $\mathbb{H}$ and a complex operator $\boldsymbol{T} = T_r + iT_i$, where T_r, T_i are defined on $\mathbb{H}$.
- Let $\nabla_r T_r$, $\nabla_r T_i$, $\nabla_i T_r$, $\nabla_i T_i$ be the respective Fréchet derivatives (gradients).
- We can define the respective $\mathbb{R}$-derivative and conjugate $\mathbb{R}$-derivative of $\boldsymbol{T}$ as follows:
- $\mathbb{R}$-derivative:

$$\nabla_{\boldsymbol{f}} \boldsymbol{T} = \frac{1}{2}\left(\nabla_r T_r + \nabla_i T_i\right) + \frac{i}{2}\left(\nabla_r T_i - \nabla_i T_r\right).$$

Wirtinger's derivatives in RKHS

- Consider a complex RKHS $\mathbb{H}$ and a complex operator $\boldsymbol{T} = T_r + iT_i$, where T_r, T_i are defined on $\mathbb{H}$.
- Let $\nabla_r T_r$, $\nabla_r T_i$, $\nabla_i T_r$, $\nabla_i T_i$ be the respective Fréchet derivatives (gradients).
- We can define the respective $\mathbb{R}$-derivative and conjugate $\mathbb{R}$-derivative of $\boldsymbol{T}$ as follows:
- $\mathbb{R}$-derivative:

$$\nabla_{\boldsymbol{f}} \boldsymbol{T} = \frac{1}{2}\left(\nabla_r T_r + \nabla_i T_i\right) + \frac{i}{2}\left(\nabla_r T_i - \nabla_i T_r\right).$$

- conjugate $\mathbb{R}$-derivative:

$$\nabla_{\boldsymbol{f}^*} \boldsymbol{T} = \frac{1}{2}\left(\nabla_r T_r - \nabla_i T_i\right) + \frac{i}{2}\left(\nabla_r T_i + \nabla_i T_r\right).$$

Rules and Properties

Several rules and properties of the ordinary Wirtinger's Calculus can be easily extended:

Rules and Properties

Several rules and properties of the ordinary Wirtinger's Calculus can be easily extended:

- If $\boldsymbol{T}$ is $\boldsymbol{f}$-holomorphic (i.e., it has a Taylor series expansion with respect to $\boldsymbol{f}$), then $\nabla_{\boldsymbol{f}^*}\boldsymbol{T} = \mathbf{0}$.

Rules and Properties

Several rules and properties of the ordinary Wirtinger's Calculus can be easily extended:

- If $\boldsymbol{T}$ is $\boldsymbol{f}$-holomorphic (i.e., it has a Taylor series expansion with respect to $\boldsymbol{f}$), then $\nabla_{\boldsymbol{f}^*}\boldsymbol{T} = \boldsymbol{0}$.
- If $\boldsymbol{T}$ is $\boldsymbol{f}^*$-holomorphic (i.e., it has a Taylor series expansion with respect to $\boldsymbol{f}^*$), then $\nabla_{\boldsymbol{f}}\boldsymbol{T} = \boldsymbol{0}$.

Rules and Properties

Several rules and properties of the ordinary Wirtinger's Calculus can be easily extended:

- If $\boldsymbol{T}$ is $\boldsymbol{f}$-holomorphic (i.e., it has a Taylor series expansion with respect to $\boldsymbol{f}$), then $\nabla_{\boldsymbol{f}^*}\boldsymbol{T} = \mathbf{0}$.
- If $\boldsymbol{T}$ is $\boldsymbol{f}^*$-holomorphic (i.e., it has a Taylor series expansion with respect to $\boldsymbol{f}^*$), then $\nabla_{\boldsymbol{f}}\boldsymbol{T} = \mathbf{0}$.
- $(\nabla_{\boldsymbol{f}}\boldsymbol{T})^* = \nabla_{\boldsymbol{f}^*}\boldsymbol{T}^*$.

Rules and Properties

Several rules and properties of the ordinary Wirtinger's Calculus can be easily extended:

- If $\boldsymbol{T}$ is $\boldsymbol{f}$-holomorphic (i.e., it has a Taylor series expansion with respect to $\boldsymbol{f}$), then $\nabla_{\boldsymbol{f}^*}\boldsymbol{T} = \mathbf{0}$.
- If $\boldsymbol{T}$ is $\boldsymbol{f}^*$-holomorphic (i.e., it has a Taylor series expansion with respect to $\boldsymbol{f}^*$), then $\nabla_{\boldsymbol{f}}\boldsymbol{T} = \mathbf{0}$.
- $(\nabla_{\boldsymbol{f}}\boldsymbol{T})^* = \nabla_{\boldsymbol{f}^*}\boldsymbol{T}^*$.
- $(\nabla_{\boldsymbol{f}^*}\boldsymbol{T})^* = \nabla_{\boldsymbol{f}}\boldsymbol{T}^*$.

Rules and Properties

Any gradient descent based algorithm minimizing a real valued operator $\boldsymbol{T}(\boldsymbol{f})$ is based on the update scheme:

$$\boldsymbol{f}_n = \boldsymbol{f}_{n-1} - \mu \cdot \nabla_{\boldsymbol{f}^*} \boldsymbol{T}(\boldsymbol{f}_{n-1}).$$

Rules and Properties

Any gradient descent based algorithm minimizing a real valued operator $\boldsymbol{T}(\boldsymbol{f})$ is based on the update scheme:

$$\boldsymbol{f}_n = \boldsymbol{f}_{n-1} - \mu \cdot \nabla_{\boldsymbol{f}^*} \boldsymbol{T}(\boldsymbol{f}_{n-1}).$$

Remark: We have used $\boldsymbol{f}$ in place of $\boldsymbol{w}$ (used before) to stress the fact that the RKHS $\mathbb{H}$ can be of infinite dimension.

Outline

Mapping to the complex RKHS

- Consider the sequence of examples $(\boldsymbol{z}(1), d(1)), (\boldsymbol{z}(2), d(2)), \ldots (\boldsymbol{z}(N), d(N))$, where $d(n) \in \mathbb{C}$ and $\boldsymbol{z}(n) \in \mathbb{C}^{\nu}$

Mapping to the complex RKHS

- Consider the sequence of examples $(\boldsymbol{z}(1), d(1)), (\boldsymbol{z}(2), d(2)), \dots (\boldsymbol{z}(N), d(N))$, where $d(n) \in \mathbb{C}$ and $\boldsymbol{z}(n) \in \mathbb{C}^{\nu}$
- Let $\boldsymbol{z}(n) = \boldsymbol{x}(n) + i\boldsymbol{y}(n)$, $\boldsymbol{x}(n), \boldsymbol{y}(n) \in \mathbb{R}^{\nu}$.

Mapping to the complex RKHS

- Consider the sequence of examples $(\boldsymbol{z}(1), d(1)), (\boldsymbol{z}(2), d(2)), \dots (\boldsymbol{z}(N), d(N))$, where $d(n) \in \mathbb{C}$ and $\boldsymbol{z}(n) \in \mathbb{C}^{\nu}$
- Let $\boldsymbol{z}(n) = \boldsymbol{x}(n) + i\boldsymbol{y}(n)$, $\boldsymbol{x}(n), \boldsymbol{y}(n) \in \mathbb{R}^{\nu}$.
- We map the points $\boldsymbol{z}(n)$ to the complex RKHS $\mathbb{H}$ an appropriate complex mapping $\boldsymbol{\Phi}$.

Choice of the complex mapping $\boldsymbol{\Phi}$

- $\boldsymbol{\Phi}$ can be the result of complexifying real kernels:

$$\begin{aligned}\boldsymbol{\Phi}(\boldsymbol{z}(n)) &= \Phi(\boldsymbol{z}(n)) + i\Phi(\boldsymbol{z}(n)) \\ &= \kappa\left((\boldsymbol{x}(n), \boldsymbol{y}(n))^T, \cdot\right) + i \cdot \kappa\left((\boldsymbol{x}(n), \boldsymbol{y}(n))^T, \cdot\right),\end{aligned}$$

Choice of the complex mapping $\mathbf{\Phi}$

- $\mathbf{\Phi}$ can be the result of complexifying real kernels:

$$\begin{aligned}\mathbf{\Phi}(\boldsymbol{z}(n)) &= \Phi(\boldsymbol{z}(n)) + i\Phi(\boldsymbol{z}(n)) \\ &= \kappa\left((\boldsymbol{x}(n), \boldsymbol{y}(n))^T, \cdot\right) + i \cdot \kappa\left((\boldsymbol{x}(n), \boldsymbol{y}(n))^T, \cdot\right),\end{aligned}$$

- $\mathbf{\Phi}$ can be any complex kernel, e.g. the complex Gaussian kernel,

$$\kappa_{\sigma,\mathbb{C}^d}(x, y) = \exp\left(-\frac{\sum_{i=1} d(z_i - w_i^*)^2}{\sigma^2}\right)$$

Note that exp is the complex exponential function, i.e.,

$$\exp(z) = \sum_{n=1}^{\infty} \frac{z^n}{n!}$$

Complex Gaussian Kernel LMS

- We apply the complex LMS to the transformed data: $(\boldsymbol{\Phi}(\boldsymbol{z}(1)), d(1)), (\boldsymbol{\Phi}(\boldsymbol{z}(2)), d(2)), \dots (\boldsymbol{\Phi}(\boldsymbol{z}(N)), d(N))$.

Complex Gaussian Kernel LMS

- We apply the complex LMS to the transformed data: $(\boldsymbol{\Phi}(\boldsymbol{z}(1)), d(1)), (\boldsymbol{\Phi}(\boldsymbol{z}(2)), d(2)), \dots (\boldsymbol{\Phi}(\boldsymbol{z}(N)), d(N))$.
- The objective of CKLMS is to minimize

$$E[|e(n)|^2] = E[|d(n) - \langle \boldsymbol{\Phi}(\boldsymbol{z}(n), \boldsymbol{f}\rangle_{\mathbb{H}}^2],$$

at each instance n.

Complex Gaussian Kernel LMS

Using the rules of Wirtinger's calculus in $\mathbb{H}$ we obtain the following update rule:

Complex Gaussian Kernel LMS

Using the rules of Wirtinger's calculus in $\mathbb{H}$ we obtain the following update rule:

$$\boldsymbol{f}(n) = \boldsymbol{f}(n-1) + \mu e(n)^* \cdot \boldsymbol{\Phi}(\boldsymbol{z}(n)),$$

where $\boldsymbol{f}(n)$ denotes the estimate at iteration n.

Complex Gaussian Kernel LMS

Assuming that $\boldsymbol{f}(0) = \mathbf{0}$, the repeated application of the weight-update equation gives:

Complex Gaussian Kernel LMS

Assuming that $\boldsymbol{f}(0) = \mathbf{0}$, the repeated application of the weight-update equation gives:

$$\begin{aligned}\boldsymbol{f}(n) =& \boldsymbol{f}(n-1) + \mu e(n)^* \boldsymbol{\Phi}(\boldsymbol{z}(n)) \\ =& \boldsymbol{f}(n-2) + \mu e(n-1)^* \boldsymbol{\Phi}(\boldsymbol{z}(n-1)) \\ & + \mu e(n)^* \boldsymbol{\Phi}(\boldsymbol{z}(n)) \\ =& \sum_{k=1}^{n} e(k)^* \boldsymbol{\Phi}(\boldsymbol{z}(k)).\end{aligned}$$

Complex Gaussian Kernel LMS

The filter output at iteration *n* becomes:

Complex Gaussian Kernel LMS

The filter output at iteration *n* becomes:

$$\begin{aligned}\hat{d}(n) &= \langle \boldsymbol{\Phi}(\boldsymbol{z}(n)), (\mathbf{n-1}) \rangle_{\mathbb{H}} \\ &= \mu \sum_{k=1}^{n-1} e(k) \langle \boldsymbol{\Phi}(\boldsymbol{z}(n)), \boldsymbol{\Phi}(\boldsymbol{z}(k)) \rangle_{\mathbb{H}} \\ &= \mu \sum_{k=1}^{n-1} e(k) \kappa_{\sigma,\mathbb{C}^d}(\boldsymbol{z}(n), \boldsymbol{z}(k))\end{aligned}$$

Outline

Sparsification

- CKLMS and other kernel based adaptive filtering algorithms require a growing network of training centers $\boldsymbol{z}(0), \boldsymbol{z}(1), \ldots, \boldsymbol{z}(n), \ldots$.

Sparsification

- CKLMS and other kernel based adaptive filtering algorithms require a growing network of training centers $\boldsymbol{z}(0), \boldsymbol{z}(1), \ldots, \boldsymbol{z}(n), \ldots$.
- Results: Increasing memory and computational requirements.

Sparsification

- CKLMS and other kernel based adaptive filtering algorithms require a growing network of training centers $\boldsymbol{z}(0), \boldsymbol{z}(1), \ldots, \boldsymbol{z}(n), \ldots$.
- Results: Increasing memory and computational requirements.
- A sparse solution is needed.
- Any sparsification algorithm can be employed. Details are given in the paper.

Outline

Experiments

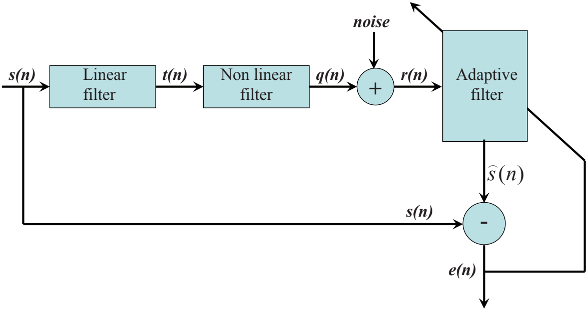


Figure: The equalization problem.

Experiments

- $t(n) = (-0.9 + 0.8i) \cdot s(n) + (0.6 - 0.7i) \cdot s(n-1)$

Experiments

- $t(n) = (-0.9 + 0.8i) \cdot s(n) + (0.6 - 0.7i) \cdot s(n-1)$
- $q(n) = t(n) + (0.1 + 0.15i) \cdot t^2(n) + (0.06 + 0.05i) \cdot t^3(n)$

Experiments

- $t(n) = (-0.9 + 0.8i) \cdot s(n) + (0.6 - 0.7i) \cdot s(n-1)$
- $q(n) = t(n) + (0.1 + 0.15i) \cdot t^2(n) + (0.06 + 0.05i) \cdot t^3(n)$
- $s(n) = 0.70(\sqrt{1-\rho^2}X(n) + i\rho Y(n))$, where $X(n)$ and $Y(n)$ are gaussian random variables.

Experiments

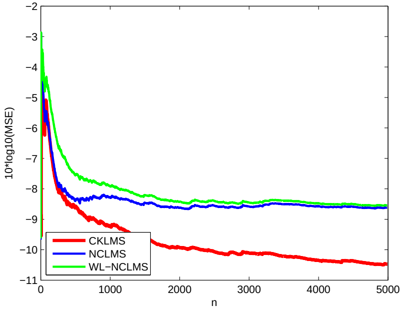
- $t(n) = (-0.9 + 0.8i) \cdot s(n) + (0.6 - 0.7i) \cdot s(n-1)$
- $q(n) = t(n) + (0.1 + 0.15i) \cdot t^2(n) + (0.06 + 0.05i) \cdot t^3(n)$
- $s(n) = 0.70(\sqrt{1-\rho^2}X(n) + i\rho Y(n))$, where $X(n)$ and $Y(n)$ are gaussian random variables.
  1. This input is circular for $\rho = \sqrt{2}/2$

Experiments

- $t(n) = (-0.9 + 0.8i) \cdot s(n) + (0.6 - 0.7i) \cdot s(n-1)$
- $q(n) = t(n) + (0.1 + 0.15i) \cdot t^2(n) + (0.06 + 0.05i) \cdot t^3(n)$
- $s(n) = 0.70(\sqrt{1-\rho^2}X(n) + i\rho Y(n))$, where $X(n)$ and $Y(n)$ are gaussian random variables.
  1. This input is circular for $\rho = \sqrt{2}/2$
  2. highly non-circular if ρ approaches 0 or 1.

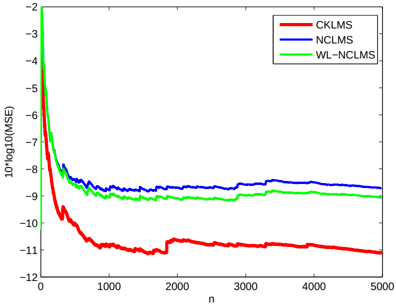
Circular Data

Learning curves for KNCLMS ($\mu = 1/2$), NCLMS ($\mu = 1/16$) and WL-NCLMS ($\mu = 1/16$) (filter length $L = 5$, delay $D = 2$) in the nonlinear channel equalization, for the **circular** input case.



Non Circular Data

Learning curves for KNCLMS ($\mu = 1/2$), NCLMS ($\mu = 1/16$) and WL-NCLMS ($\mu = 1/16$) (filter length $L = 5$, delay $D = 2$) in the nonlinear channel equalization, for the **non-circular** input case ($\rho = 0.1$).



Conclusions

Main contributions of this work:

Conclusions

Main contributions of this work:

1. The development of a wide framework that allows the development of complex-valued kernel algorithms.

Conclusions

Main contributions of this work:

1. The development of a wide framework that allows the development of complex-valued kernel algorithms.
2. The extension of Wirtinger's Calculus in complex RKHS as a means for the elegant and efficient computations of gradients that are involved in many adaptive filtering algorithms.

Conclusions

Main contributions of this work:

1. The development of a wide framework that allows the development of complex-valued kernel algorithms.
2. The extension of Wirtinger's Calculus in complex RKHS as a means for the elegant and efficient computations of gradients that are involved in many adaptive filtering algorithms.
3. The development of the Complex Gaussian Kernel LMS algorithm as a particular example.

Conclusions

Main contributions of this work:

1. The development of a wide framework that allows the development of complex-valued kernel algorithms.
2. The extension of Wirtinger's Calculus in complex RKHS as a means for the elegant and efficient computations of gradients that are involved in many adaptive filtering algorithms.
3. The development of the Complex Gaussian Kernel LMS algorithm as a particular example.
    - Experiments verify that CKLMS gives significantly better results compared to CLMS and WL-CLMS for nonlinear channels.