

UNIVERSIDADE FEDERAL DO ABC

ENGENHARIA DE INFORMAÇÃO

Estudo de Critérios baseados em Aprendizagem por  
Teoria da Informação com Kernel Epanechnikov

Trabalho de Graduação III

Lucas Henrique Gois de Campos

Centro de Engenharia, Modelagem e Ciências Sociais Aplicadas

Santo André

2020

Lucas Henrique Gois de Campos

# **Estudo de Critérios Baseados em Aprendizagem por Teoria da Informação com Kernel Epanechnikov**

Trabalho de Graduação III apresentado ao Curso de Engenharia de Informação da Universidade Federal do ABC como requisito parcial para obtenção do título de Engenheiro de Informação

Orientadora: Dra. Aline de Oliveira Neves Panazio

Santo André

2020

## **Agradecimentos**

Agradeço meus pais, Jorge e Raquel, e minha irmã, Marianna, por todo o apoio, pela ajuda, pelo incentivo nos momentos difíceis e por compreenderem a minha ausência enquanto eu me dedicava à realização deste trabalho.

Aos meus amigos que sempre estiveram ao meu lado, pela amizade incondicional e pelo apoio demonstrado ao longo do curso.

À Prof<sup>a</sup> Aline, pela oportunidade de trabalhar com alguém tão inteligente e dedicada, por toda sabedoria e conselhos que recebi, pela ajuda e pela paciência com que guiou o meu aprendizado.

À UFABC e aos professores, essenciais no meu processo de formação profissional, pela dedicação, e por tudo o que aprendi ao longo dos anos do curso.

## Resumo

A equalização baseada em aprendizagem por Teoria de Informação (ITL) é capaz de lidar com problemas nos quais técnicas clássicas encontram dificuldade, como no caso da equalização de canais quando o sinal transmitido é correlacionado. Isso se deve a capacidade dos novos critérios de utilizar a informação dos sinais envolvidos de forma mais completa, explorando suas características estatísticas no processo. Estes métodos, no entanto, necessitam da estimação da função densidade de probabilidade dos sinais em tempo real, um processo que é usualmente feito com estimadores baseados no uso de *kernels*. Na literatura, a função *kernel* frequentemente utilizada é a Gaussiana, porém esta função não é considerada ótima em relação a eficiência. Neste trabalho é proposto o uso do *kernel* Epanechnikov para a estimação, assim mostramos como o desempenho dos algoritmos pode melhorar com esta nova função.

Para realizar este estudo, foram considerados algoritmos baseados em ITL. Entre eles temos o *Stochastic Quadratic Distance* (SQD) e o *Modified Stochastic Quadratic Distance* (MSQD), que buscam igualar as densidades de probabilidade do sinal transmitido e do sinal equalizado. Também analisamos a correntropia (COR), definida como uma nova medida de correlação generalizada que pode ser utilizada no processo de equalização do sinal, sendo particularmente interessante por conta das suas propriedades. Além disso, incluímos na análise o *Correlation Retrieval* (CR), que faz uso exclusivo de momentos estatísticos de segunda ordem. Através da modelagem e simulação do processo de equalização, verificou-se o desempenho destes algoritmos utilizando os *kernels* Gaussiano e Epanechnikov.

*Palavras-Chave:* Aprendizagem por Teoria de Informação, Kernel não-Gaussiano, Equalização não-supervisionada, Kernel Epanechnikov, Critério da Distância Quadrática, Correntropia, Correlation Retrieval, Estatísticas de Segunda Ordem

## Abstract

Information Theoretic Learning (ITL) based equalization is capable of dealing with cases in which classical techniques encounter difficulty, such as in the case of transmission of correlated signals. This is due to the capacity of new criteria to use the information of the signals involved in a more comprehensive manner, comparing their statistical characteristics in the process. These methods, however, require the estimation of the probability density function of the signals in real time, a process that is usually carried out with estimators based on the use of kernels. In literature, the kernel function that is frequently used is the Gaussian, yet this function is not considered optimal. In this work, the Epanechnikov kernel is proposed for the estimation, thus we show how the performance of the algorithms can improve with this new function.

To carry out this study, algorithms based on ITL were considered. Among them we have the SQD and the MSQD, which seek to equate the probability densities of the transmitted signal and the equalized signal. We also analyzed the Correntropy (COR), which uses a new measure of generalized correlation in the signal equalization process, it being particularly interesting because of its properties. Furthermore, we analyzed Correlation Retrieval (CR), which makes exclusive use of second-order statistical moments. Through the modeling and simulation of the equalization process, the performance of these algorithms using the Gaussian and Epanechnikov kernels was found.

*Keywords:* Information theoretic learning, non-Gaussian kernels, Unsupervised equalization, Epanechnikov kernel, Stochastic Quadratic Distance, Correntropy, Correlation Retrieval, Second-order statistical moments

# Lista de Figuras

1	Diagrama de blocos da estrutura de um algoritmo de equalização . . . . .	11
2	Diagrama de pólos de $c_1(z)$ . . . . .	22
3	Diagrama de pólos de $c_2(z)$ . . . . .	22
4	Simulação do SQD variando o $\sigma$ para o canal $c_1(z)$ . . . . .	23
5	Simulação do SQD variando o passo $\mu$ para o canal $c_1(z)$ . . . . .	24
6	Simulação do SQD variando o $\sigma$ para o canal $c_2(z)$ . . . . .	25
7	Simulação do SQD variando L para o canal $c_2(z)$ . . . . .	26
8	Simulação do SQD variando o passo $\mu$ para o canal $c_2(z)$ . . . . .	27
9	Simulação do SQD variando o $\sigma$ para o canal $c_2$ com pré-filtragem $f_1(z) = 1 + 0.2z^{-1}$ . . . . .	28
10	Simulação do SQD variando o $\sigma$ para o canal $c_2$ com pré-filtragem $f_2(z) = 1 + 0.5z^{-1} + 0.2z^{-2}$ . . . . .	29
11	Simulação do MSQD variando o L para o canal $c_2$ . . . . .	31
12	Simulação do MSQD variando o $\sigma$ para o canal $c_2$ . . . . .	32
13	Simulação do MSQD variando o $\sigma$ para o canal $c_2$ com pré-filtragem $f_1(z) = 1 + 0.2z^{-1}$ . . . . .	33
14	Simulação do MSQD variando o $\sigma$ para o canal $c_2$ com pré-filtragem $f_2(z) = 1 + 0.5z^{-1} + 0.2z^{-2}$ . . . . .	34
15	Diagrama de pólos do canal $m(z)$ . . . . .	40
16	Simulação das superfícies de erro, fixando $\sigma = 0.25$ . . . . .	41
17	Simulação das superfícies de erro, fixando $\sigma = 0.75$ . . . . .	42
18	Simulação das superfícies de erro, fixando $\sigma = 1$ . . . . .	43

19	Simulação das superfícies de erro, fixando $\sigma = 5$ . . . . .	44
20	Simulação das superfícies de erro, fixando $\sigma = 20$ . . . . .	45
21	Diagrama de pólos do canal $h(z)$ . . . . .	46
22	Simulação do COR variando o $\sigma$ para o canal $h(z)$ . . . . .	47
23	Simulação do COR variando o $N$ para o canal $h(z)$ . . . . .	48
24	Simulação do COR variando o $P$ para o canal $h(z)$ . . . . .	49
25	Diagrama de pólos do canal $g(z)$ . . . . .	50
26	Simulação do COR e CR com $\sigma = 1$ para o canal $g(z)$ . . . . .	51
27	Simulação do COR e CR com $\sigma = 5$ para o canal $g(z)$ . . . . .	52

# Lista de Abreviaturas e Siglas

<i>AMISE</i>	Asymptotic Mean Integrated Square Error
<i>BPSK</i>	Binary Phase-Shift Keying
<i>COR</i>	Correntropia
<i>CR</i>	Correlation Retrieval
<i>dB</i>	Decibel
<i>FIR</i>	Finite Impulse Response
<i>IIS</i>	Interferência Intersimbólica
<i>ITL</i>	Information Theoretic Learning
<i>MISE</i>	Mean Integrated Square Error
<i>MSQD</i>	Modified SQD
<i>pdf</i>	Função Densidade de Probabilidade
<i>RKHS</i>	Reproducing Kernel Hilbert Space
<i>SQD</i>	Stochastic Quadratic Distance



# Sumário

1	Introdução . . . . .	7
2	Fundamentação Teórica . . . . .	11
2.1	Sistema em Estudo . . . . .	11
2.2	Estimador Baseado em Kernel . . . . .	13
3	Critério da Distância Quadrática (SQD) . . . . .	17
3.1	Desenvolvimento do SQD com o kernel Gaussiano . . . . .	17
3.2	Desenvolvimento do SQD com o kernel Epanechnikov . . . . .	18
3.3	MSQD . . . . .	20
3.4	Resultados de Simulação . . . . .	20
3.4.1	SQD . . . . .	21
3.4.2	MSQD . . . . .	30
4	CrITÉRIOS baseados na Correntropia . . . . .	35
4.1	Desenvolvimento do método com o kernel Gaussiano . . . . .	36
4.2	Desenvolvimento do método com o kernel Epanechnikov . . . . .	36
4.3	Correlation Retrieval (CR) . . . . .	37
4.4	Comparação entre CritÉrios . . . . .	38
4.5	Análise de Desempenho dos algoritmos . . . . .	45
5	Conclusão . . . . .	53
6	Referências Bibliográficas . . . . .	56

# 1 INTRODUÇÃO

Os sistemas de comunicação digitais desempenham um papel importante na atualidade, sendo um dos pilares da sociedade moderna, que hoje é capaz de transmitir áudio, vídeo e dados de forma quase instantânea para qualquer lugar no planeta. Mais versáteis e robustos que os sistemas analógicos, o sistema digital mantém a qualidade do sinal transmitido por grandes distâncias, permitindo a correção de erros [1].

Entretanto, mesmo com a evolução dos sistemas de comunicação, ainda há muitos desafios, como por exemplo, ao transmitir um sinal, o mesmo passa por canais de transmissão não ideais que introduzem distorções e assim dificultando, ou até mesmo impossibilitando, a recuperação do sinal pelo receptor. Dessa forma, se faz importante o estudo acerca do processamento digital de sinais, que visa diminuir as distorções sofridas pelo sinal original. Uma das soluções propostas é algum tipo de processamento no receptor para compensar as distorções sofridas, como a equalização [2].

O equalizador é normalmente implementado como um filtro com resposta ao impulso finita no receptor, o qual é responsável por compensar as distorções ocasionadas pelo canal de transmissão [2]. A equalização de canais é um fator importante para os sistemas de comunicação modernos, tendo sido estudada há bastante tempo para recuperação de sinais. Assim, as técnicas utilizadas podem ser divididas em duas categorias: a equalização supervisionada e a não supervisionada, também chamada de equalização cega.

Na equalização supervisionada há a transmissão prévia de uma sequência de símbolos, conhecida pelo receptor e comumente chamada de sequência de treinamento, a qual é utilizada para ajustar os coeficientes do equalizador antes da recepção efetiva do sinal. Entretanto utilizar sequências de treinamento possui suas desvantagens, visto que este método é pouco eficiente em sistemas com muitos receptores, sendo necessário transmitir a sequência de treinamento novamente para cada receptor para ajustar os coeficientes novamente. Além disso, caso o canal seja variante no tempo, é necessário parar a transmissão para realizar a etapa de treinamento novamente.

A outra categoria é a equalização não supervisionada ou cega, que contorna as desvan-

tagens da equalização supervisionada. Neste caso, não é necessário o uso de sequências de treinamento, pois o equalizador utiliza características estatísticas do sinal transmitido, as quais já são conhecidas pelo receptor. Normalmente, este método define critérios utilizando momentos de ordem superior do sinal transmitido e recebido, assim é possível ajustar os coeficientes do equalizador através de um algoritmo do tipo gradiente descendente, buscando o ponto de mínimo do critério [2].

A equalização cega aproveita melhor a capacidade do canal, sem a necessidade de uma sequência de treinamento. Entretanto, critérios clássicos de equalização cega também possuem certas desvantagens em relação à equalização supervisionada como, por exemplo, a velocidade de convergência do algoritmo tende a ser mais lenta do que na equalização supervisionada, por isso pode perder parte considerável das informações transmitidas, além de não possuírem um bom desempenho frente a sinais correlacionados.

Recentemente, critérios baseados em teoria de informação [3] têm sido utilizados no campo de equalização cega. Como tais critérios utilizam mais informação sobre os sinais disponíveis, acabam por conseguir resolver casos que as técnicas clássicas não conseguem. Os critérios e algoritmos desenvolvidos neste contexto são chamados de aprendizagem por teoria da informação (ITL - Information Theoretic Learning).

Dentre os critérios de equalização baseados em ITL, temos o Critério da Distância Quadrática (SQD), que busca comparar as funções densidade de probabilidade dos sinais transmitido e recebido. A partir deste critério, também temos o MSQD, ou SQD Modificado, que é basicamente uma simplificação que visa utilizar apenas o termo que efetivamente mede a distância entre as funções de densidade de probabilidade do critério anterior. Por fim, temos o critério que utiliza uma nova medida de correlação generalizada, chamada de correntropia (COR), a qual possui um desempenho interessante frente a sinais correlacionados [4].

Entretanto, os critérios baseados em ITL demandam a utilização da função densidade de probabilidade dos sinais transmitido e recebido. Para isso, tem-se utilizado um método não paramétrico baseado no uso de funções *kernel*, visto que assim não é necessário ter conhecimento prévio do sinal em questão. O método da janela de Parzen [5] se encaixa

justamente neste contexto. Tradicionalmente, o *kernel* mais utilizado na literatura é o *kernel* Gaussiano, uma função exponencial e, portanto, fácil de ser calculada e manipulada [6, 7, 8]. Porém, é possível obter melhores resultados ao estimar densidades de probabilidade com outros *kernels* [9, 10]. Segundo [10], o *kernel* Epanechnikov é ótimo nesse sentido.

Assim, uma primeira proposta deste trabalho foi estudar os algoritmos SQD e MSQD implementados *kernel* Epanechnikov e compará-lo com o algoritmo obtido através do uso do *kernel* Gaussiano. Tal estudo resultou na publicação do artigo (Anexo I): [11].

GOIS, L. e NEVES, A. "Equalização Adaptativa Cega de Sinais com Kernel Não-Gaussiano", XXXVII Simpósio Brasileiro de Telecomunicações e Processamento de Sinais, 2019, Petrópolis - RJ, 2019.

O algoritmo baseado no uso da COR, por sua vez, não explora todos os momentos estatísticos do sinal [4, 12], sendo os momentos considerados dependentes do *kernel* escolhido e do seu tamanho, no caso do *kernel* Gaussiano, por exemplo, são considerados todos os momentos de ordem par. Por outro lado, como o *kernel* Epanechnikov é dado por um polinômio de segunda ordem, a correntropia obtida com esta função leva em conta somente estatísticas de segunda ordem.

Deste modo, com o objetivo de estudar mais a fundo o critério resultante a partir do uso do *kernel* Epanechnikov, resolvemos compará-lo ao *Correlation Retrieval*, critério que foi proposto em [13] e que se baseia somente nas correlações dos sinais envolvidos.

Dessa forma, pretende-se simular o algoritmo baseado na COR com o *kernel* Epanechnikov e variar seus parâmetros, de forma a encontrar a melhor combinação de parâmetros para cada cenário de simulação considerado, e assim comparar com os melhores resultados obtidos quando utilizamos o *kernel* Gaussiano buscando determinar a eficiência de cada *kernel* para cada situação. Por fim, faremos também uma análise teórica do critério da correntropia utilizando o *kernel* Epanechnikov, comparando-o ao critério obtido usando-se o *kernel* Gaussiano e também ao *Correlation Retrieval*.

Este estudo resultou na publicação do seguinte artigo (Anexo II): [14].

GOIS, L.; FANTINATO, D. ; SUYAMA, R. ; NEVES, A. "Relações entre Critérios

baseados na correntropia e Estatísticas de Segunda Ordem para Equalização de Canais de Comunicação", 2020 Brazilian Technology Symposium, 2020, Campinas - SP, Proceedings of the 2020 Brazilian Technology Symposium, 2020.

## 2 FUNDAMENTAÇÃO TEÓRICA

### 2.1 Sistema em Estudo

A equalização é um método utilizado na correção de distorções sofridas por um sinal transmitido em um canal não ideal, buscando compensar as distorções ocasionadas. Este método pode ser implementado de forma adaptativa, acompanhando as mudanças do canal, sendo dividido em dois grupos: a equalização supervisionada e a não supervisionada, também chamada de equalização cega. Na Figura 1 a seguir, vemos a representação de um sistema de equalização:

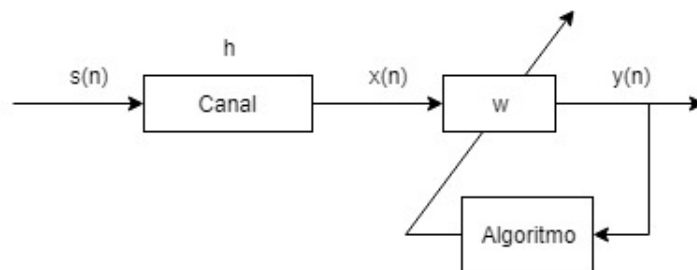


Figura 1: Diagrama de blocos da estrutura de um algoritmo de equalização

Neste sistema, temos o sinal transmitido,  $s(n)$ , através de um canal com resposta ao impulso,  $\mathbf{h} = [h_0, h_1, h_2, \dots, h_{L_h-1}]$ , onde  $L_h$  é o número de elementos em  $h$  e  $h_i$  são os coeficientes do canal. A saída do canal é descrita como:

$$x(n) = \sum_{i=0}^{L_h-1} h_i s(n-i) \quad (1)$$

Para recuperar a informação do sinal transmitido  $s(n)$ , consideramos um equalizador como um filtro FIR (*Finite Impulse Response* - FIR), com coeficientes dados por  $\mathbf{w} = [w_0, w_1, w_2, \dots, w_{M-1}]$ , onde  $M$  é o número de elementos de  $\mathbf{w}$ . O sinal de saída do equalizador é definido como:

$$y(n) = \sum_{i=0}^{k_w-1} w_i x(n-i) = \mathbf{w}^T \mathbf{x}(\mathbf{n}) \quad (2)$$

sendo  $w_i$  o elemento de ordem  $i$  do filtro. Assim, os efeitos ocasionados pelo canal  $\mathbf{h}$  devem

ser minimizados ou anulados pelo filtro  $\mathbf{w}$ , cujos coeficientes são ajustados iterativamente.

Na equalização supervisionada há conhecimento do canal ou a utilização de uma sequência de símbolos previamente conhecida pelo receptor, a sequência de treinamento. O algoritmo ajusta o vetor de coeficientes do equalizador através desta sequência, e assim minimizando o erro entre esta sequência e o sinal de saída do equalizador, levando aos coeficientes ótimos dados pela equação de Wiener-Hopf [2]:

$$\mathbf{w}_0 = \mathbf{R}_x^{-1} \mathbf{p} \quad (3)$$

onde  $\mathbf{R}_x$  é a matriz de autocorrelação do vetor  $\mathbf{x}(\mathbf{n}) = [x(n)x(n-1)\dots x(n-M+1)]$ , que apresenta uma janela de amostras do sinal de entrada do equalizador  $x(n)$  no instante  $n$ ,  $M$  o número de coeficientes do equalizador, portanto o mesmo tamanho de  $\mathbf{w}_0$ , e  $\mathbf{p}$  é o vetor de correlação cruzada entre  $x(n)$  e o sinal desejado  $s(n)$ . A equação (3) é conhecida como Solução de Wiener [2], que no campo da equalização é a solução ótima quando o erro quadrático médio é usado como critério. Apesar disso, a equação (3) é inviável na prática, pois seria necessário estimar  $\mathbf{R}_x$  e  $\mathbf{p}$  em tempo real, o que demandaria uma quantidade enorme de amostras para conseguir estimar estas grandezas.

Em certas situações, a equalização supervisionada possui algumas desvantagens. Quando existe a necessidade da utilização de uma sequência de treinamento, não há de fato a transferência de informação, visto que a sequência já é conhecida pelo receptor. Outra desvantagem é quando há a presença de vários receptores e apenas um transmissor, dessa forma seria necessário interromper o fluxo de dados para reiniciar a transmissão sempre que um novo receptor se conectasse, o que envolveria transmitir a sequência de treinamento novamente e utilizar mais recursos de transmissão [6].

Para situações em que a equalização supervisionada se torna inviável, ou até mesmo impossível, é necessário definir outra forma de adaptação dos coeficientes. Assim, adota-se a equalização cega, na qual o receptor não conhece o sinal transmitido e utiliza apenas as características estatísticas deste sinal no processo de equalização [2, 9, 10].

Os principais critérios de equalização cega seguem os teoremas de Benveniste-Goursat-Ruget e o teorema de Shalvi-Weinstein [15]. O teorema de Benveniste-Goursat-Ruget

afirma que a equalização é obtida ao igualarmos as densidades de probabilidade do sinal transmitido e o do sinal de saída do equalizador. O conhecimento da densidade de probabilidade implica no conhecimento de todos os momentos estatísticos do sinal. Já o teorema de Shalvi-Weinstein utiliza apenas a variância e a curtose. [15, 9]

Os sistemas de equalização cega possuem certa dificuldade frente a sinais correlacionados ou com ruído impulsivo, por isso vem sendo propostos novos critérios que utilizam teoria da informação no processo de ajuste dos coeficientes, buscando melhorar o desempenho dos algoritmos em cenários não favoráveis. Estes novos critérios em filtragem adaptativa são chamados de aprendizagem baseada na teoria de informação (ITL - *Information Theoretic Learning*) [3, 9].

Os algoritmos baseados em ITL a serem estudados neste trabalho, utilizam uma função *kernel* para estimação das densidades de probabilidade necessárias. Os trabalhos existentes na literatura, comumente, utilizam o *kernel* Gaussiano por ser fácil de ser calculado e manipulado. Visto que poucos estudos abordam a eficiência de *kernels* não-Gaussianos, este trabalho propõe a adoção do *kernel* Epanechnikov, uma vez que é o *kernel* de melhor eficiência na estimação de densidades de probabilidade [10], enquanto que o *kernel* Gaussiano é o mais utilizado em estatística e *machine learning* [10].

## 2.2 Estimador Baseado em Kernel

Uma das formas de se estimar densidades de probabilidade consistem em utilizar funções *kernel*. Por ser uma forma não-paramétrica de estimar a função densidade de probabilidade, não há a necessidade de conhecer previamente nenhuma outra característica da variável aleatória. Além disso, a estimação baseada em *kernel* permite que esta seja executada em tempo real, o que torna esta abordagem interessante no contexto de processamento de sinais e imagens [16].

O estimador baseado em *kernel* (KDE - *Kernel Density Estimator*) pondera cada observação de acordo com a distância de um valor central, dessa forma, um ponto mais próximo de outro terá mais influência do que outros pontos mais distantes [10]. A função densidade de probabilidade estimada pelo *kernel* é, normalmente, simétrica, assim



atribuindo o mesmo peso para todas as observações situadas à mesma distância do valor central [9].

A função de densidade de probabilidade estimada deve satisfazer a seguinte condição:

$$\int_{-\infty}^{+\infty} K(u)du = 1 \quad (4)$$

Desta forma, para estimar a função densidade de probabilidade do sinal, que passaremos a chamar de pdf, é utilizado o método da Janela de Parzen [5]:

$$\hat{f}_z(z) = \frac{1}{L\sigma} \sum_{i=1}^L K\left(\frac{z - z_i}{\sigma}\right) = \frac{1}{L} \sum_{i=1}^L K_\sigma(z - z_i) \quad (5)$$

onde  $L$  é o número de símbolos utilizados para estimar a densidade de probabilidade,  $z$  é um processo estocástico estacionário discreto,  $\sigma$  é a largura do *kernel* e, por fim,  $K(\cdot)$  é a função utilizada como *kernel*. A pdf estimada herda as propriedades de continuidade e diferenciabilidade da função  $K(\cdot)$  [9]. Assim, a função  $K(\cdot)$  determina o formato e o tamanho da curva, através de  $\sigma$ , que assume o papel de variância da pdf para o *kernel* Gaussiano.

Em [10], é proposto a eficiência do *kernel* baseada na proximidade da pdf estimada com a pdf real. Para isso, a eficiência é medida através da média da integral do erro quadrático AMISE (*Asymptotic Mean Integrated Square Error*) [10, 9], obtida ao expandir em série de Taylor e eliminar os termos de ordem superior da MISE (*Mean Integrated Square Error*).

$$MISE = E[I(x)] = E \int \left[ \hat{f}(x) - f(x) \right]^2 dx \quad (6)$$

Na Tabela 1 podemos ver os *kernels* seguidos de suas respectivas equações, variância elevada a ordem do *kernel*,  $\sigma_k^2$ , e a AMISE.

Kernel	Equação	$\sigma_k^2$	AMISE
Epanechnikov	$\frac{3}{4}(1-x^2)$	$\frac{1}{5}$	1
Uniforme	$U(-1, 1)$	$\frac{1}{3}$	1.0758
<i>Biweight</i>	$\frac{15}{16}(1-x^2)^2$	$\frac{1}{7}$	1.0061
<i>Triweight</i>	$\frac{35}{32}(1-x^2)^3$	$\frac{1}{9}$	1.0135
Triângulo	$(1- x )$	$\frac{1}{6}$	1.0143
Normal	$\frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}}$	1	1.0135

Tabela 1: Alguns dos *kernels* utilizados para estimação de funções densidade de probabilidade, considerando  $\sigma = 1$  [10]

Em [5, 6, 8] e nos trabalhos encontrados na literatura, a função comumente utilizada é a Gaussiana, que corresponde a uma distribuição normal padrão e possui uma boa eficiência. Assim, de acordo com [10], o *kernel* Gaussiano é dado por:

$$K_G(\mathbf{z} - \mathbf{z}_i) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(z-z_i)^2}{2\sigma^2}} \quad (7)$$

onde  $\sigma$  é o desvio padrão. O parâmetro  $\sigma$  também é chamado de tamanho ou largura do *kernel*, o seu valor impacta diretamente no desempenho do estimador, por isso deve ser analisado com cuidado. Além disso, se trata de uma função exponencial, o que torna mais fácil de ser trabalhado nos critérios baseados em ITL.

O *kernel* Epanechnikov é utilizado com frequência em estatística e machine learning [9], sendo o *kernel* mais usual além do Gaussiano. Ademais, este *kernel* é ótimo no sentido de minimizar a AMISE. O *kernel* é definido como um polinômio de segundo grau, ajustado para corresponder com uma função de densidade, além de ser analiticamente

simples, como podemos ver na seguinte equação:

$$K_E(z - z_i) = \frac{3}{4\sigma} \left( 1 - \left( \frac{z - z_i}{\sigma} \right)^2 \right), -\sigma < z - z_i < \sigma \quad (8)$$

onde  $\sigma$  é a largura do *kernel*. Neste trabalho focaremos especificamente na utilização destes dois *kernels*, buscando entender como critérios baseados em ITL se comportam e comparando o desempenho obtido pelos respectivos algoritmos utilizando os dois *kernels* definidos aqui. A seguir detalharemos os critérios implementados, os quais utilizaremos na comparação dos *kernels* Gaussiano e Epanechnikov.

### 3 CRITÉRIO DA DISTÂNCIA QUADRÁTICA (SQD)

Sendo um dos mais populares dentre os métodos baseados em ITL, o SQD (sigla em inglês de *Stochastic Quadratic Distance*) utiliza um critério baseado na distância quadrática entre as pdfs do sinal de saída do equalizador e do sinal transmitido. Dessa forma, a função custo é dada como [5, 17]:

$$J_{SQD}(\mathbf{w}) = \int_{-\infty}^{+\infty} (f_y(z) - f_s(z))^2 dz \quad (9)$$

onde  $f_y$  e  $f_s$  são, respectivamente, as pdfs do sinal de saída do equalizador e do sinal transmitido inicialmente,  $s(n)$ .

#### 3.1 Desenvolvimento do SQD com o kernel Gaussiano

Em (9), podemos notar que se trata de uma função custo quadrática, dessa forma, ao expandir a equação, se faz necessário calcular a convolução entre dois *kernels*. Neste caso, o *kernel* Gaussiano (7) é extremamente vantajoso por conta da seguinte propriedade:

$$\int_{-\infty}^{+\infty} K_G(y - A_1)K_G(y - A_2)dy = K_{G_{\sigma\sqrt{2}}}(A_1 - A_2) \quad (10)$$

onde  $\sigma\sqrt{2}$  passa a ser o tamanho do *kernel*. Logo, estimando as pdfs em (9), utilizando a janela de Parzen (5) e a propriedade (10), chegamos na seguinte função custo com o *kernel* Gaussiano [5, 17]:

$$\begin{aligned} J_{SQD-Gau}(\mathbf{w}) = & \frac{1}{L^2} \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} K_{G_{\sigma\sqrt{2}}}(|y(n-j)|^2 - |y(n-i)|^2) + \\ & + \frac{1}{N_S^2} \sum_{i=1}^{N_S} \sum_{j=1}^{N_S} K_{G_{\sigma\sqrt{2}}}(|d_j|^2 - |d_i|^2) - \\ & - \frac{2}{LN_S} \sum_{i=1}^{N_S} \sum_{j=0}^{L-1} K_{G_{\sigma\sqrt{2}}}(|y(n-j)|^2 - |d_i|^2) \quad (11) \end{aligned}$$

onde  $N_S$  é o número de símbolos da constelação do sinal transmitido, enquanto que  $d_i$  são os símbolos na posição  $i$  de um vetor  $\mathbf{d}$  que contém os símbolos da constelação do sinal transmitido,  $s(n)$ . À vista disso, o primeiro termo é o estimador de informação relacionado à entropia de Renyi de segunda ordem, o segundo termo não depende do vetor  $\mathbf{w}$ , por isso pode ser descartado no processo de otimização, e por fim o terceiro termo efetivamente calcula a distância entre as pdfs, encaminhando a saída do equalizador para a pdf desejada [6, 9].

Logo, chegamos à equação de atualização dos pesos, onde é utilizado o Método do Gradiente Descendente:

$$w(n+1) = w(n) - \mu_{\sigma\sqrt{2}} \nabla_w J_{SQD}(w) \quad (12)$$

Aplicando o *kernel* Gaussiano ao critério, temos:

$$\begin{aligned} \nabla_w J_{SQD-Gau}(w) = & \\ & \frac{1}{L^2} \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} (|y(n-j)|^2 - |y(n-i)|^2) (y(n-j) \mathbf{x}^*(n-j) - y(n-i) \mathbf{x}^*(n-i)) * \\ & * K_{G_{\sigma\sqrt{2}}}(|y(n-j)|^2 - |y(n-i)|^2) - \\ & - \frac{2}{LN_S} \sum_{i=1}^{N_S} \sum_{j=0}^{L-1} (|y(n-j)|^2 - |d_i|^2) y(n-j) \mathbf{x}^*(n-j) K_{G_{\sigma\sqrt{2}}}(|y(n-j)|^2 - |d_i|^2) \quad (13) \end{aligned}$$

Denominaremos este algoritmo como SQD-Gau, afim de facilitar o entendimento das simulações.

### 3.2 Desenvolvimento do SQD com o kernel Epanechnikov

A implementação do *kernel* Epanechnikov no SQD foi primeiramente estudada em [9], que foi aprofundado e levou a publicação do artigo [11]. Entretanto, antes de podermos aplicar o *kernel* à estimação do critério SQD, precisamos calcular o resultado da convolução entre os dois *kernels*, de forma equivalente ao que foi feito em (10) para o *kernel* Gaussiano.

Com (8) e considerando o caso real, temos:

$$\begin{aligned} & \int_{-\sigma}^{+\sigma} \frac{3}{4\sigma} \left( 1 - \left( \frac{y - y(n-i)}{\sigma} \right)^2 \right) \frac{3}{4\sigma} \left( 1 - \left( \frac{y - y(n-j)}{\sigma} \right)^2 \right) dy \\ &= \frac{9}{8\sigma^3} \left[ \frac{8\sigma^4}{15} - \frac{2\sigma^2}{3} (|y(n-j)|^2 - |y(n-i)|^2)^2 + (y(n-i)y(n-j))^4 \right] \end{aligned} \quad (14)$$

Assim, poderíamos reescrever (11) usando a expressão acima:

$$\begin{aligned} J_{SQD-Epa}(w) = & \frac{1}{L^2} \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} \frac{9}{8\sigma^3} \left[ \frac{8\sigma^4}{15} - \frac{2\sigma^2}{3} (|y(n-j)|^2 - |y(n-i)|^2)^2 + (y(n-i)y(n-j))^4 \right] + \\ & + \frac{1}{N_S^2} \sum_{i=1}^{N_S} \sum_{j=1}^{N_S} \frac{9}{8\sigma^3} \left[ \frac{8\sigma^4}{15} - \frac{2\sigma^2}{3} (|d_j|^2 - |d_i|^2)^2 + d_i^4 d_j^4 \right] - \\ & - \frac{2}{LN_S} \sum_{i=1}^{N_S} \sum_{j=0}^{L-1} \frac{9}{8\sigma^3} \left[ \frac{8\sigma^4}{15} - \frac{2\sigma^2}{3} (|y(n-j)|^2 - |d_i|^2)^2 + (y(n-j)d_i)^4 \right] \end{aligned} \quad (15)$$

Novamente, a atualização dos coeficientes do equalizador utiliza o método gradiente descendente, como em (12). Derivando (15), obtemos a seguinte expressão:

$$\begin{aligned} \nabla_w J_{SQD-Epa}(w) = & \frac{1}{L^2} \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} \frac{9}{8\sigma^3} \left[ -\frac{4\sigma^2}{3} (|y(n-j)|^2 - |y(n-i)|^2) (y(n-j)\mathbf{x}(n-j) - \mathbf{x}(n-i)y(n-i)) + \right. \\ & \left. + 4y^4(n-j)y^3(n-i)\mathbf{x}(n-i) + 4y^4(n-i)y^3(n-j)\mathbf{x}(n-j) \right] - \\ & - \frac{2}{LN_S} \sum_{i=1}^{N_S} \sum_{j=0}^{L-1} \frac{9}{8\sigma^3} \left[ -\frac{4\sigma^2}{3} (|y(n-j)|^2 - |d_i|^2) (y(n-j)\mathbf{x}(n-j)) + 4y^3(n-j)d_i^4\mathbf{x}(n-j) \right] \end{aligned} \quad (16)$$

Este algoritmo será denominado SQD-Epa.

### 3.3 MSQD

A versão modificada do SQD, a qual chamaremos de MSQD (*Modified Stochastic Quadratic Distance*), leva em consideração apenas o termo cruzado da função custo do SQD (11), visto que é o termo que mede efetivamente a distância entre as pdfs do sinal de saída do equalizador e do sinal transmitido [5, 8, 18], além de diminuir o custo computacional do algoritmo. Dessa forma, a função custo é definida como:

$$J_{MSQD-Gau}(w) = -\frac{2}{LN_S} \sum_{i=1}^{N_S} \sum_{j=0}^{L-1} K_G(|y(n-j)|^2 - |d_i|^2) \quad (17)$$

O gradiente de (17) é dado por:

$$\nabla_w J_{MSQD-Gau}(w) = -\frac{2}{LN_S} \sum_{i=1}^{N_S} \sum_{j=0}^{L-1} (|y(n-j)|^2 - |d_i|^2) y(n-j) \mathbf{x}^*(n-j) K_G(|y(n-j)|^2 - |d_i|^2) \quad (18)$$

Aplicando (18) na equação de atualização dos coeficientes, como foi feito anteriormente com o SQD em (12), temos o algoritmo que denominaremos MSQD-Gau.

Assim como ocorreu no MSQD-Gau (18), consideraremos apenas o último termo de (16). O MSQD-Epa será determinado pelo gradiente:

$$\nabla_w J_{MSQD-Epa}(w) = -\frac{2}{LN_S} \sum_{i=1}^{N_S} \sum_{j=0}^{L-1} \frac{9}{8\sigma^3} \left[ \frac{-4\sigma^2}{3} (|y(n-j)|^2 - |d_i|^2) (y(n-j) \mathbf{x}(n-i)) + 4y^3(n-j) d_i^4 \mathbf{x}(n-j) \right] \quad (19)$$

### 3.4 Resultados de Simulação

Nesta seção, será feita uma análise do desempenho do SQD e MSQD com o *kernel* Epanechnikov, a partir dos valores de seus parâmetros. Nas simulações usamos um vetor de

pesos  $\mathbf{w}$  de tamanho 21. As simulações com o *kernel* Epanechnikov serão denotadas por Ke, enquanto que Kg indica as simulações utilizando o *kernel* Gaussiano. Os vetores de peso foram iniciados com um center spike. Em todas as figuras apresentadas foi feita uma média de 20 simulações e utilizado o maior passo de convergência possível.

Iniciamos a análise dos algoritmos utilizando um sinal  $s(n)$  descorrelacionado, dado pela modulação BPSK (Binary Phase Shift Keying), que utiliza símbolos extraídos da constelação  $-1, +1$ . Em seguida, passamos para o uso de um sinal correlacionado, onde o mesmo sinal BPSK passa por uma pré-filtragem.

Como medida de desempenho, utilizamos a interferência intersimbólica residual (IIS) definida como:

$$IIS(n) = \frac{\sum_j |\vartheta_j(n)|^2 - \max_j (|\vartheta_j(n)|^2)}{\max_j (|\vartheta_j(n)|^2)} \quad (20)$$

onde  $\vartheta(n) = [\vartheta_0(n)\vartheta_1(n)\dots\vartheta_M(n)]$  é a resposta combinada entre o canal e o equalizador no instante de tempo  $n$ .

### 3.4.1 SQD

Foram modelados dois canais para simulações de desempenho: um canal de fase mista  $c_1(z) = 0.2 + z^{-1} + 0.5z^{-2}$  e um canal de fase mínima  $c_2(z) = 1 + 0.5z^{-1} + 0.1z^{-2}$ . As simulações consistem em utilizar cada um desses canais para analisar a variação do passo de adaptação, o tamanho do *kernel* e o tamanho da janela. Nas Figuras 2 e 3 veem-se os diagramas de pólos e zeros dos canais  $c_1$  e  $c_2$  respectivamente.



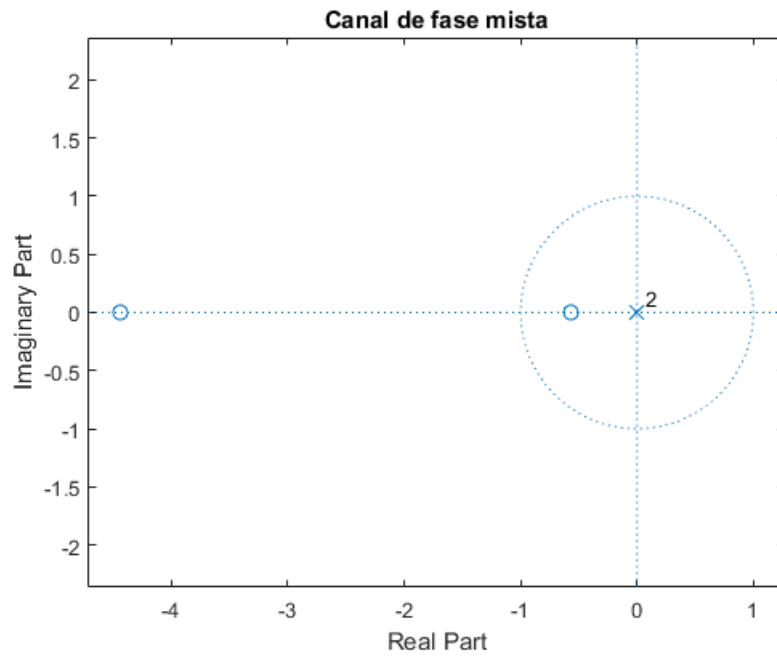


Figura 2: Diagrama de pólos de  $c_1(z)$

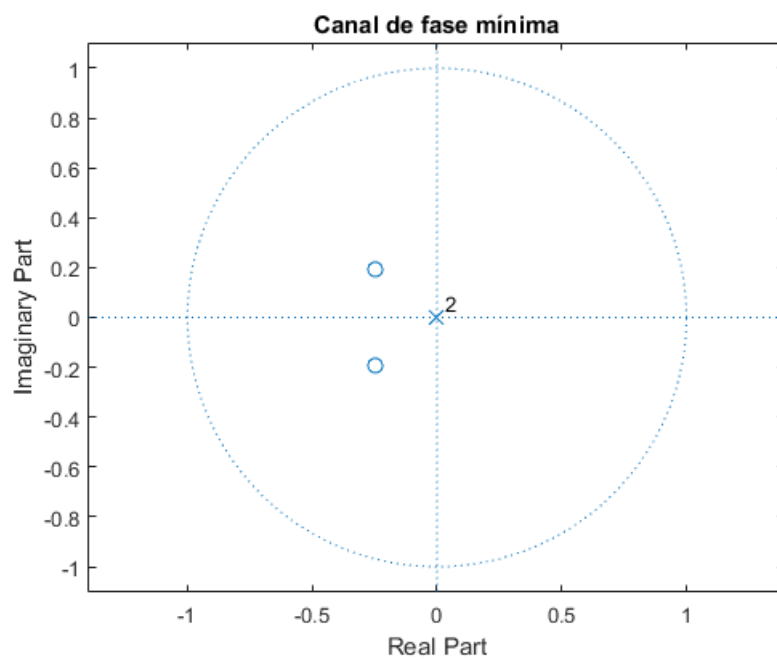


Figura 3: Diagrama de pólos de  $c_2(z)$

A fim de analisar o desempenho do algoritmo com relação aos seus parâmetros como: passo de adaptação ( $\mu$ ) e largura do *kernel* ( $\sigma$ ), começamos variando estes parâmetros e

observando o desempenho do algoritmo. Como o sinal  $s(n)$  é descorrelacionado (BPSK), mantivemos  $L = 2$  nestas primeiras simulações. Posteriormente, será utilizado um sinal correlacionado.

Todas as figuras mostram também o desempenho obtido pelo algoritmo com o *kernel* Gaussiano. Os melhores parâmetros, neste caso, foram obtidos do trabalho realizado em [6], onde este *kernel* foi explorado mais detalhadamente. A notação Kg foi usada para simulações com o *kernel* Gaussiano e Ke para o *kernel* Epanechnikov.

### Sinais descorrelacionados

Para um sinal BPSK, o vetor de pesos  $\mathbf{w}$  utilizado tem 21 coeficientes, iniciado com um *center spike*, e foi feita uma média de 20 simulações.

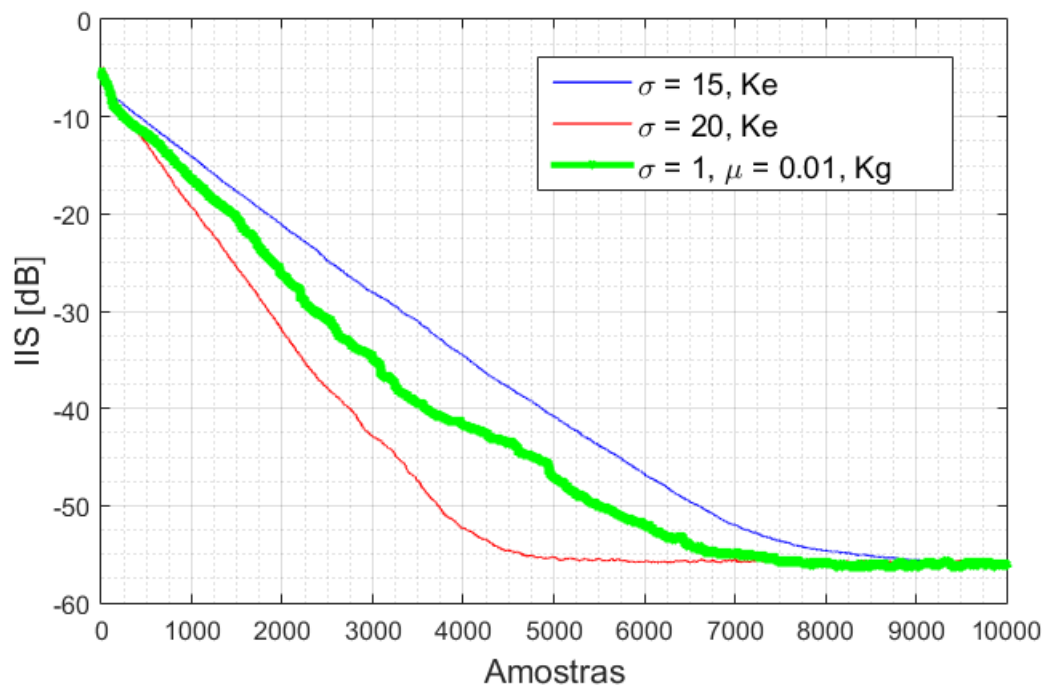


Figura 4: Simulação do SQR variando o  $\sigma$  para o canal  $c_1(z)$

Com  $c_1$ , a Figura 4 ilustra como o SQR, com o *kernel* Epanechnikov e Gaussiano, se comporta conforme variamos o tamanho do *kernel*, no caso  $\sigma = 15$  e  $20$  para o SQR-Epa

e  $\sigma = 1$  para o SQD-Gau, o melhor caso obtido. Mantivemos o passo fixo em  $\mu = 1 \times 10^{-5}$  no SQD-Epa e  $\mu = 0.01$  no SQD-Gau, para ambos utilizamos  $L = 2$ .

Para o SQD-Epa, o uso de valores bem maiores, chegando a  $\sigma = 20$ , leva o algoritmo a apresentar melhores desempenhos. Para valores abaixo desse, a velocidade de convergência do algoritmo tende a diminuir. Diferentemente do SQD-Gau, para o qual  $\sigma = 1$  leva aos melhores resultados, como visto no trabalho realizado em [6].

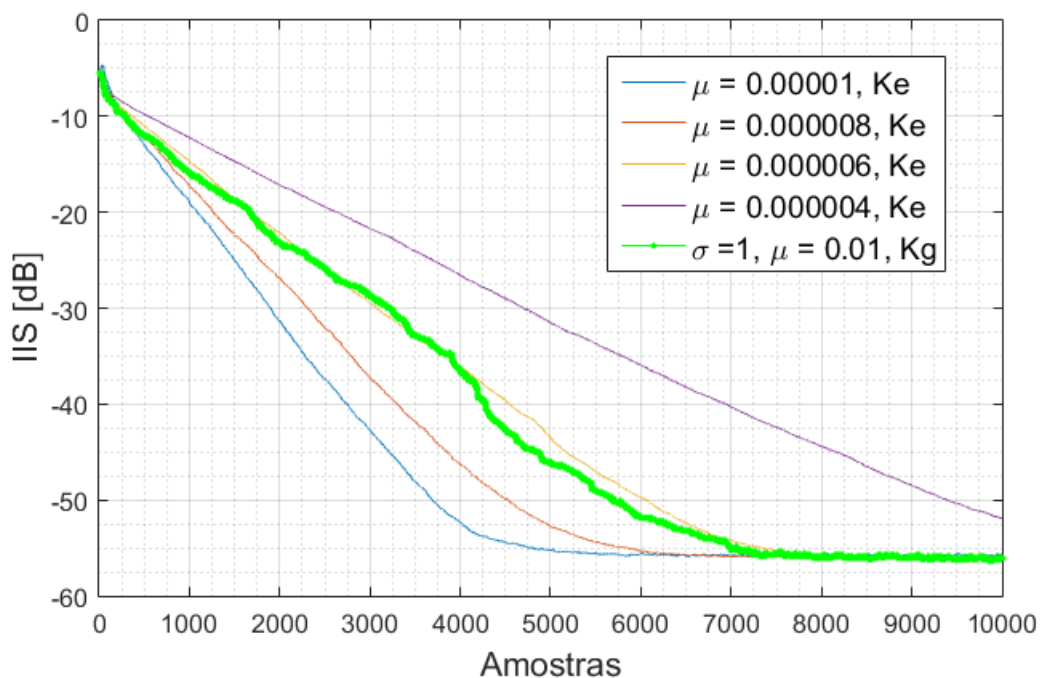


Figura 5: Simulação do SQD variando o passo  $\mu$  para o canal  $c_1(z)$

A Figura 5 mostra como a variação do passo afeta o desempenho do SQD para o canal  $c_1$  mantendo  $\sigma$  fixo igual a 20 para Ke, obtido na Figura 4, pois apresentou um melhor desempenho, o passo  $\mu$  foi aumentado gradualmente até o maior valor que o algoritmo converge, no caso  $\mu = 1 \times 10^{-5}$ . No SQD-Gau, foi utilizado  $\sigma = 1$  e  $\mu = 0.01$ . Para os dois algoritmos  $L = 2$ .

Vemos que, em geral, o passo necessário para que o SQD-Epa convirja é bem menor do que o passo usado para o SQD-Gau. Ainda assim, a Figura 5 já mostra como o SQD-Epa pode apresentar um desempenho melhor do que o algoritmo com o *kernel* Gaussiano,

necessitando menos amostras para convergir. O passo  $\mu = 1 \times 10^{-5}$  é o maior valor que pode ser usado para o SQD-Epa que ainda garante a convergência do algoritmo.

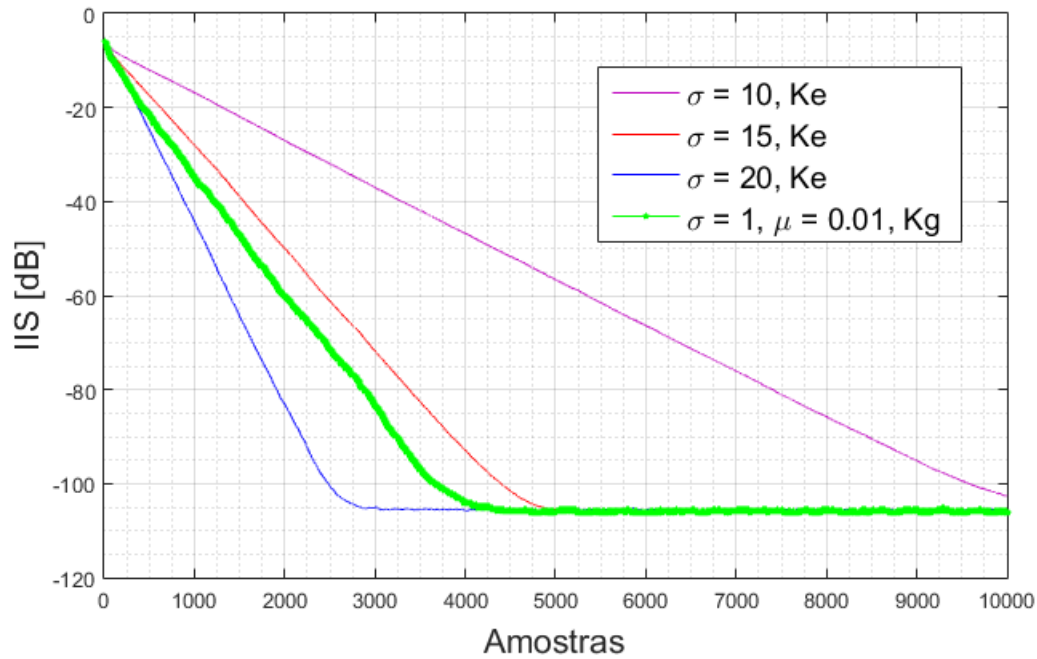


Figura 6: Simulação do SQD variando o  $\sigma$  para o canal  $c_2(z)$

Utilizando  $c_2$ , a Figura 6 apresenta a variação do parâmetro  $\sigma$  com o algoritmo SQD-Epa, comparando-o ao SQD-Gau, utilizando  $L = 2$  para ambos. Em SQD-Epa, o passo foi fixado em  $\mu = 1 \times 10^{-5}$ , verificando o resultado para  $\sigma = 10, 15$  e  $20$ . Para o SQD-Gau,  $\sigma = 1$  e  $\mu = 0.01$ .

O resultado encontrado foi similar ao da Figura 4, onde  $\sigma = 20$  apresentou o melhor desempenho em relação à velocidade de convergência, chegando a ser mais rápido que o melhor caso encontrado com o SQD-Gau.

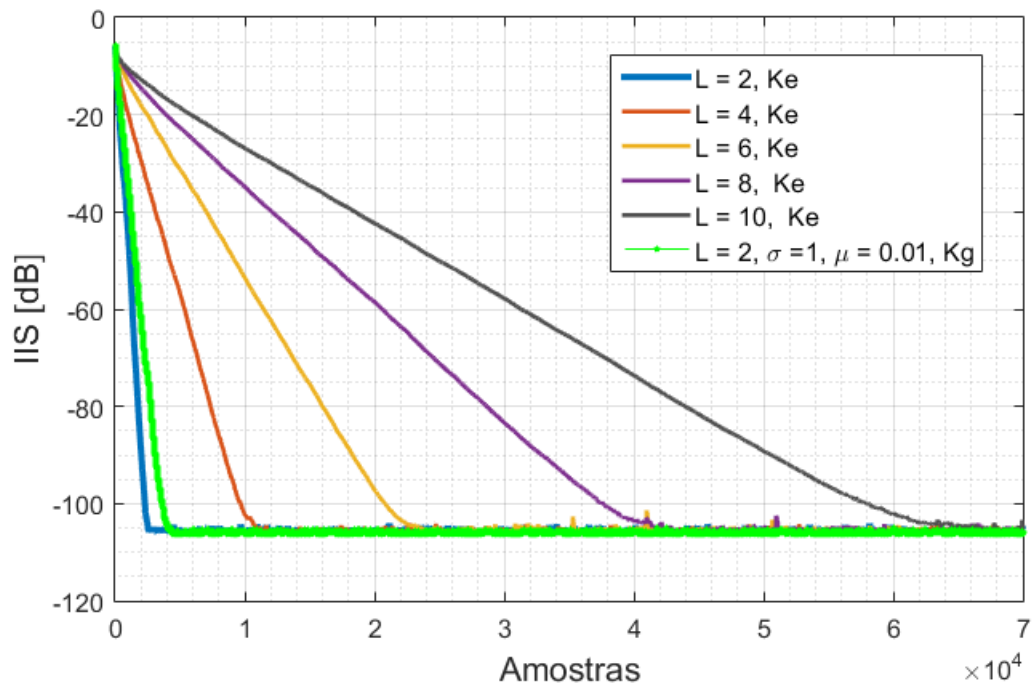


Figura 7: Simulação do SQD variando  $L$  para o canal  $c_2(z)$

A Figura 7 ilustra o desempenho do algoritmo SQD-Epa, conforme variamos o tamanho da janela temporal,  $L = 2, 4, 6, 8$  e  $10$ , para o caso do canal  $c_2$ , fixamos  $\mu = 1 \times 10^{-5}$  e  $\sigma = 20$ , o melhor caso alcançado ao variar o  $\sigma$  na Figura 6. Com o SQD-Gau, utilizou-se  $\sigma = 1$ ,  $\mu = 0.01$  e foi fixado  $L = 2$ .

Como o sinal utilizado aqui é decorrelacionado é de se esperar que o algoritmo consiga bons desempenhos com valores baixos para  $L$ , lembrando que este parâmetro está ligado à quantidade de atrasos usados na estimativa da pdf de  $y(n)$ . Conforme mostra a Figura 7, o melhor desempenho é obtido para  $L = 2$ , tanto para o caso com *kernel* Gaussiano como para o caso com *kernel* Epanechnikov. Por estarmos utilizando neste caso um canal de fase mínima, o desempenho das duas versões do algoritmo conseguem atingir desempenhos bastante semelhantes, com o SQD-Epa sendo ligeiramente mais veloz.

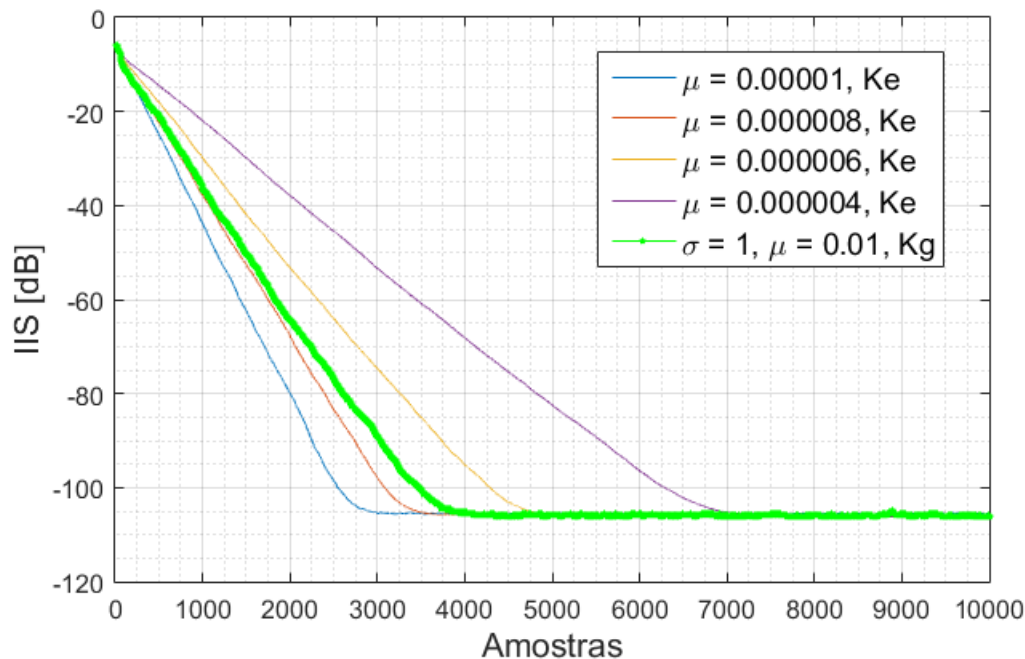


Figura 8: Simulação do SQD variando o passo  $\mu$  para o canal  $c_2(z)$

A Figura 8 mostra como a variação do passo afeta o desempenho do SQD para o canal  $c_2$  mantendo  $\sigma$  fixo igual a 20 para o SQD-Epa, obtido na Figura 4, pois apresentou um melhor desempenho, o passo  $\mu$  foi incrementado até o valor máximo, no caso  $\mu = 1 \times 10^{-5}$ . No SQD-Gau, foi utilizado  $\sigma = 1$  e  $\mu = 0.01$ . Para os dois algoritmos  $L = 2$ .

Assim como ocorreu na Figura 5, além de manter as mesmas condições, na Figura 8 vemos que o passo necessário para que o SQD-Epa convirja é bem menor do que o passo usado para o SQD-Gau no canal  $c_2$ . Novamente, a Figura 7 mostra como o SQD-Epa pode apresentar um desempenho melhor do que o algoritmo com o *kernel* Gaussiano, convergindo de forma mais rápida. O passo  $\mu = 1 \times 10^{-5}$  é o maior valor que pode ser usado para o SQD-Epa que ainda garante a convergência do algoritmo.

### Sinais correlacionados

Com o objetivo de controlar a correlação do sinal transmitido, filtramos o sinal BPSK por um pré-codificador,  $f(z)$ , a fim de gerar o sinal  $s(n)$  correlacionado que, na sequência,

será transmitido pelo canal. Assim, com o processo de equalização continuamos querendo recuperar o sinal  $s(n)$ .

Os filtros utilizados como pré-codificadores foram  $f_1(z) = 1 + 0.2z^{-1}$  e  $f_2(z) = 1 + 0.5z^{-1} + 0.2z^{-2}$ . É importante notar que o filtro  $f_1(z)$  introduz correlação entre as amostras nos instantes  $n$  e  $n-1$  de  $s(n)$ , enquanto que  $f_2(z)$  introduz entre as amostras nos instantes  $n$ ,  $n-1$  e  $n-2$ . Para todas as simulações a seguir, foi utilizado o canal  $c_2$  e utilizamos o melhor valor de passo de adaptação encontrado. Para o caso em que utilizamos o pré-codificador  $f_1(z)$ , consideramos  $L = 2$  e para os casos com  $f_2(z)$ , utilizamos  $L = 4$ . Vale ressaltar que o vetor de pesos  $\mathbf{w}$  utilizado tem 21 coeficientes, iniciado com um *center spike*, e foi feita uma média de 20 simulações e utilizado o maior passo de convergência possível.

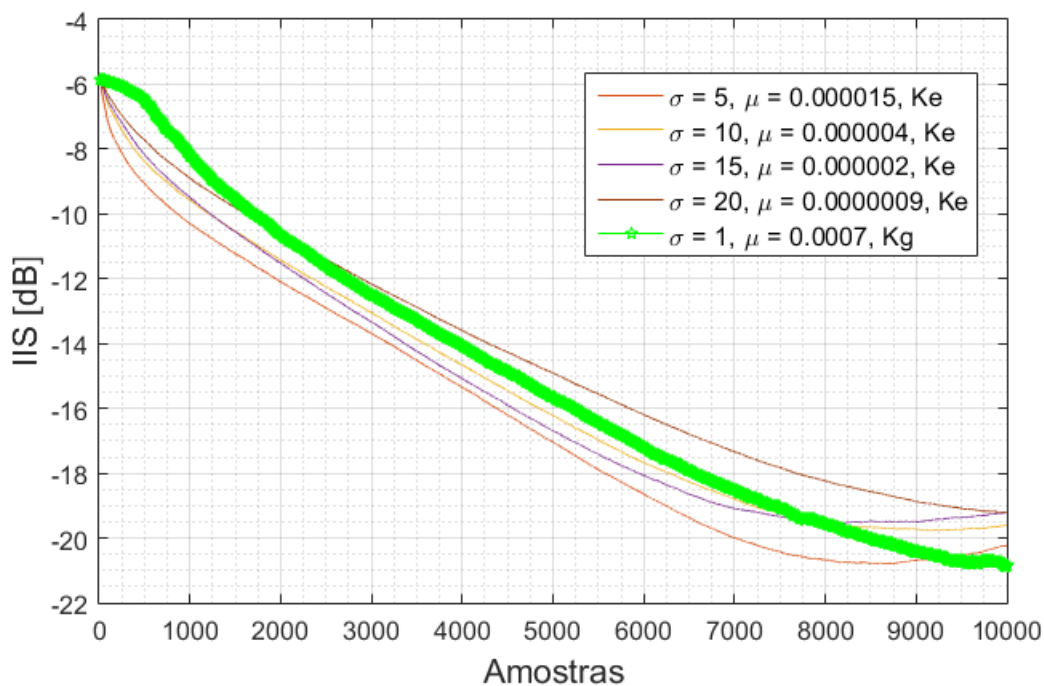


Figura 9: Simulação do SQD variando o  $\sigma$  para o canal  $c_2$  com pré-filtragem  $f_1(z) = 1 + 0.2z^{-1}$

Considerando primeiramente o pré-codificador  $f_1(z)$  e o SQD, a Figura 9 mostra o resultado obtido considerando vários valores possíveis para  $\sigma$  no caso do *kernel* Epanechnikov. Dessa forma, fixamos  $L = 2$  para ambos os *kernels*, encontrando o melhor passo

para cada  $\sigma$ , sendo  $\mu = 0.000015$  para  $\sigma = 5$ ,  $\mu = 0.000004$  para  $\sigma = 10$ ,  $\mu = 0.000002$  para  $\sigma = 15$  e  $\mu = 0.0000009$  para  $\sigma = 20$ . Já para o SQD-Gau, utilizamos  $\sigma = 1$  e  $\mu = 0.007$ , o melhor caso para esse *kernel*.

Podemos observar que as variações de  $\sigma$  não mudam muito o desempenho do algoritmo e praticamente temos o mesmo desempenho utilizando o SQD-Gau e o SQD-Epa. Aqui,  $\sigma = 5$  levou ao melhor desempenho, levando o algoritmo a convergir mais rápido do que o SQD-Gau, revertendo a tendência em obter melhores desempenhos com valores de  $\sigma$  mais altos, como aconteceu nas Figuras 4 e 6.

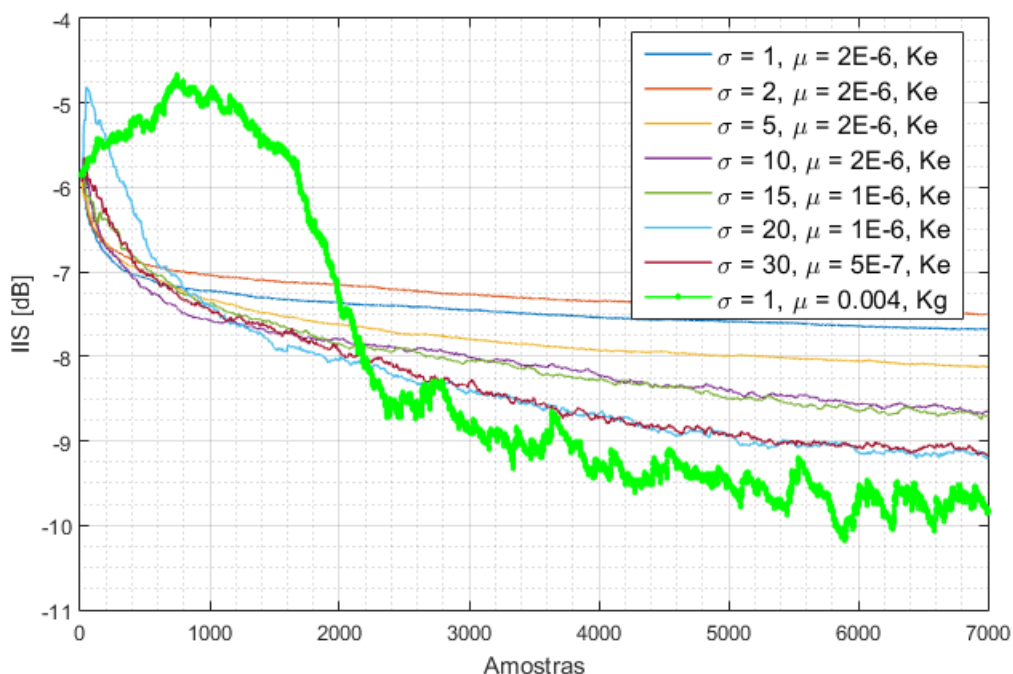


Figura 10: Simulação do SQD variando o  $\sigma$  para o canal  $c_2$  com pré-filtragem  $f_2(z) = 1 + 0.5z^{-1} + 0.2z^{-2}$

A Figura 10 ilustra o desempenho dos algoritmos com o uso do pré-codificador  $f_2(z)$ . Neste caso, fixamos  $L = 4$  em ambos os *kernels*, variando o  $\sigma$  no SQD-Epa e utilizando o maior  $\mu$  para cada caso. Assim, utilizamos  $\mu = 2 \times 10^{-6}$  para  $\sigma = 1, 2, 5$  e  $10$ ,  $\mu = 1 \times 10^{-6}$  para  $\sigma = 15$  e  $20$ , e  $\mu = 5 \times 10^{-7}$  para  $\sigma = 30$ . Enquanto que para o SQD-Gau, utilizamos  $\sigma = 1$  e  $\mu = 0.004$ .

Primeiramente, comparando com a Figura 9, podemos ver que existe uma perda de



desempenho neste caso. Na Figura 9, com  $f_1(z)$ , havia sido possível obter IIS residuais abaixo de -20dB. Neste caso, o menor valor atingido fica em torno de -10dB. Assim, temos uma situação mais difícil de ser equalizada. Os melhores desempenhos obtidos pelo SQD-Epa foram com  $\sigma = 20$  e  $\sigma = 30$ . Apesar do SQD-Gau apresentar uma oscilação maior, ele consegue atingir um patamar de IIS residual menor que o SQD-Epa.

### 3.4.2 MSQD

Será analisado o desempenho do algoritmo com relação ao passo de adaptação ( $\mu$ ) e largura do *kernel* ( $\sigma$ ), variando estes parâmetros e observar seu desempenho. Novamente utilizaremos o sinal  $s(n)$  (BPSK), primeiramente descorrelacionado e, por fim, correlacionado. Mantivemos  $L = 2$  nestas primeiras simulações. Também será utilizado o mesmo canal  $c_2(z) = 1 + 0.5z^{-1} + 0.1z^{-2}$  na análise de desempenho do SQD.

Todas as figuras incluem o melhor desempenho obtido pelo algoritmo com o *kernel* Gaussiano. Os melhores parâmetros foram obtidos do trabalho realizado em [6]. A notação Kg foi usada para simulações com o *kernel* Gaussiano e Ke para o *kernel* Epanechnikov.

### Sinais descorrelacionados

Para o MSQD , começamos novamente considerando a transmissão de um sinal descorrelacionado BPSK. A Figura 11 ilustra a variação do tamanho da janela. Para o MSQD-Epa, fixamos  $\sigma = 20$ , já considerando o melhor valor de passo para cada caso em  $c_2$ , no caso utilizando  $\mu = 8 \times 10^{-6}$  para  $L = 2$ ,  $\mu = 1 \times 10^{-5}$  para  $L = 4, 6, 8$  e  $10$ . Enquanto que mantivemos  $\sigma = 1$  e  $L = 2$  para o MSQD-Gau. As figuras também incluem o melhor desempenho obtido pelo SQD-Epa, com  $L = 2$ ,  $\sigma = 20$  e  $\mu = 1 \times 10^{-5}$ , para facilitar a comparação entre os algoritmos. As curvas geradas são resultado de uma média de 20 simulações.

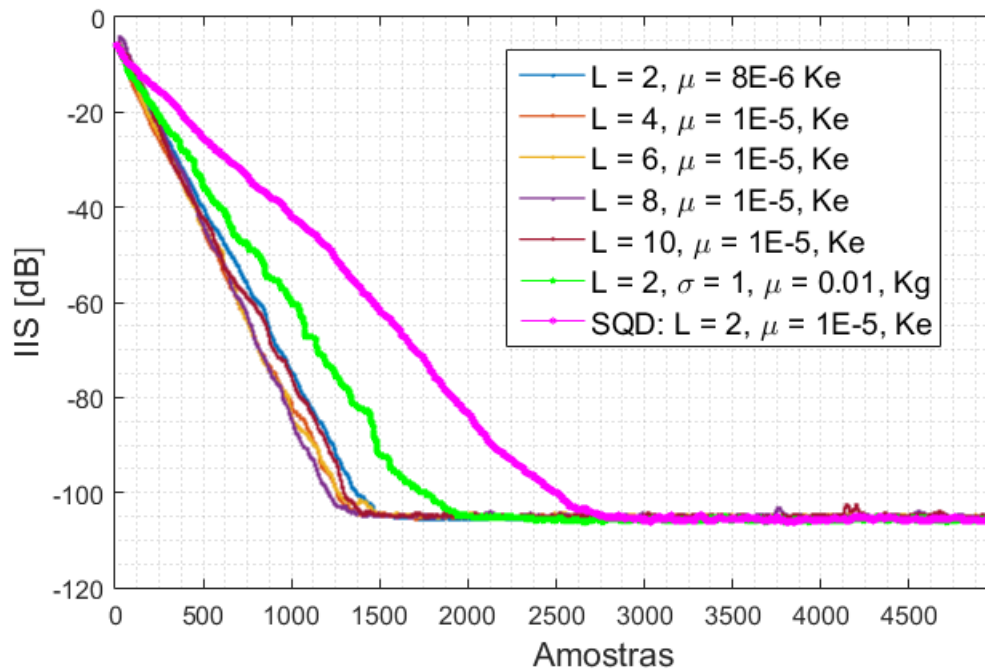


Figura 11: Simulação do MSQD variando o  $L$  para o canal  $c_2$

Na Figura 11 podemos ver que o MSQD-Epa possui um desempenho superior ao MSQD-Gau independente do valor de  $L$  utilizado, convergindo mais rapidamente para o mesmo patamar de IIS. Além disso, também é possível perceber que o melhor caso do SQD-Epa, quando  $L = 2$ , possui um desempenho inferior, em termos de velocidade de convergência, a todas as simulações do MSQD.

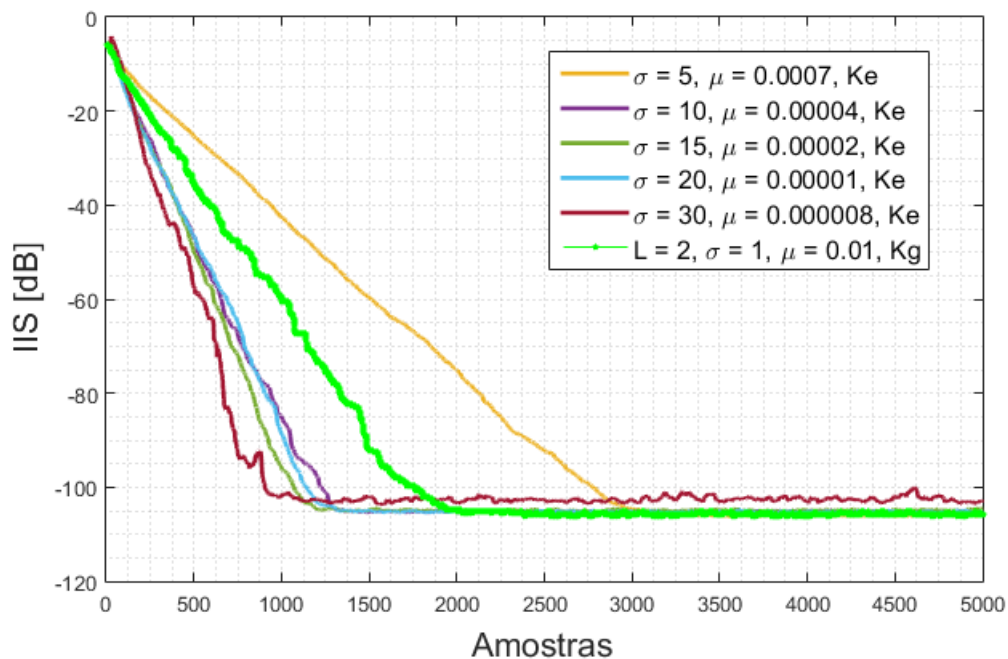


Figura 12: Simulação do MSQD variando o  $\sigma$  para o canal  $c_2$

Na Figura 12, com  $L = 2$  fixo e o canal  $c_2$ , foi variado o parâmetro  $\sigma$ , utilizando  $\mu = 7 \times 10^{-4}$  para  $\sigma = 5$ ,  $\mu = 4 \times 10^{-5}$  para  $\sigma = 10$ ,  $\mu = 2 \times 10^{-5}$  para  $\sigma = 15$ ,  $\mu = 1 \times 10^{-5}$  para  $\sigma = 20$  e  $\mu = 8 \times 10^{-6}$  para  $\sigma = 30$ , no caso, o melhor passo para cada curva do MSQD-Epa. Na Figura também é possível encontrar a curva correspondente ao MSQD-Gau, com  $\sigma = 1$ ,  $L = 2$  e  $\mu = 0.01$ .

Vemos que o desempenho MSQD-Epa foi praticamente o mesmo para  $\sigma$  maiores que 10. Além do mais, se mostrou superior em termos de velocidade de convergência em comparação com o melhor caso do MSQD-Gau. E, novamente, o sinal convergiu no mesmo patamar de IIS para todos os casos.

### Sinais correlacionados

Considerando agora o caso correlacionado, a Figura 13 ilustra o desempenho dos algoritmos utilizando o pré-codificador  $f_1(z)$  e o canal  $c_2$ . Fixando  $L = 2$  e variando o tamanho do *kernel*  $\sigma$  para o MSQD-Epa, com os passos  $\mu = 5 \times 10^{-6}$  para  $\sigma = 5$ ,

$\mu = 1.2 \times 10^{-6}$  para  $\sigma = 10$ ,  $\mu = 5 \times 10^{-7}$  para  $\sigma = 1.5$ ,  $\mu = 3 \times 10^{-7}$  para  $\sigma = 20$  e  $\mu = 15 \times 10^{-7}$  para  $\sigma = 30$ . Já a curva do MSQD-Gau, foi feita com  $\sigma = 1$ ,  $L = 2$  e  $\mu = 0.0007$ . Por fim, compararemos com a curva do SQD-Epa, gerada com  $\sigma = 5$ ,  $L = 2$  e  $\mu = 1.5 \times 10^{-5}$ .

Podemos notar um comportamento bem semelhante ao obtido pelo SQD e mostrado na Figura 9. A variação no valor de  $\sigma$  influi pouco no desempenho do algoritmo MSQD-Epa. Independentemente do valor de  $\sigma$ , o MSQD-Epa converge mais rapidamente do que o MSQD-Gau, sendo que o melhor desempenho é obtido com  $\sigma = 30$ . Assim como no caso de sinais descorrelacionados (Figura 9), aqui também podemos observar que o MSQD possui um desempenho superior ao SQD, convergindo mais rapidamente.

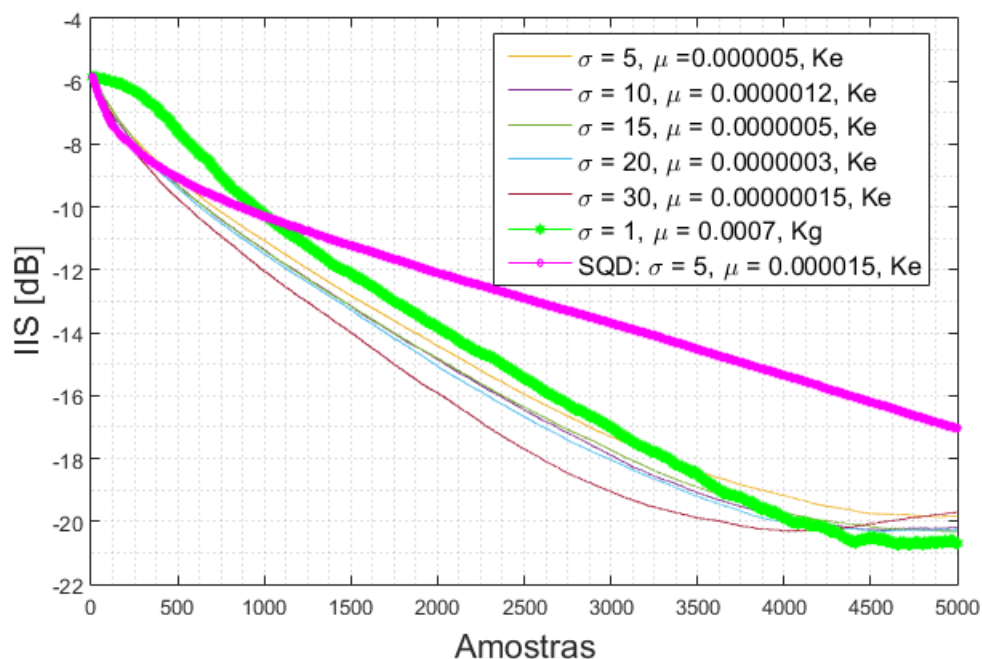


Figura 13: Simulação do MSQD variando o  $\sigma$  para o canal  $c_2$  com pré-filtragem  $f_1(z) = 1 + 0.2z^{-1}$

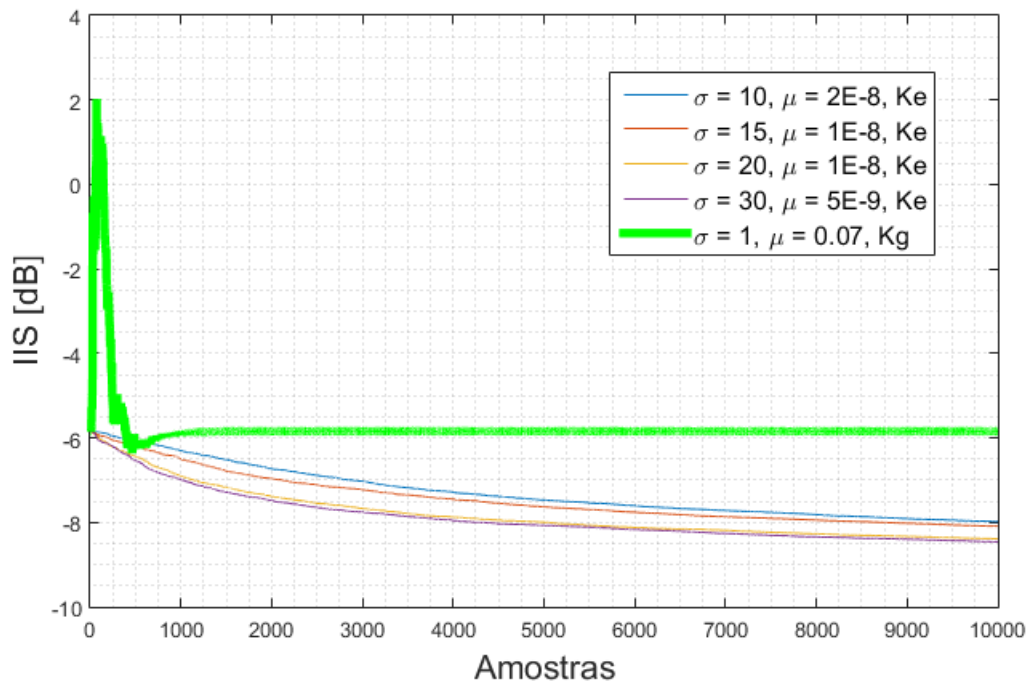


Figura 14: Simulação do MSQD variando o  $\sigma$  para o canal  $c_2$  com pré-filtragem  $f_2(z) = 1 + 0.5z^{-1} + 0.2z^{-2}$

Utilizando o pré-codificador  $f_2(z)$ ,  $L = 4$  e  $c_2$  para ambos os *kernels*, o resultado é mostrado na Figura 14 ao variarmos o tamanho do *kernel* no MSQD-Epa,  $\sigma$ . Neste caso, as curvas foram geradas com  $\mu = 2 \times 10^{-8}$  para  $\sigma = 10$ ,  $\mu = 1 \times 10^{-8}$  para  $\sigma = 15$  e  $20$ ,  $\mu = 5 \times 10^{-9}$  para  $\sigma = 30$ . Comparando com o desempenho do MSQD-Gau, com  $\sigma = 1$  e  $\mu = 0.07$ .

Neste caso, independente do *kernel* utilizado, o MSQD não conseguiu atingir um bom desempenho. Ainda assim, o MSQD-Epa conseguiu convergir para um patamar de IIS residual cerca de 2 dB abaixo do atingido pelo MSQD-Gau.

## 4 CRITÉRIOS BASEADOS NA CORRENTROPIA

A correntropia (COR) pode ser vista como uma nova medida de correlação generalizada, a qual está relacionada à estimativa da entropia quadrática de Renyi utilizando a janela de Parzen. Esta medida, proposta pela primeira vez em [5], vem despertando interesse de diversas áreas de especialização, principalmente por possuir como vantagem a caracterização estatística dupla: a capacidade de explorar diferentes momentos estatísticos dos sinais envolvidos e também sua estrutura temporal [14, 4].

Assim como na correlação, a correntropia considera diferentes atrasos entre as amostras. Pelo fato de utilizar uma função *kernel*, a medida mapeia os dados para um espaço de dimensão maior (*Reproducing Kernel Hilbert Space* – RKHS) de acordo com o *kernel* utilizado, sendo que as operações são feitas neste espaço e retornam para o espaço original dos dados [14, 4]. Dito isso, o formato da função *kernel* utilizada é um fator muito importante a ser considerado, visto que influencia o espaço de maior dimensão e os momentos estatísticos que serão levados em conta [14, 19]. Além disso, diferentes *kernels* levarão a critérios com características diferentes.

Em [5] os autores consideram a aplicação da medida em um critério que pode ser aplicado na equalização cega de sinais. Os resultados mostram como a correntropia é sensível às características temporais do sinal, o que torna o critério adequado para lidar com sinais correlacionados.

A definição da medida de correntropia é dada por:

$$V_A(m) = E [K(A(n) - A(n - m))] \quad (21)$$

onde  $E[.]$  denota a esperança matemática,  $K$  é a função *kernel*,  $A$  é um processo estocástico qualquer e  $m$  é o intervalo entre as amostras desta variável.

A equação (21) pode ser aproximada pelo cálculo através de uma média temporal utilizando  $N$  amostras da seguinte forma:

$$\hat{V}_A(m) = \frac{1}{N - m + 1} \sum_{i=1}^{N-m} K(A(n - i) - A(n - m - i)) \quad (22)$$

onde  $N - m + 1$  é a janela de amostras que usamos para estimar a correntropia.

#### 4.1 Desenvolvimento do método com o kernel Gaussiano

Na estimação da COR, escolhe-se comumente o *kernel* Gaussiano pelo fato de conseguir representar de forma satisfatória as características dos sinais envolvidos. A fim de utilizar a correntropia para equalização de canal, é possível propor um critério que visa minimizar a diferença entre as Correntropias do sinal original e do sinal na saída do equalizador, assim, a função custo é dada pela equação (23).

$$J_{COR}(w) = \sum_{m=1}^P (V_s[m] - V_y[m])^2 \quad (23)$$

onde  $P$  é o número de intervalos usados,  $V_s$  é a correntropia do sinal transmitido  $s(n)$  e  $V_y$  é a correntropia do sinal de saída do equalizador  $y(n)$ . Utilizando um algoritmo baseado no gradiente descendente para otimizar (23), a atualização dos pesos  $\mathbf{w}$  é dada por:

$$w(n+1) = w(n) + \mu \sum_{m=1}^P (V_s[m] - V_y[m]) \frac{\partial V_y[m]}{\partial w} \quad (24)$$

onde

$$\frac{\partial V_y[m]}{\partial w} = -\frac{1}{N-m} \sum_{i=n-N-m+1}^n K_G(y(i) - y(i-m)) (y(i) - y(i-m)) (\mathbf{x}(i) - \mathbf{x}(i-m)) \quad (25)$$

Este algoritmo será denominado COR-Gau, por utilizar o *kernel* Gaussiano.

#### 4.2 Desenvolvimento do método com o kernel Epanechnikov

A utilização do *kernel* Epanechnikov no cálculo na correntropia muda todo o critério que estamos considerando, visto que (23) será dada por uma expressão bastante diferente do que a utilizada na seção anterior, na qual utilizou-se o *kernel* Gaussiano. Dessa forma, as equações (22) e (25) são, respectivamente, reescritas com o *kernel* Epanechnikov como:

$$V_A(m) = \frac{1}{N-m+1} \sum_{n=m}^N \frac{3}{4\sigma} \left( 1 - \left( \frac{A(n) - A(n-m)}{\sigma} \right)^2 \right) \quad (26)$$

$$\frac{\partial V_y[m]}{\partial w} = -\frac{1}{N-m} \sum_{i=n-N-m+1}^n \frac{3}{2\sigma} \left( \frac{y(n-i) - y(n-m-i)}{\sigma^2} \right) (\mathbf{x}(n-i) - \mathbf{x}(n-m-i)) \quad (27)$$

Por conta do *kernel* Epanechnikov, este algoritmo será denominado COR-Epa.

Este critério pode ter soluções (mínimos locais e globais) diferentes das obtidas com o critério que utiliza o *kernel* Gaussiano. Por isso, pretendemos estudar este novo método, considerando tanto uma análise teórica para avaliar as soluções apresentadas por este novo critério, como também o desempenho obtido pelo algoritmo correspondente.

### 4.3 Correlation Retrieval (CR)

O critério *Correlation Retrieval* (CR), proposto em [13], faz o uso exclusivo de momentos estatísticos de segunda ordem, explorando a correlação dos sinais para vários atrasos entre amostras diferentes, sendo eficiente diante de sinais correlacionados. A função custo do CR compara as funções de autocorrelação do sinal transmitido e recebido:

$$J_{CR}(w) = \sum_{m=0}^P |r_y(m) - r_s(m)|^q \quad (28)$$

onde  $r_s(m) = E|s_n s_{n-m}^*|$  e  $r_y(m) = E|y_n y_{n-m}^*|$  são as funções de autocorrelação com atraso  $m$ , sendo  $P$  o máximo atraso arbitrário e  $q$  um valor positivo não nulo, equivalente ao da equação (23). Geralmente assume-se  $q = 2$ . Como o canal é desconhecido, é necessário estimar a correlação do sinal recebido utilizando uma média temporal entre as amostras:

$$\hat{r}_y(m) = \frac{1}{N} \sum_{i=0}^{N-1} y_{n-i} y_{n-m-i}^* \quad (29)$$



onde  $N$  indica o número de pares usados na estimação. Dessa forma, é possível definir a função de atualização dos coeficientes como:

$$w(n+1) = w(n) + \mu \sum_{m=0}^P (r_y[m] - r_s[m]) \frac{\partial r_y[m]}{\partial w} \quad (30)$$

onde  $\mu$  é o passo de adaptação e:

$$\frac{\partial r_y[m]}{\partial w} = \frac{1}{N-m} \sum_{i=n-N-m+1}^n y(n-i)x(n-m-i) + y(n-m-i)\mathbf{x}(n-i) \quad (31)$$

#### 4.4 Comparação entre Critérios

A Correntropia, implementada com o *kernel* Gaussiano, considera apenas os momentos pares. Assim, quando aumentamos o  $\sigma$ , os momentos de ordem superior decrescem e os momentos de segunda ordem se destacam [4, 19, 20]. Isto pode ser verificado ao expandir a equação (7) em série de Taylor:

$$V_A^G = \frac{1}{\sqrt{2\pi}\sigma} \sum_{k=0}^{\infty} \frac{(-1)^k}{2^k \sigma^{2k} k!} E[A^{2k}] \quad (32)$$

onde  $A$  é um processo estocástico qualquer. Assim, considerando apenas até a expansão de segunda ordem em torno de zero, temos:

$$V_A^G \approx \frac{1}{\sqrt{2\pi}\sigma} \left( 1 - \frac{1}{\sigma^2} (\sigma_A^2 - r_A(m)) \right) \quad (33)$$

onde  $\sigma_A^2$  é a variância do sinal e  $r_A$  a autocorrelação com atraso  $m$ . Ou seja, estamos comparando as variâncias e as autocorrelações dos sinais  $s(n)$  e  $y(n)$  em diversos atrasos entre as amostras.

Já o *kernel* Epanechnikov se trata de um polinômio de segunda ordem e, por conta disso, levará em consideração somente os momentos de segunda ordem. Utilizando (26), a correntropia com o *kernel* Epanechnikov é definida como:

$$\begin{aligned}
V_y^E(m) &= E \left[ \frac{3}{4\sigma} \left( 1 - \left( \frac{y(n) - y(n-m)}{\sigma} \right)^2 \right) \right] \\
&= \frac{3}{4\sigma} - \frac{3}{2\sigma^3} E [y^2(n)] + \frac{3}{2\sigma^3} E [A(n)y(n-m)] \\
&= \frac{3}{4\sigma} - \frac{3}{2\sigma^3} (\sigma_y^2 - r_y(m)) \quad (34)
\end{aligned}$$

É possível entender o termo constante em (34) e a variância do sinal como uma restrição de amplitude, assim chegando na seguinte aproximação:

$$V_y^E \approx r_y(m) \quad (35)$$

Considerando que esta aproximação também é válida para  $s(n)$ , a função custo do COR-Epa (23) se torna aproximadamente equivalente à função custo de CR (28). Desta forma podemos esperar que os dois critérios apresentem soluções semelhantes.

Já comparando a correntropia com o *kernel* Gaussiano e a correntropia com o *kernel* Epanechnikov, pode-se notar a semelhança entre as equações (33) e (34). Dito isso, é esperado que os dois algoritmos alcancem resultados parecidos a medida que aumentamos  $\sigma$  [9], visto que quanto maior a largura do *kernel*, menor é a contribuição dos momentos de ordem superior em (33). Entretanto, vale ressaltar que (33) representa uma aproximação até segunda ordem, ou seja, as superfícies de erro de COR-Gau e COR-Epa serão diferentes e os pontos de mínimos podem não coincidir.

Dentro deste contexto, faremos uma comparação das superfícies de erro de COR e CR, buscando analisar as soluções encontradas, os mínimos locais e globais, baseando-se no momento estatístico utilizado por ambos durante o processo de equalização.

Para isso, nas funções custo (23) e (28), foi necessário fixar o máximo atraso arbitrário  $P = 5$  para todos os critérios, visto que a utilização de valores mais altos ocasionaria na redução de mínimos locais. Vale ressaltar que todas as amostras do sinal foram consideradas na estimação das funções custo. Dessa forma, o único parâmetro variável é o tamanho do *kernel*  $\sigma$  no algoritmo COR. Nas simulações de superfície de erro, o canal

utilizado foi  $m(z) = 1 + 0.6z^{-1}$ , um canal de fase mínima, cujo diagrama de polos e zeros pode ser encontrado na Figura 15.

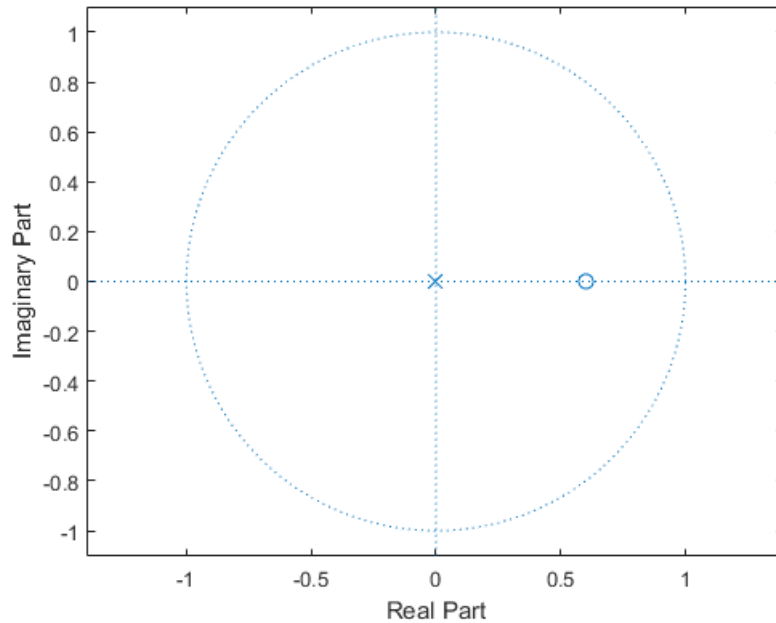


Figura 15: Diagrama de pólos do canal  $m(z)$

Nas figuras a seguir, há a sobreposição das superfícies de erro para os três casos: COR-Epa, COR-Gau e o CR. Os pontos de mínimos de cada caso são indicados pelo asterisco. As superfícies de erro são geradas ao utilizar um vetor de pesos  $\mathbf{w}$  de tamanho 2, e assim variar os dois coeficientes para obter o valor da função custo, sendo possível encontrar os mínimos globais e locais.

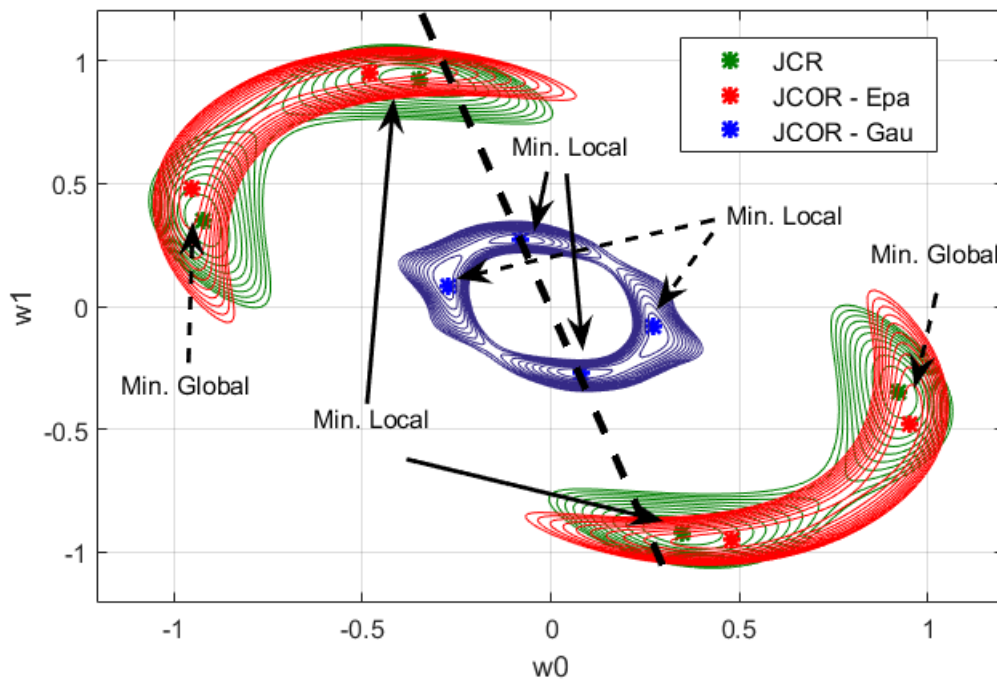


Figura 16: Simulação das superfícies de erro, fixando  $\sigma = 0.25$

Na Figura 16 vemos as superfícies de erro do COR e do CR para equalização de um sinal BPSK, com  $\sigma = 0.25$ . Percebe-se que a superfície de erro do COR-Gau possui uma amplitude diferente das demais, o que ocorre devido ao baixo valor de  $\sigma$ . Podemos notar no entanto que, traçando uma reta que passa pelas soluções globais do critério, esta é próxima das soluções apresentadas pelo critério CR. Assim, fazendo uma correção de ganho, estas soluções se mostram relativamente próximas.

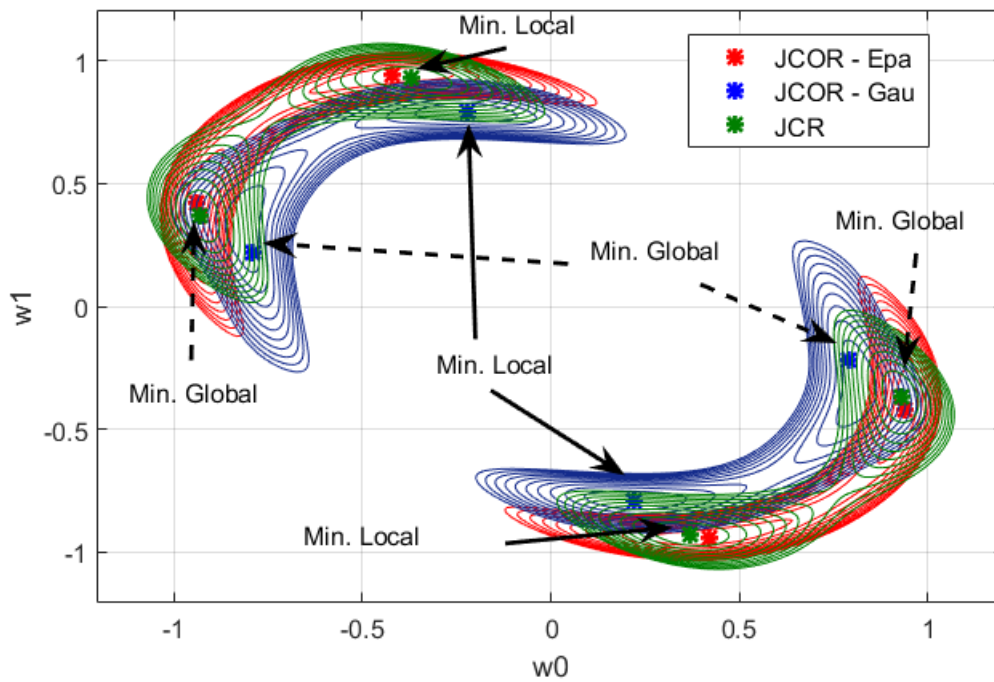


Figura 17: Simulação das superfícies de erro, fixando  $\sigma = 0.75$

A Figura 17 mostra as superfícies de erro do COR e do CR para equalização de um sinal BPSK, com  $\sigma = 0.75$ . Nota-se a proximidade dos pontos mínimos dos critérios COR-Epa e CR, que se encontram ainda mais próximos do que foi visto na Figura 16, enquanto que os mínimos do COR-Gau continuam mais afastados. Entretanto, observando as Figura 16 e 17, é possível perceber que a superfície de COR-Gau se aproximou consideravelmente das outras quando aumentamos o tamanho do *kernel*.

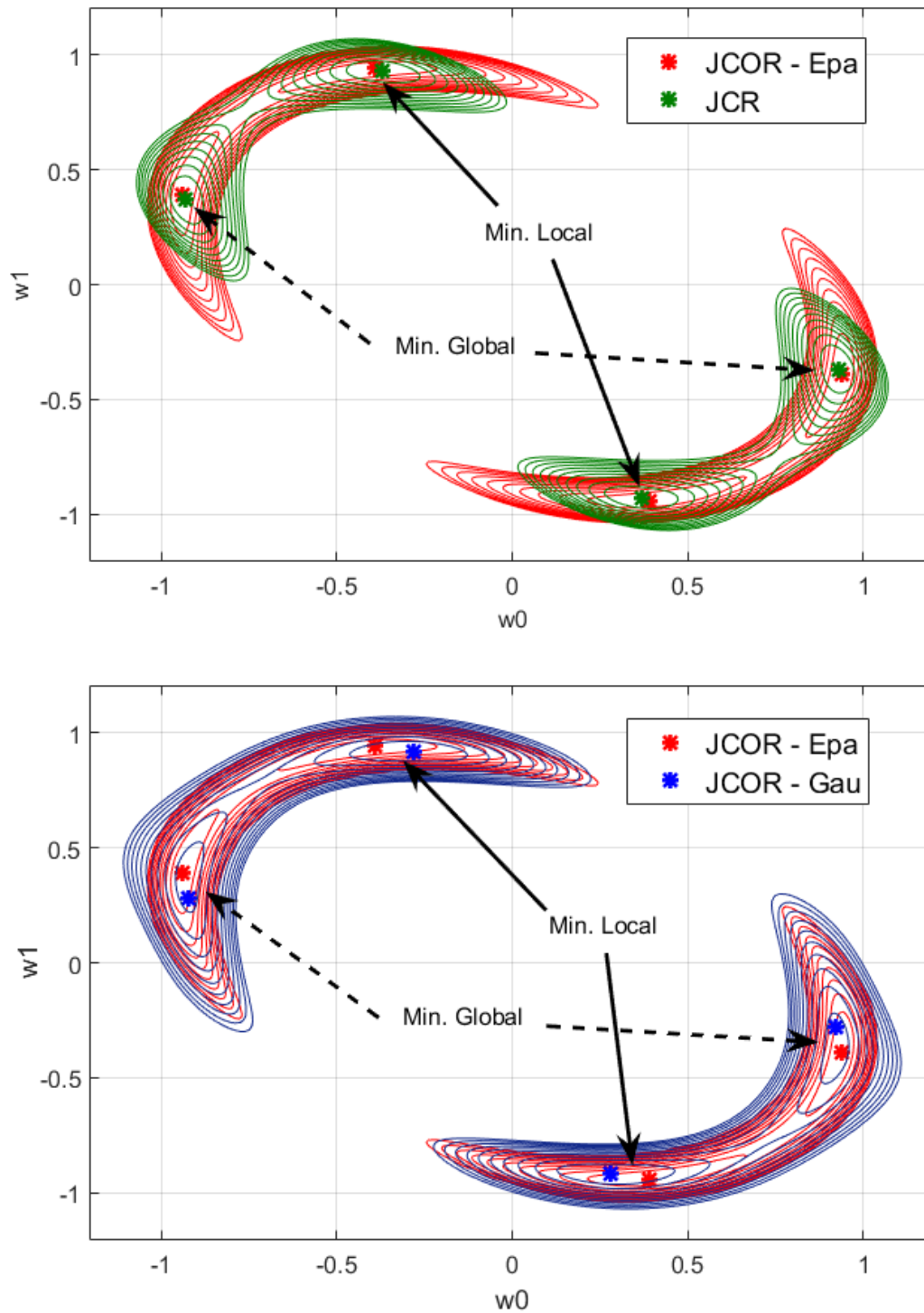


Figura 18: Simulação das superfícies de erro, fixando  $\sigma = 1$

Na Figura 18 encontramos as superfícies de erro do COR e do CR para equalização de um sinal BPSK, com  $\sigma = 1$ . Em contraste com a Figura 17, nota-se que a superfície de erro do COR-Gau está ainda mais próxima dos demais algoritmos, assim como os pontos de mínimo. Porém, percebe-se que os mínimos dos algoritmos COR-Epa e CR continuam mais próximos em comparação ao COR-Gau. Apesar disso, o COR-Gau assumiu um formato mais semelhante ao do COR-Epa, indicando que o aumento do tamanho *kernel* está aproximando cada vez mais as duas superfícies de erro e, conseqüentemente, a forma como a informação está sendo explorada [12].

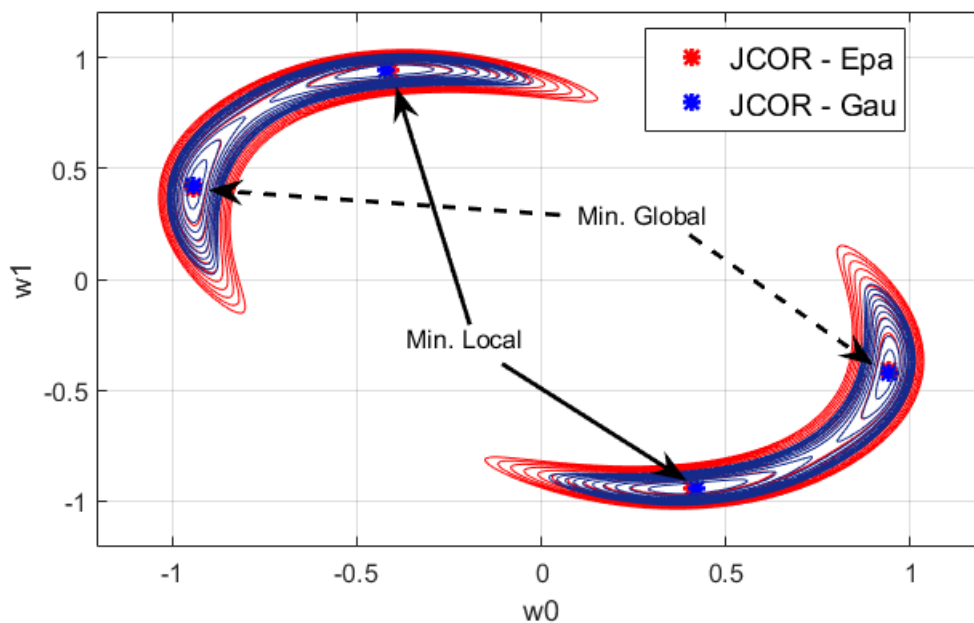


Figura 19: Simulação das superfícies de erro, fixando  $\sigma = 5$

Na Figura 19 encontramos as superfícies de erro do COR e do CR para equalização de um sinal BPSK, com  $\sigma = 5$ . Aumentando ainda mais o tamanho do *kernel*, nota-se que os pontos de mínimos dos dois algoritmos estão bem próximos, como já era esperado do COR-Gau ao aumentarmos o valor de  $\sigma$ . Outro ponto importante a se notar, é como os mínimos dos algoritmos COR-Epa e COR-Gau, que já eram próximos para  $\sigma$  menores nas Figuras 16, 17 e 18, passam a coincidir.

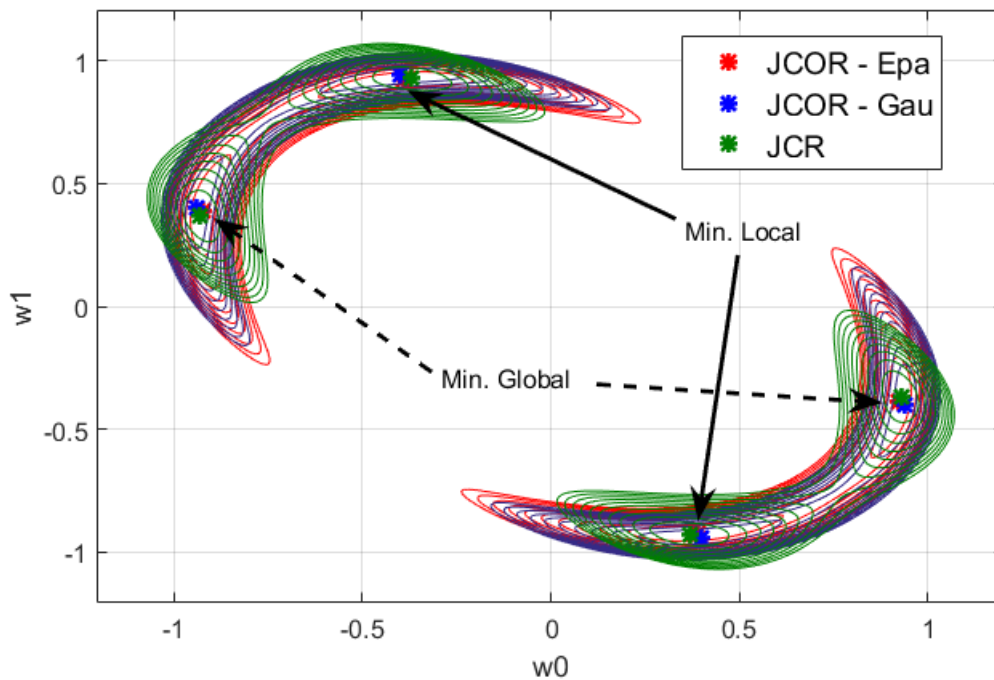


Figura 20: Simulação das superfícies de erro, fixando  $\sigma = 20$

Na Figura 20 encontramos as superfícies de erro do COR e do CR para equalização de um sinal BPSK, com  $\sigma = 20$ . Com o tamanho do *kernel* bem maior que as demais simulações realizadas nas Figuras 16, 17, 18 e 19, percebe-se que os mínimos dos três algoritmos passam a coincidir. Isso significa que podemos esperar resultados semelhantes para os três algoritmos.

#### 4.5 Análise de Desempenho dos algoritmos

Nesta seção, será feita uma análise do desempenho do algoritmo baseado na COR com o *kernel* Epanechnikov, comparando-o com o mesmo algoritmo utilizando o *kernel* Gaussiano e o critério CR. Nestas simulações utilizaremos um vetor de pesos  $\mathbf{w}$  de tamanho 3. Vale ressaltar que os vetores de peso foram iniciados com um center spike. Em todas as figuras apresentadas foi feita uma média de 50 simulações e utilizado o maior passo de convergência encontrado.



## Sinais descorrelacionados

Inicialmente, analisaremos as curvas de convergência quando utilizado o *kernel* Epanechnikov, em todos os casos há a comparação com o melhor caso para o *kernel* Gaussiano. Para as simulações de desempenho, utilizaremos um canal de fase mista  $h(z) = 0.1856 - 0.9650z^{-1} + 0.1856z^{-2}$ , ilustrado pela Figura 21, para introduzir distorções no sinal. O sinal utilizado durante a análise terá modulação BPSK (Binary Phase Shift Key), que consiste de símbolos extraídos da constelação  $-1, +1$ .

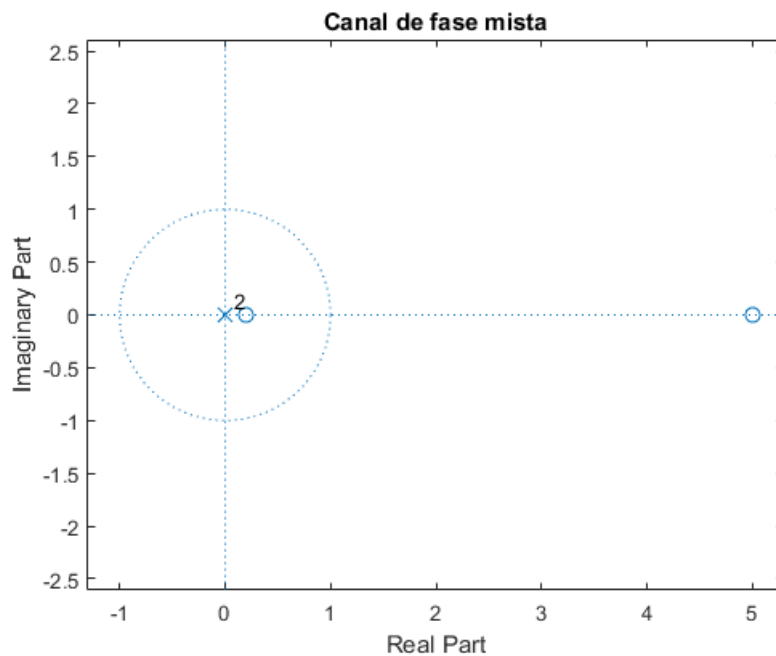


Figura 21: Diagrama de pólos do canal  $h(z)$

Nas simulações usamos um vetor de pesos  $\mathbf{w}$  de tamanho 3. As simulações com o *kernel* Epanechnikov serão denotadas por Ke, enquanto que Kg indica as simulações utilizando o *kernel* Gaussiano. Vale ressaltar que os vetores de peso foram iniciados com um center spike. Em todas as figuras apresentadas foi feita uma média de 50 simulações e utilizado o maior passo de convergência possível.

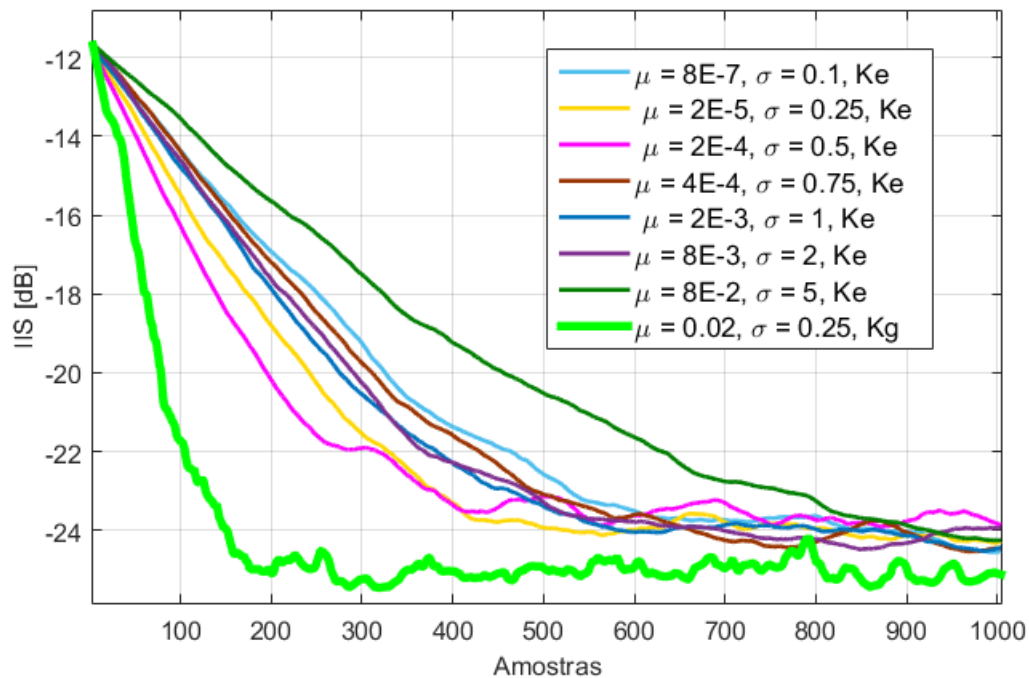


Figura 22: Simulação do COR variando o  $\sigma$  para o canal  $h(z)$

A Figura 22 mostra o desempenho do COR-Epa em comparação com o melhor caso utilizando o COR-Gau, variando entre diferentes tamanhos de *kernel* ( $\sigma$ ) na equalização de um sinal BPSK e obtendo a curva com o maior passo de adaptação ( $\mu$ ) possível, sendo estes  $\mu = 8 \times 10^{-7}$  para  $\sigma = 0.1$ ,  $\mu = 2 \times 10^{-5}$  para  $\sigma = 0.25$ ,  $\mu = 2 \times 10^{-4}$  para  $\sigma = 0.5$ ,  $\mu = 4 \times 10^{-4}$  para  $\sigma = 0.75$ ,  $\mu = 2 \times 10^{-3}$  para  $\sigma = 1$ ,  $\mu = 8 \times 10^{-3}$  para  $\sigma = 2$  e  $\mu = 8 \times 10^{-2}$  para  $\sigma = 5$ . Mantivemos  $P = 4$  e  $N = 10$  para as simulações utilizando o *kernel* Gaussiano, de onde foi obtido  $\sigma = 0.25$  como o melhor resultado. Já para o *kernel* Epanechnikov, foram utilizados  $\mu = 0.02$ ,  $N = 100$  e  $P = 50$ , estes últimos valores são intermediários escolhidos para uma análise inicial.

Observando os resultados da Figura 22, notamos que o valor da IIS residual que conseguimos atingir não variou muito, entretanto, a velocidade de convergência foi a mais afetada ao variar o tamanho do *kernel*. Nota-se que o COR-Epa converge mais rapidamente para  $\sigma$  pequenos, sendo o melhor resultado entre 0.25 e 0.5. Entretanto, o algoritmo utilizando o *kernel* Gaussiano mostrou atingir um patamar de IIS mais baixo

que todos os casos de Ke.

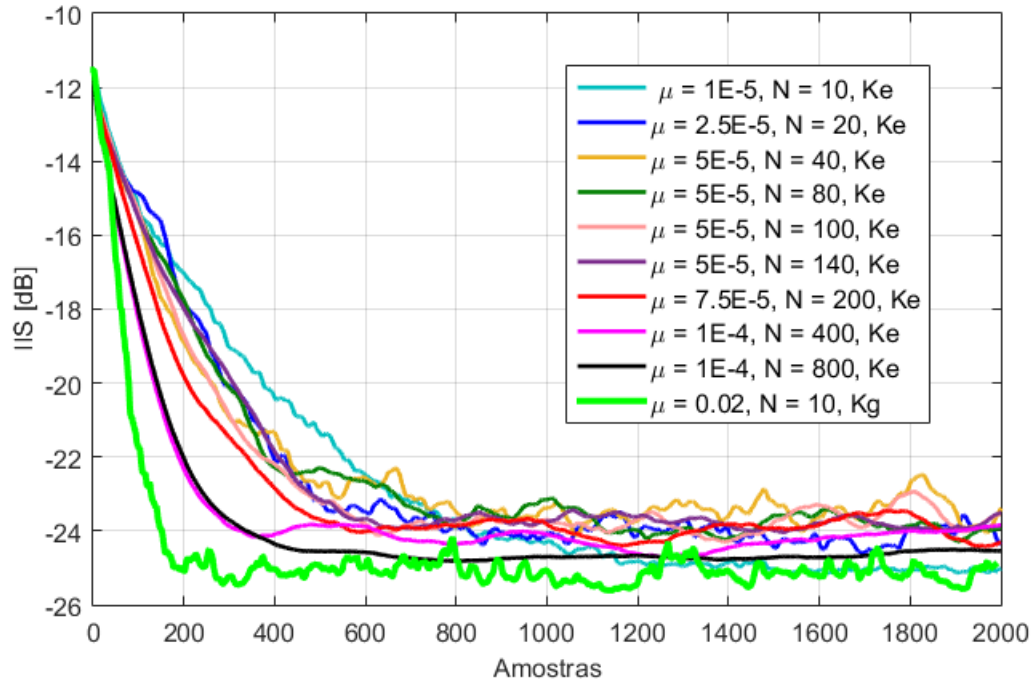


Figura 23: Simulação do COR variando o  $N$  para o canal  $h(z)$

Na Figura 23 vemos o desempenho do COR-Epa utilizando o *kernel* Epanechnikov em comparação com o melhor caso utilizando o COR-Gau ao variarmos o tamanho da janela de amostras utilizadas para estimação ( $N$ ). Fixamos o  $P = 4$  e  $\sigma = 0.25$  nas simulações utilizando o COR-Gau, pois foi o melhor caso ao variarmos o tamanho do *kernel*, assim encontramos que  $N = 10$  apresenta a maior velocidade de convergência com o passo  $\mu = 0.02$ . Para o *kernel* Epanechnikov, foi usado  $P = 4$  e  $\sigma = 0.25$ , já que este último é o melhor resultado encontrado quando variamos o *kernel* na Figura 23. Os passos utilizados para cada curva são,  $\mu = 1 \times 10^{-5}$  para  $N = 10$ ,  $\mu = 2 \times 10^{-5}$  para  $N = 20$ ,  $\mu = 5 \times 10^{-5}$  para  $N = 40, 80, 100$  e  $140$ ,  $\mu = 7.5 \times 10^{-5}$  para  $N = 200$ ,  $\mu = 1 \times 10^{-4}$  para  $N = 400$  e  $800$ .

É interessante observar que, tal como ocorreu na Figura 22 onde variamos o  $\sigma$ , a IIS residual não variou consideravelmente com os valores de  $N$  e a variação deste parâmetro influenciou apenas a velocidade de convergência. Apesar de o algoritmo convergir mais

rapidamente para  $N$  maiores no caso do *kernel* Epanechnikov, esses valores exigem um custo computacional muito maior, demandando mais tempo de processamento, dessa forma acaba por não ser vantajoso utilizar uma janela de amostras muito grande para estimação. Enfim, pode-se perceber que o algoritmo com o *kernel* Gaussiano se mostrou mais rápido do que todos os casos utilizando o *kernel* Epanechnikov.

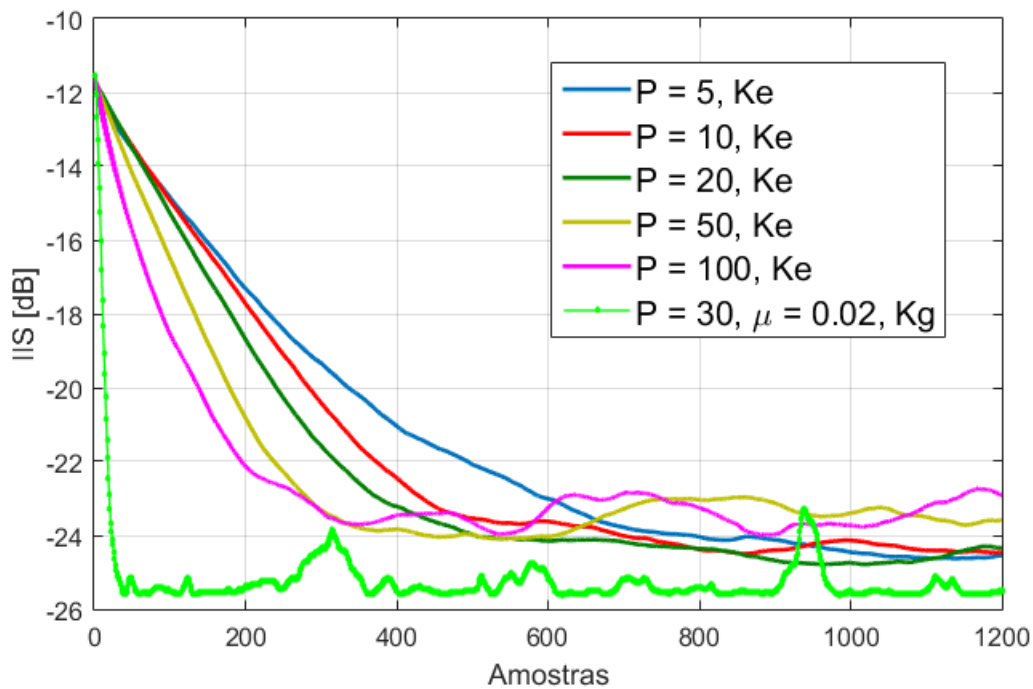


Figura 24: Simulação do COR variando o  $P$  para o canal  $h(z)$

A Figura 24 mostra o desempenho do COR-Epa em comparação com o melhor caso utilizando do COR-Gau, variando o número de intervalos usados na função custo ( $P$ ) entre 5 e 100, todos as curvas com o passo  $\mu = 1 \times 10^{-5}$ . No caso do *kernel* Epanechnikov, optou-se usar  $\sigma = 0.25$  por ser o melhor resultado apresentado na Figura 22 e  $N = 200$ , apesar de não ser o melhor caso, como podemos notar na Figura 23, os valores de  $P$  tornam-se restritos ao valor de  $N$ , sendo que  $P$  deve ser menor do que  $N$ , por isso foi escolhido um valor relativamente alto, mas que ao mesmo tempo não tornasse o processamento demorado. Para o *kernel* Gaussiano, foi fixado  $N = 40$  e  $\sigma = 0.25$ , onde se encontrou o melhor resultado quando  $P = 30$  com  $\mu = 0.02$ .

Examinando as curvas, notamos que a IIS residual não teve uma variação significativa, mas pode-se notar como a velocidade de convergência aumenta notavelmente à medida que o valor de  $P$  é incrementado para o *kernel* Epanechnikov, convergindo de forma mais rápida para valores próximos ao limite máximo. Entretanto, o *kernel* Gaussiano apresentou uma velocidade de convergência maior do que todos os casos simulados usando o *kernel* Epanechnikov, além disso, é possível notar que o algoritmo converge mais rápido quando é utilizado um valor de  $P$  próximo de  $N$ , para ambos os casos.

### Sinais correlacionados

Nesta seção, analisaremos o desempenho do COR-Epa, variando os seus parâmetros e comparando com o COR-Gau e o critério CR. Utilizaremos novamente o sinal  $s(n)$  dado por um sinal BPSK pré-filtrado por  $f(z) = 1 + 0.5z^{-1}$ . O canal é dado por  $g(z) = 0.2 + 1z^{-1} + 0.4z^{-2} + 0.1z^{-3}$ , ilustrado pela Figura 25. O equalizador será um vetor  $\mathbf{w}$  de tamanho 21, iniciado como um vetor *center spike*. As figuras mostram uma média de 30 simulações.

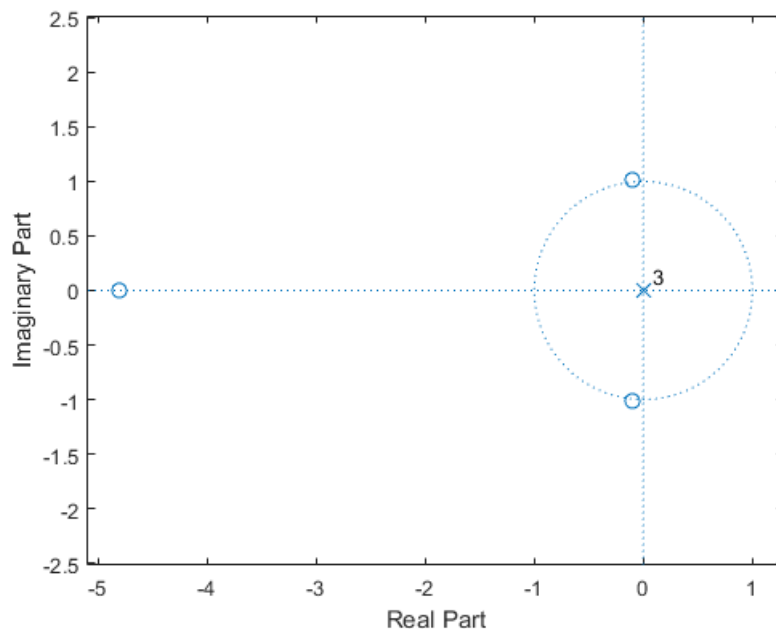


Figura 25: Diagrama de pólos do canal  $g(z)$

Na Figura 26, temos o desempenho do COR-Epa e do COR-Gau com o critério *Correlation Retrieval* (CR), ao fixarmos  $N = 50$  para todos e  $\sigma = 1$  para o COR, variamos  $P$  para os três algoritmos. Para  $P = 5$  e 10, utilizamos  $\mu_{CE} = 4 \times 10^{-6}$  (COR-Epa),  $\mu_{CR} = 1 \times 10^{-5}$  (CR). Já para  $P = 20$ ,  $\mu_{CE} = 1 \times 10^{-6}$ ,  $\mu_{CR} = 1 \times 10^{-5}$  e  $\mu_{CG} = 1 \times 10^{-4}$  (COR-Gau). Vale ressaltar que os passos foram escolhidos de forma que os algoritmos apresentassem a mesma velocidade de convergência.

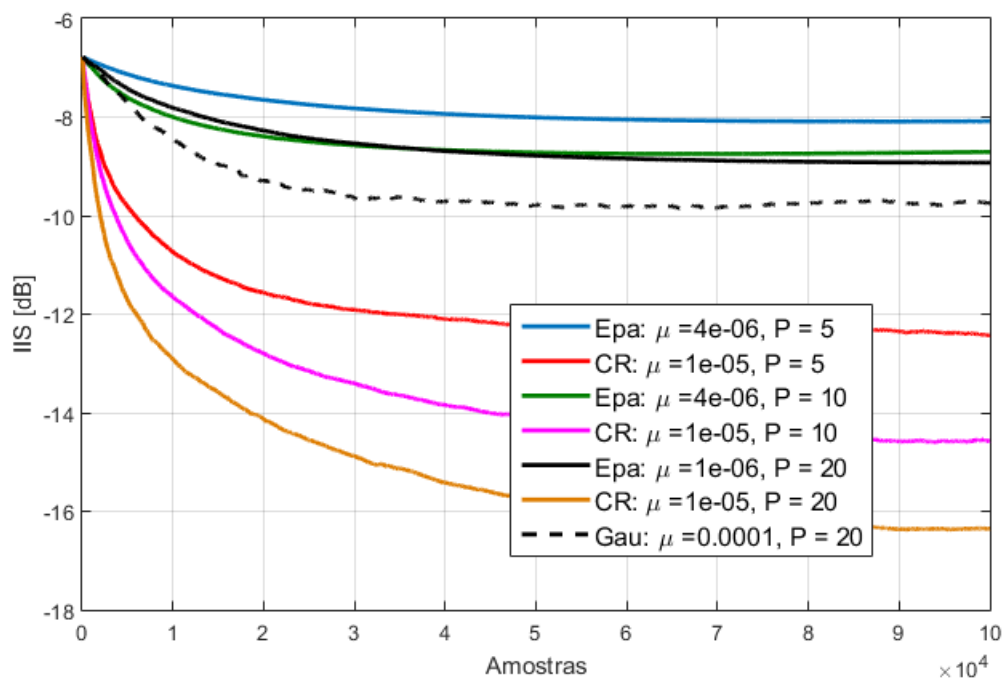


Figura 26: Simulação do COR e CR com  $\sigma = 1$  para o canal  $g(z)$

É possível notar na Figura 26 que os melhores desempenhos foram do CR, independente do valor de  $P$ , obtendo uma IIS residual bem menor no geral. Tanto COR-Epa quanto COR-Gau convergiram num patamar de IIS residual não satisfatório, ainda assim, COR-Gau apresentou um desempenho superior, chegando em torno de -10 dB. Isso nos indica que a informação explorada pelos algoritmos é diferente, alcançando resultados distintos.

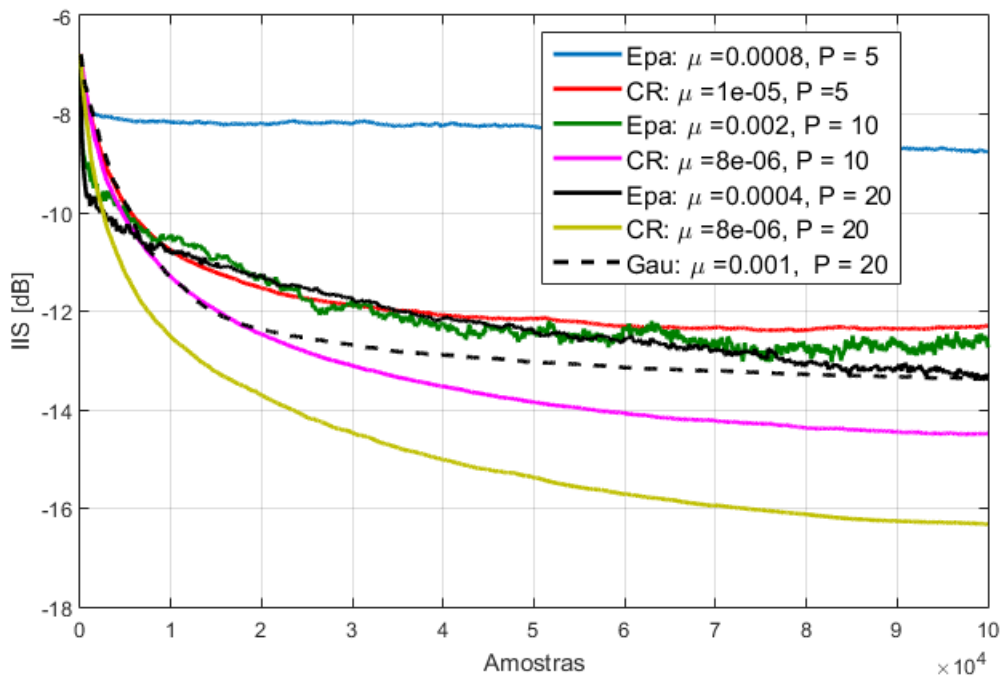


Figura 27: Simulação do COR e CR com  $\sigma = 5$  para o canal  $g(z)$

Na Figura 27, o desempenho dos algoritmos foi obtido usando-se  $N = 50$  e  $\sigma = 5$ . Para  $P = 5$ , utilizamos  $\mu_{CE} = 8 \times 10^{-4}$  e  $\mu_{CR} = 1 \times 10^{-5}$ . Já para  $P = 10$ ,  $\mu_{CE} = 2 \times 10^{-3}$  e  $\mu_{CR} = 8 \times 10^{-6}$ . Enquanto que para  $P = 20$ ,  $\mu_{CE} = 4 \times 10^{-4}$  e  $\mu_{CR} = 8 \times 10^{-6}$  e  $\mu_{CG} = 1 \times 10^{-3}$ .

Neste caso, tanto para o COR-Epa como para o COR-Gau, vemos que os dois convergem para patamares de IIS similares, mostrando como, neste caso, os dois algoritmos convergem para a mesma solução ao aumentarmos o  $\sigma$ , algo já esperado. Além disso, o COR-Epa e o COR-Gau se aproximaram mais do resultado de CR.

## 5 CONCLUSÃO

Critérios de equalização não supervisionados baseados em ITL têm sido bastante utilizados para tratar situações nas quais as técnicas clássicas falham, como no caso da transmissão de sinais correlacionados. Neste trabalho, abordamos algoritmos baseados em critérios que buscam estimar densidades de probabilidade utilizando *kernels* no processo de equalização do sinal.

Estimadores baseados em *kernel* têm sido muito utilizados com este fim, apresentando bons desempenhos. No entanto, dentro do contexto de equalização, a literatura apresenta somente o uso de *kernels* Gaussianos. Como tal função não é ótima para a estimação de funções densidade de probabilidade, neste trabalho, propomos o uso do *kernel* Epanechnikov. Deduzimos novamente os algoritmos SQD, MSQD e COR a partir desta nova função *kernel* e mostramos, dentro de alguns contextos com sinais correlacionados e decorrelacionados, como o uso desta nova função pode melhorar o desempenho dos algoritmos.

Em relação à variação dos parâmetros do SQD e do MSQD com o *kernel* Gaussiano, não é possível determinar uma relação clara entre seus valores e os desempenhos dos algoritmos, apenas podemos afirmar que o SQD possui resultados mais estáveis e uma boa velocidade de convergência para  $\sigma$  próximo de 1 e  $L$  próximo de 2 com sinais decorrelacionados. Por outro lado, o MSQD que não apresentou mudanças quando variou-se o tamanho da janela. Tendo em vista que a variação do tamanho da janela ocasionou poucas mudanças no IIS residual, deve-se ressaltar que o algoritmo é mais lento para valores de  $L$  altos, por isso nem sempre é favorável utilizar janelas grandes.

Ao implementar o *kernel* Epanechnikov no SQD e MSQD, vemos que o desempenho do algoritmo é melhor na maioria dos casos estudados, entretanto houve uma mudança significativa na relação entre o tamanho do *kernel* e a velocidade de convergência, visto que agora ambos SQD-Epa e MSQD-Epa convergem mais rapidamente para  $\sigma$  próximos de 20. Além disso, é possível perceber que o passo de adaptação é muito menor do que o utilizado para o *kernel* Gaussiano. No entanto, o algoritmo continua mais lento para



convergir com valores altos de  $L$ , sendo que o melhor caso persiste em  $L = 2$ .

Já nas simulações com sinais correlacionados, percebeu-se que, ao variarmos o  $\sigma$  do SQD-Epa, o algoritmo converge mais rapidamente e atinge um patamar de IIS mais alto. O MSQD-Epa, por outro lado, apresentou mais dificuldades para convergir ao utilizarmos o filtro  $f_2$ , mas no geral, convergiu de forma mais rápida e com uma IIS residual menor.

Na sequência, estudamos também a medida de correntropia calculada com o *kernel* Epanechnikov. Neste caso, é importante notar que, ao comparar com a correntropia usando o *kernel* Gaussiano, estamos considerando um critério diferente. Isto se deve ao fato de que a escolha do *kernel* influi diretamente na função custo do critério considerado. Por este motivo, iniciamos estudando os critérios resultantes, tanto de forma teórica, buscando aproximações para suas funções custo como através de suas superfícies de erro. Como o *kernel* Epanechnikov é um polinômio de segunda ordem, levando a correntropia a explorar somente estatísticas de ordem superior, achamos interessante também comparar o critério resultante com o *correlation retrieval*, o qual se baseia somente nas correlações entre os sinais envolvidos.

Através desta análise pudemos verificar as semelhanças existentes entre o critério da correntropia com o *kernel* Epanechnikov e o *correlation retrieval*, como esperado. Além disso, visto que a correntropia com o *kernel* Gaussiano também explora mais fortemente os momentos de segunda ordem a partir do momento que se aumenta o valor da largura do *kernel*, vimos como, nesta situação, suas soluções também se aproximam das obtidas pelos dois outros critérios.

Já com relação ao estudo do desempenho do algoritmo baseado na COR utilizando o *kernel* Epanechnikov para sinais descorrelacionados, no geral, vimos que variar os parâmetros afeta principalmente a velocidade de convergência do algoritmo. Assim, o tamanho do *kernel* afeta o algoritmo, de forma que o mesmo converge mais rapidamente à medida que o  $\sigma$  está em torno de 0.25 e 0.5. Quanto ao efeito do tamanho da janela de amostras utilizadas para estimação, podemos notar que o algoritmo converge mais rapidamente para  $N$  maiores. No entanto, tal situação torna o processamento do algoritmo mais lento, não sendo recomendável utilizá-los. Além disso, a curva tende a ter menos

oscilações quando aumentamos o valor de  $N$ .

Em contrapartida, a velocidade de convergência aumenta consideravelmente quando aumentamos o número de intervalos usados na função custo, mesmo comportamento verificado também para o *kernel* Gaussiano. Ademais, vale ressaltar que o valor de  $P$  depende do valor de  $N$ , por isso utilizar um valor alto de  $P$  significa também ter que usar um valor ainda maior para  $N$ , o que tornará o processamento do algoritmo mais lento.

Podemos concluir que o desempenho do algoritmo baseado na COR utilizando o *kernel* Epanechnikov, diante sinais descorrelacionados, chega a ser satisfatório, alcançando a um patamar de IIS residual próximo ao Gaussiano. Porém, o *kernel* Epanechnikov possui um desempenho abaixo em termos de velocidade de convergência, que é um fator importante se tratando de equalização cega.

Implementando o COR-Gau e o COR-Epa com sinais correlacionados, também foi feita a comparação com o CR. Neste caso, o algoritmo tende a ter uma dificuldade para convergir com  $\sigma$  baixos, sendo que o COR-Epa atingiu um patamar de erro alto em relação ao COR-Gau e ao CR com  $\sigma = 1$ . Aumentando o  $\sigma = 5$ , tanto o COR-Epa quanto o COR-Gau passam a atingir um patamar de erro menor e a se aproximar, como previsto em [9, 20] .

## 6 REFERÊNCIAS BIBLIOGRÁFICAS

- [1] HAYKIN, S.; MOHER, M. *Communication Systems*. [S.l.]: Bookman, 1978.
- [2] HAYKIN, S. *Adaptive Filter Theory, Prentice-Hall*. [S.l.]: Prentice-Hall, 1996.
- [3] PRINCIPE, J. C. et al. Information theoretic learning. *Unsupervised adaptive filtering*, Citeseer, v. 1, p. 265–319, 2000.
- [4] SANTAMARÍA, I.; POKHAREL, P. P.; PRINCIPE, J. C. Generalized correlation function: definition, properties, and application to blind equalization. *IEEE Transactions on Signal Processing*, IEEE, v. 54, n. 6, p. 2187–2197, 2006.
- [5] LÁZARO, M.; SANTAMARÍA, I.; PANTALEÓN, C. Matched pdf-based blind equalization. *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2003.
- [6] KRASZCZUK, E.; NEVES, A. Análise de critérios de equalização cega baseados em aprendizagem por teoria da informação. *XXXI Simpósio Brasileiro de Telecomunicações – SBrT2013*, 2013.
- [7] FANTINATO, D.; NEVES, A.; ATTUX, R. Analysis of a novel density matching criterion within the itl framework for blind channel equalization. *Circuits Systems and Signal Processing*, v. 37, p. 203–231, 2018.
- [8] PRINCIPE, J. *Information Theoretic Learning Rényi’s Entropy and Kernel Perspectives*. [S.l.]: Springer, 2010.
- [9] MARTINS, E. *Análise do Desempenho de Algoritmos de Equalização Cega Baseados em ITL com Kernel Epanechnikov*. Tese (Doutorado) — Universidade Federal do ABC, 2018.
- [10] SCOTT, D. W. *Multivariate Density Estimation and Non Parametric Methods*. [S.l.]: Willey, 2015.

- [11] GOIS, L.; NEVES, A. Equalização adaptativa cega de sinais com kernel não-gaussiano. *XXXVII Simpósio Brasileiro de Telecomunicações – SBrT2019*, 2019.
- [12] FANTINATO, D. G. et al. New methods for adaptive equalization based on information theoretic learning= novas metodologias de aprendizado baseado na teoria da informação para equalização adaptativa. [sn], 2017.
- [13] FANTINATO, D. G. et al. Blind deconvolution of correlated sources based on second-order statistics. *SBrT 2013*, 2013.
- [14] GOIS, L. et al. Relações entre critérios baseados na correntropia e estatísticas de segunda ordem para equalização de canais de comunicação. *2020 Brazilian Technology Symposium*, 2020.
- [15] SUYAMA, R. et al. Unsupervised signal processing: Concepts, applications and trends. *Telecommunications: Advances and Trends in Transmission, Networking and Applications*, p. 29, 2006.
- [16] SILVERMAN, B. W. *Density estimation for statistics and data analysis*. [S.l.]: CRC press, 1986. v. 26.
- [17] LÁZARO, M. et al. Stochastic blind equalization based on pdf fitting using parzen estimator. *IEEE Trans. Signal Process*, v. 53(2), p. 696–704, 2005.
- [18] LÁZARO, M. et al. Blind equalization by sampled pdf fitting. *Proceedings of the Fourth International Symposium Independent Component Analysis Blind Equalization*, p. 1041–1046, 2003.
- [19] YANG, Z.; WALDEN, A. T.; MCCOY, E. J. Correntropy: Implications of nongaussianity for the moment expansion and deconvolution. *Signal processing*, Elsevier, v. 91, n. 4, p. 864–876, 2011.
- [20] LIU, W.; PORKHAREL, P. P.; PRINCIPE, J. C. Correntropy: Properties and applications in non-gaussian signal processing. *IEE Transacions on Signal Processing*, 2007.

# Anexos

# Anexo I

GOIS, L. e NEVES, A. "Equalização Adaptativa Cega de Sinais com Kernel Não-Gaussiano", XXXVII Simpósio Brasileiro de Telecomunicações e Processamento de Sinais, 2019, Petrópolis - RJ, 2019.

# Equalização Adaptativa Cega de Sinais com Kernel não-Gaussiano

Lucas Gois e Aline Neves

**Resumo**— A equalização baseada em aprendizagem por Teoria de Informação, por utilizar mais completamente a informação dos sinais envolvidos no processo, é capaz de resolver casos que técnicas clássicas não conseguem, como no caso da transmissão de sinais correlacionados. Estes métodos, no entanto, necessitam da estimação da função densidade de probabilidade dos sinais em tempo real. Normalmente, estimadores baseados no uso de *kernels* são utilizados. Na literatura, a função utilizada é sempre a Gaussiana. No entanto, esta função não é ótima. Neste trabalho aplicamos o kernel Epanechnikov para a estimação e mostramos como o desempenho dos algoritmos pode melhorar com esta nova função.

**Palavras-Chave** — Aprendizagem por Teoria da Informação, distância entre pdfs, kernel Epanechnikov

**Abstract**— Information Theoretic Learning based equalization is capable of solving problems where classic methods fail such as in the case of correlated signals. Nonetheless, these methods need the estimation of the signals probability density function. Usually, kernel based estimators are used, with a Gaussian kernel. However, such function is not optimum. In this work, we apply an Epanechnikov kernel to the estimation and show how the performance of the algorithms may improve

**Keywords** — Information Theoretic Learning, PDF Matching, Epanechnikov kernel

## I. INTRODUÇÃO

O processamento digital de sinais desempenha um papel importante na atualidade, sendo um dos pilares da sociedade moderna. Um sinal, ao ser transmitido, passa por canais de transmissão não ideais que geram distorções no sinal. Visando minimizar o efeito da interferência intersimbólica sofrida pelo sinal original, é proposto algum tipo de processamento no receptor para compensar as distorções sofridas, como a equalização. As técnicas de equalização podem ser divididas em duas categorias: a equalização supervisionada e a não supervisionada.

Na equalização supervisionada há uma sequência de símbolos, conhecida pelo receptor, que é transmitida e usada como referência na adaptação do filtro equalizador. Na equalização não supervisionada ou cega, somente as características estatísticas do sinal transmitido são conhecidas pelo receptor e utilizadas no processo de equalização.

Recentemente, critérios baseados em teoria de informação têm sido utilizados para equalização não supervisionada [7]. Como tais critérios utilizam mais informação sobre os sinais disponíveis, acabam conseguindo resolver casos que as técnicas clássicas não conseguem, como o caso no qual o sinal

transmitido é correlacionado ou casos em que o ruído é impulsivo [7]. Os critérios e algoritmos desenvolvidos neste contexto são chamados de *aprendizagem por teoria da informação* (ITL - Information Theoretic Learning). Um dos critérios de equalização mais utilizados neste contexto consiste em se comparar as densidades de probabilidade dos sinais de saída do equalizador e do sinal transmitido (SQD – Stochastic Quadratic Distance) [1,6]. A estimação das densidades de probabilidade necessárias é feita utilizando-se a janela de Parzen, que nada mais é do que um estimador baseado no uso de kernels. Até o momento, os trabalhos presentes na literatura utilizam somente o kernel gaussiano para executar tal tarefa [3,5,7].

No entanto, conforme discutido em [2,4], o kernel gaussiano não é o melhor quando se trata de estimar densidades de probabilidade. Segundo [2], o kernel Epanechnikov é ótimo neste sentido. Sendo assim, este trabalho desenvolve o algoritmo SQD e SQD modificado [1,8] com este novo kernel, e analisa o seu desempenho quando comparado ao obtido com o uso do kernel gaussiano em diversos cenários de simulação.

Este artigo está organizado da seguinte forma. A seção II contempla os algoritmos baseados em ITL SQD e MSQD, assim como suas funções custo utilizando kernel gaussiano. Na seção III apresentamos o kernel Epanechnikov e os novos algoritmos gerados a partir dele, seguido da seção IV que apresenta os resultados de simulações. Por fim, na seção V é apresentada a conclusão do trabalho.

## II. ALGORITMOS BASEADOS EM APRENDIZAGEM POR TEORIA DA INFORMAÇÃO

A Fig 1 abaixo ilustra, através de um diagrama de blocos, o sistema em estudo. O sinal,  $s(n)$ , é transmitido por um canal com resposta ao impulso  $h(n)$ , resultando no sinal  $x(n)$ . A fim de recuperar o sinal inicialmente transmitido, utilizamos um equalizador,  $w(n)$ . A saída deste filtro é dada por:

$$y(n) = \mathbf{w}^T(n)\mathbf{x}(n)$$

onde  $\mathbf{w}(n)=[w_0 \ w_1 \ \dots \ w_N]$  é o vetor com os coeficientes do equalizador e  $\mathbf{x}(n) = \mathbf{h}^T(n)\mathbf{s}(n) = [x(n) \ x(n-1) \ \dots \ x(n-N)]$ .

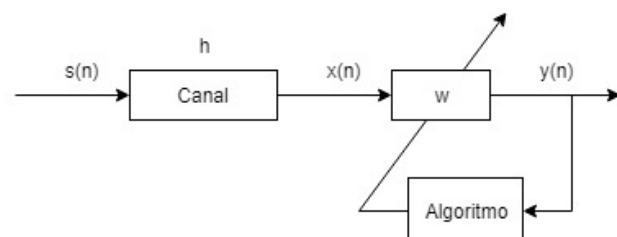


Fig.1 Diagrama de blocos do sistema em estudo.

Em seguida discutiremos os algoritmos a serem estudados neste trabalho.

### A. Algoritmo da Distância Quadrática

Um dos critérios mais utilizados dentro do contexto de ITL é o critério baseado na distância quadrática entre a função densidade de probabilidade (pdf) do sinal na saída do equalizador e do sinal transmitido. Tal critério, proposto inicialmente em [1], foi denominado SQD – Stochastic Quadratic Distance. Sua função custo pode ser definida como [1,6]:

$$J_{SQD}(w) = \int_{-\infty}^{\infty} (f_y(z) - f_s(z))^2 dz \quad (1)$$

onde  $f_y$  e  $f_s$  são, respectivamente, as pdf's do sinal de saída do equalizador e do sinal transmitido inicialmente,  $s(n)$ . Para estimar  $f_y$  e  $f_s$ , é utilizado o método da janela de Parzen [1]:

$$\hat{f}_z(z) = \frac{1}{L} \sum_{i=1}^L K(z - z_i) \quad (2)$$

onde  $L$  é o número de símbolos utilizados para a estimação e  $K(\cdot)$  é a função utilizada como kernel. Em [1,3,7] e nos trabalhos encontrados na literatura, a função utilizada é sempre a gaussiana. Assim, seguindo a proposta de [2], o kernel gaussiano é dado por:

$$K_G(A) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{A^2}{2\sigma^2}} \quad (3)$$

onde  $\sigma$  é o desvio padrão e  $A$  é uma variável aleatória qualquer. O parâmetro  $\sigma$  também é conhecido como tamanho ou largura do kernel e tem um papel crucial na convergência do método como será discutido na seção IV. Como o critério dado em (1) é uma função quadrática, é necessário calcular a convolução entre dois kernels. Neste processo, o kernel gaussiano apresenta uma importante vantagem, pois possui a seguinte propriedade [1]:

$$\int_{-\infty}^{\infty} K_G(y - A_1) K_G(y - A_2) dy = K_{G_{\sigma\sqrt{2}}}(A_1 - A_2) \quad (4)$$

onde  $\sigma\sqrt{2}$  passa a ser o tamanho do kernel. Logo, estimando as pdfs em (1) através de (2), e levando (4) em consideração, chegamos em [1,6]:

$$\begin{aligned} J_{SQD}(w) = & \frac{1}{L^2} \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} K_{G_{\sigma\sqrt{2}}}(|y(n-j)|^2 - |y(n-i)|^2) + \\ & \frac{1}{N_s^2} \sum_{i=1}^{N_s} \sum_{j=1}^{N_s} K_{G_{\sigma\sqrt{2}}}(|d_i|^2 - |d_j|^2) - \\ & \frac{2}{LN_s} \sum_{i=1}^{N_s} \sum_{j=0}^{L-1} K_{G_{\sigma\sqrt{2}}}(|y(n-j)|^2 - |d_i|^2) \end{aligned} \quad (5)$$

onde  $N_s$  é o número de símbolos da constelação do sinal transmitido, enquanto que  $d_i$  são os símbolos da constelação.

Para simplificar a notação, de agora em diante, sempre que tratarmos do kernel gaussiano, faremos  $\sigma = \sigma\sqrt{2}$  e, portanto denotaremos o kernel simplesmente por  $K_G$ .

Utilizando o método do gradiente descendente, chegamos a equação de atualização dos pesos do equalizador:

$$w(n+1) = w(n) - \mu \nabla_w J_{SQD}(w) \quad (6)$$

onde  $\mu$  representa o passo de adaptação e  $\nabla_w J_{SQD}(w)$  é o gradiente de (5). Este algoritmo será denominado SQD-G, por utilizar o kernel gaussiano.

### B. Algoritmo da Distância Quadrática Modificado

O algoritmo SQD Modificado (MSQD), leva em conta apenas o termo cruzado da função custo do algoritmo SQD, já que é este termo que efetivamente mede a distância entre as pdfs do sinal de saída do equalizador e do sinal transmitido [1,7,8]. Assim, sua função custo é dada pelo último termo de (5):

$$J_{MSQD}(w) = -\frac{2}{LN_s} \sum_{i=1}^{N_s} \sum_{j=0}^{L-1} K_G(|y(n-j)|^2 - |d_i|^2) \quad (8)$$

O gradiente de (8) é dado por:

$$\nabla_w J_{MSQD}(w) = -\frac{2}{LN_s} \sum_{i=1}^{N_s} \sum_{j=0}^{L-1} S_{yd} S_{yx} K_G(S_{yd}) \quad (9)$$

onde  $S_{yd} = |y(n-j)|^2 - |d_i|^2$  e  $S_{yx} = y(n-j)x^*(n-j)$  e a equação de atualização dos coeficientes do equalizador segue o que foi feito em (6). Este algoritmo será denotado MSQD-G.

### III. KERNEL EPANECHNIKOV

Segundo o estudo realizado em [2], utilizando-se como medida a média da integral do erro médio quadrático (AMISE – Average Mean Integrated Squared Error) da estimação de pdfs, o kernel Epanechnikov é ótimo para a estimação de densidades de probabilidade. O kernel Epanechnikov é dado por:

$$K_E(A) = \frac{3}{4\sigma} \left( 1 - \left( \frac{A}{\sigma} \right)^2 \right) \quad (10)$$

onde  $\sigma$  é a largura do kernel.

No entanto, para podermos aplicá-lo à estimação do critério SQD, primeiramente precisamos calcular o resultado da convolução entre dois kernels, de forma equivalente ao que foi feito em (4) para o kernel gaussiano. Utilizando-se, portanto, (10) e considerando o caso real, chegamos a:

$$\begin{aligned} \int_{-\sigma}^{\sigma} \frac{3}{4\sigma} \left( 1 - \left( \frac{y - y(n-i)}{\sigma} \right)^2 \right) \frac{3}{4\sigma} \left( 1 - \left( \frac{y - y(n-j)}{\sigma} \right)^2 \right) dy = \\ \frac{9}{8\sigma^3} \left[ \frac{8\sigma^4}{15} - \frac{2\sigma^2}{3} (S_{yy})^2 + M_{yy}^4 \right] \end{aligned} \quad (11)$$



onde  $S_{yy} = |y(n-j)|^2 - |y(n-i)|^2$  e  $M_{yy} = y(n-i)y(n-j)$ . Assim, poderíamos reescrever (5) usando a expressão acima:

$$\begin{aligned} J_{\text{SQD-Epa}}(w) = & \frac{1}{L^2} \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} \frac{9}{8\sigma^3} \left[ \frac{8\sigma^4}{15} - \frac{2\sigma^2}{3} (S_{yy})^2 + M_{yy}^4 \right] \\ & + \frac{1}{N_s^2} \sum_{i=1}^{N_s} \sum_{j=1}^{N_s} \frac{9}{8\sigma^3} \left[ \frac{8\sigma^4}{15} - \frac{2\sigma^2}{3} (S_{dd})^2 + d_i^4 d_j^4 \right] \\ & - \frac{2}{LN_s} \sum_{i=1}^{N_s} \sum_{j=0}^{L-1} \frac{9}{8\sigma^3} \left[ \frac{8\sigma^4}{15} - \frac{2\sigma^2}{3} (S_{yd})^2 + M_{yd}^4 \right] \end{aligned} \quad (12)$$

onde  $S_{dd} = |d_j|^2 - |d_i|^2$ ,  $M_{yd} = y(n-j)d_i$  e  $S_{yd}$  foi definido em (9). A atualização dos coeficientes do equalizador é feita usando-se o método gradiente descendente a partir da função acima, assim como foi feito em (6). Este algoritmo será denominado SQD-Epa.

Para desenvolvimento do MSQD com o kernel Epanechnikov, denominado MSQD-Epa, basta considerarmos somente o último termo de (12). O gradiente será dado então por:

$$\begin{aligned} \nabla_w J_{\text{MSQD-Epa}}(w) = & -\frac{2}{LN_s} \sum_{i=1}^{N_s} \sum_{j=0}^{L-1} \frac{9}{8\sigma^3} \left( \frac{-4\sigma^2}{3} S_{yd} S_{yx} \right) \\ & + 4y^3(n-j) d_i^4 x(n-j) \end{aligned} \quad (13)$$

Na próxima seção, apresentaremos alguns resultados de simulação comparando o SQD e o MSQD originais com suas versões obtidas com o kernel Epanechnikov.

#### IV. RESULTADOS DE SIMULAÇÃO

Foram utilizados dois canais para simulações de desempenho: um canal de fase mista  $C_1(z) = 0.2 + z^{-1} + 0.5z^{-2}$  e um canal de fase mínima  $C_2(z) = 1 + 0.5z^{-1} + 0.1z^{-2}$ . Para todas as simulações, foi utilizado um vetor  $w$  com 21 coeficientes, inicializado com *center spike*, ou seja, o coeficiente  $w_{11} = -1$  e todo o resto nulo.

Como medida de desempenho, utilizamos a interferência intersimbólica residual (IIS) definida como:

$$IIS(n) = \frac{\sum_j |\vartheta_j(n)|^2 - \max_j (|\vartheta_j(n)|^2)}{\max_j (|\vartheta_j(n)|^2)} \quad (14)$$

onde  $\vartheta(n) = [\vartheta_0(n) \ \vartheta_1(n) \ \dots \ \vartheta_M(n)]$  é a resposta combinada entre o canal e o equalizador no instante de tempo  $n$ .

Iniciamos a análise do algoritmo utilizando um sinal  $s(n)$  descorrelacionado, dado pela modulação BPSK (*Binary Phase Shift Keying*). Em seguida, passamos para o uso de um sinal correlacionado.

Para o caso do SQD-G e MSQD-G, como os algoritmos já foram bastante estudados [1], [4], foi considerada a escolha de parâmetros que levam ao melhor desempenho dos algoritmos para cada caso simulado. Os resultados mostrados são uma média de 20 simulações.

#### A. Sinais Descorrelacionados

A fim de analisar o desempenho do algoritmo com relação aos seus parâmetros, i.e., passo de adaptação  $\mu$  e largura do kernel,  $\sigma$ , começamos variando estes parâmetros e observando o desempenho do algoritmo. Como o sinal  $s(n)$  é descorrelacionado, mantivemos  $L=2$  nestas primeiras simulações. Todas as figuras mostram também o desempenho obtido pelo algoritmo com o kernel gaussiano. Os melhores parâmetros, neste caso, foram obtidos do trabalho realizado em [3]. A notação Kg foi usada para simulações com o kernel gaussiano e Ke para o kernel Epanechnikov.

A Fig.2 mostra como a variação do passo afeta o desempenho do SQD para o canal  $C_1(z)$ , mantendo  $\sigma$  fixo igual a 20. Vemos que, em geral, o passo necessário para que o SQD-Epa convirja é bem menor do que o passo usado para o SQD-G. Ainda assim, a figura já mostra como o SQD-Epa pode apresentar um desempenho melhor do que o algoritmo com o kernel gaussiano. O passo  $\mu = 1 \times 10^{-5}$  é o maior valor que pode ser usado para o SQD-Epa que ainda garante a convergência do algoritmo.

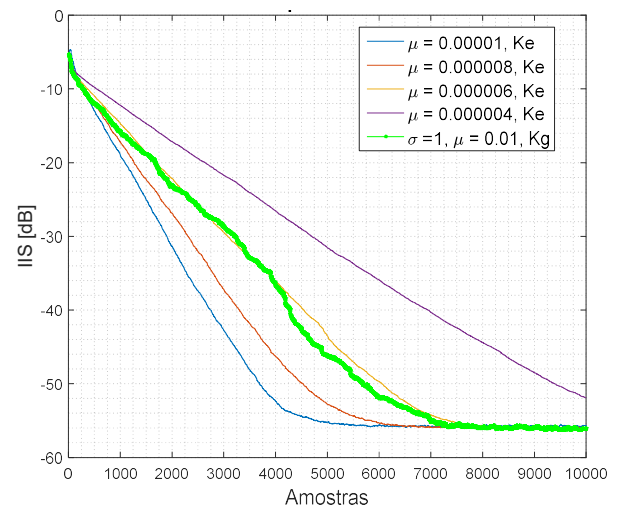


Fig. 2. Desempenho do algoritmo SQD variando o passo de adaptação, para um sinal BPSK utilizando o canal  $C_1(z) = 0.2 + z^{-1} + 0.5z^{-2}$ .

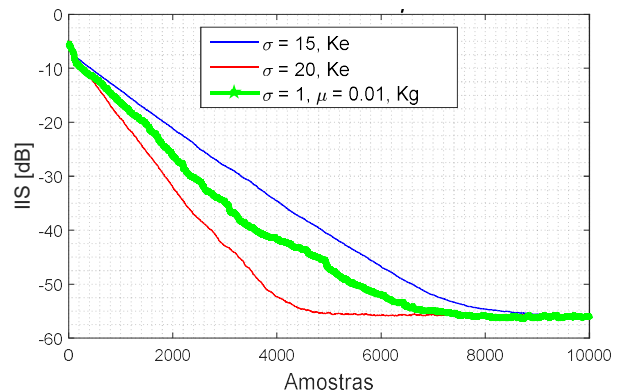


Fig. 3. Desempenho do algoritmo SQD variando o  $\sigma$ , para um sinal BPSK utilizando o canal  $C_1(z) = 0.2 + z^{-1} + 0.5z^{-2}$ .

Ainda mantendo  $C_1(z)$ , a Fig. 3 ilustra como o algoritmo se comporta conforme variamos o valor de  $\sigma$ , mantendo o passo fixo em  $\mu = 1 \times 10^{-5}$ . Diferentemente do SQD-G, para o qual  $\sigma=1$  leva aos melhores resultados, para o SQD-Epa, o uso de valores bem maiores, chegando a  $\sigma=20$ , leva o

algoritmo a apresentar melhores desempenhos. Para valores abaixo desse, a velocidade de convergência do algoritmo tende a diminuir.

A Fig. 4 abaixo ilustra o desempenho do algoritmo SQD-Epa, conforme variamos o tamanho da janela temporal,  $L$ , para o caso do canal  $C_2(z)$ ,  $\mu = 1 \times 10^{-5}$  e  $\sigma = 20$ . Como o sinal utilizado aqui é descorrelacionado, é de se esperar que o algoritmo consiga bons desempenhos com valores baixos para  $L$ . Conforme mostra a Fig. 4, o melhor desempenho é obtido para  $L=2$ , tanto para o caso com kernel gaussiano como para o caso com kernel Epanechnikov. Ainda, por estarmos utilizando neste caso um canal de fase mínima, o desempenho das duas versões do algoritmo conseguem atingir desempenhos bastante semelhantes.

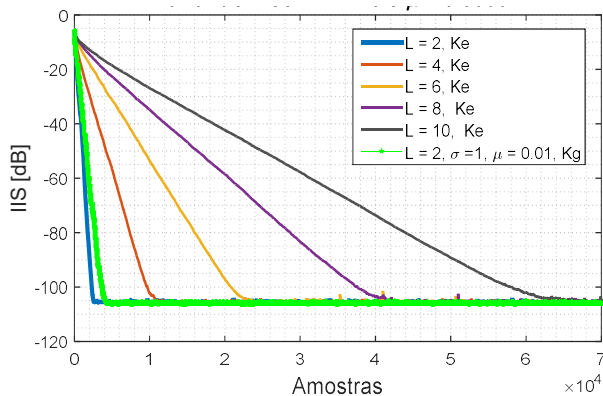


Fig. 4. Desempenho do algoritmo SQD variando o  $L$ , para um sinal BPSK utilizando o canal  $C_2(z) = 1 + 0.5z^{-1} + 0.1z^{-2}$ .

Já para o algoritmo MSQD, a Fig. 5 ilustra a variação do tamanho da janela,  $L$ , já considerando o melhor valor de passo para cada caso e  $C_2(z)$ . Para o MSQD-Epa,  $\sigma = 20$  e mantivemos  $\sigma = 1$  para o MSQD-G. A figura também inclui o melhor desempenho obtido pelo SQD-Epa, para facilitar a comparação.

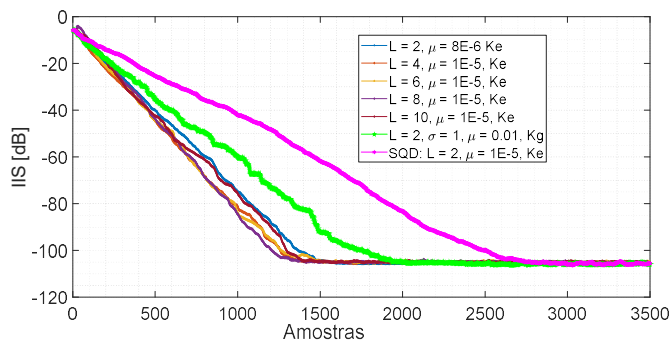


Fig. 5. Desempenho do algoritmo MSQD variando o  $L$ , para um sinal BPSK utilizando o canal  $C_2(z) = 1 + 0.5z^{-1} + 0.1z^{-2}$ , comparado com o desempenho do SQD para  $L = 2$  da Fig. 4.

Na Fig. 5 podemos ver que o MSQD-Epa possui um desempenho superior ao MSQD-G independente do valor de  $L$  utilizado, convergindo mais rapidamente para o mesmo patamar de IIS. Além disso, também é possível perceber que o melhor caso do SQD-Epa, quando  $L=2$ , possui um desempenho inferior, em termos de velocidade de convergência, a todas as simulações do MSQD.

## B. Sinais Correlacionados

Para o estudo dos algoritmos em um contexto com a transmissão de sinais correlacionados, utilizamos o diagrama de blocos mostrado na Fig. 6. Com o objetivo de controlar a correlação do sinal transmitido, filtramos o sinal BPSK por um pré-codificador,  $f(n)$ , a fim de gerar o sinal  $s(n)$  correlacionado que, na sequência, será transmitido pelo canal. Assim, com o processo de equalização continuamos querendo recuperar o sinal  $s(n)$ .

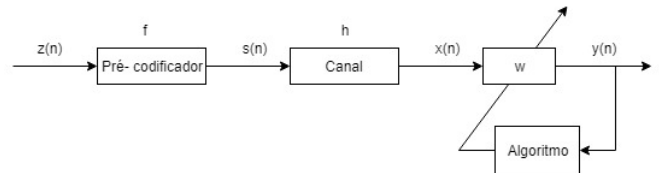


Fig. 6. Diagrama de blocos do sistema utilizando a pré-filtragem.

Os filtros utilizados como pré-codificadores foram  $f_1(z) = 1 + 0.2z^{-1}$  e  $f_2(z) = 1 + 0.5z^{-1} + 0.2z^{-2}$ . É importante notar que o filtro  $f_1(z)$  introduz correlação entre as amostras nos instantes  $n$  e  $n-1$  de  $s(n)$ , enquanto que  $f_2(z)$  introduz entre as amostras nos instantes  $n$ ,  $n-1$  e  $n-2$ . Para todas as simulações a seguir, foi utilizado o canal  $C_2(z)$  e utilizamos o melhor valor de passo de adaptação encontrado. Para o caso em que utilizamos o pré-codificador  $f_1$ , consideramos  $L=2$  e para os casos com  $f_2$ , utilizamos  $L=4$ .

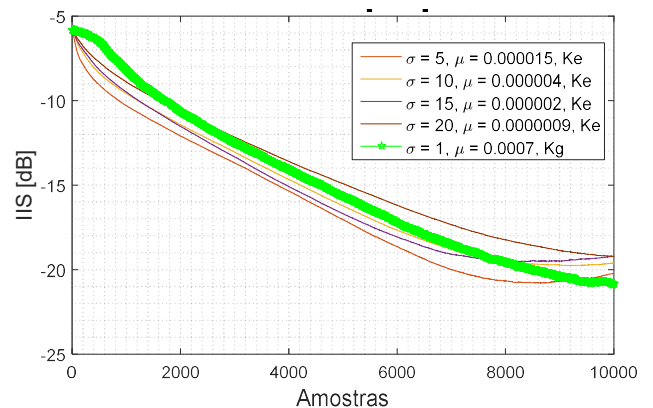


Fig. 7. Desempenho do algoritmo SQD para diversos valores de  $\sigma$  para um sinal BPSK pré-filtrado com  $f_1 = 1 + 0.2z^{-1}$ , utilizando canal o  $C_2(z) = 1 + 0.5z^{-1} + 0.1z^{-2}$ .

Considerando primeiramente o pré-codificador  $f_1$  e o algoritmo SQD, a Fig. 7 mostra o resultado obtido considerando vários valores possíveis para  $\sigma$  no caso do kernel Epanechnikov. Podemos observar que as variações de  $\sigma$  não mudam muito o desempenho do algoritmo e praticamente temos o mesmo desempenho utilizando o SQD-G e o SQD-Epa. Este foi o único caso simulado no qual o SQD-Epa obteve um bom desempenho com valor baixo de  $\sigma$ . Aqui,  $\sigma=5$  levou ao melhor desempenho, levando o algoritmo a convergir mais rápido do que o SQD-G, revertendo a tendência em obter melhores desempenhos com valores de  $\sigma$  mais altos apresentada na Fig.3.

A Fig. 8 abaixo ilustra o desempenho dos algoritmos com o uso do pré-codificador  $f_2$ . Primeiramente, comparando com a Fig. 7, podemos ver que existe uma perda de desempenho neste caso. Na Fig. 7, com  $f_1$ , havia sido possível obter IIS residuais abaixo de  $-20$ dB. Neste caso, o menor valor

atingido fica em torno de -10dB. Assim, temos uma situação mais difícil de ser equalizada. Os melhores desempenhos obtidos pelo SQD-Epa foram com  $\sigma=20$  e  $\sigma=30$ . Apesar do SQD-G apresentar uma oscilação maior, ele consegue atingir um patamar de IIS residual menor que o SQD-Epa.

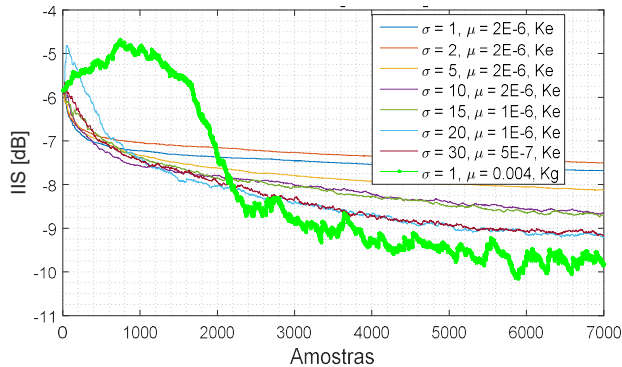


Fig. 8. Desempenho do algoritmo SQD para diversos valores de  $\sigma$  para um sinal BPSK pré-filtrado com  $f_2 = 1 + 0.5z^{-1} + 0.2z^{-2}$  utilizando o canal  $C_2(z) = 1 + 0.5z^{-1} + 0.1z^{-2}$ .

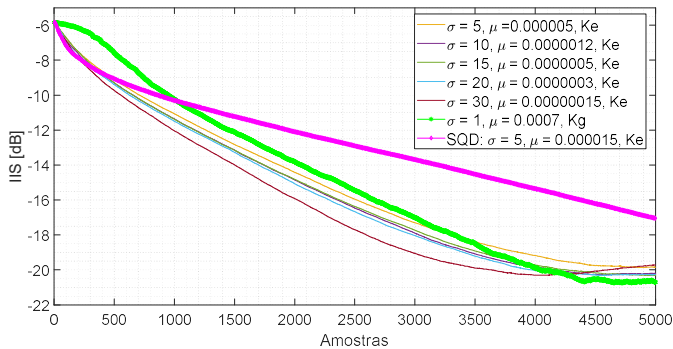


Fig. 9. Desempenho do algoritmo MSQD para diversos valores de  $\sigma$  para um sinal BPSK pré-filtrado  $f_1 = 1 + 0.2z^{-1}$ , utilizando o canal  $C_2(z) = 1 + 0.5z^{-1} + 0.1z^{-2}$ , comparado com o desempenho do SQD para  $\sigma = 5$  da Fig. 7.

Considerando agora o MSQD, a Fig. 9 ilustra o desempenho dos algoritmos utilizando o pré-codificador  $f_1$ . Podemos notar um comportamento bem semelhante ao obtido pelo SQD e mostrado na Fig. 7. A variação no valor de  $\sigma$  influi pouco no desempenho do algoritmo MSQD-Epa. Independentemente do valor de  $\sigma$ , o MSQD-Epa converge mais rapidamente do que o MSQD-G, sendo que o melhor desempenho é obtido com  $\sigma=30$ . Assim como no caso de sinais decorrelacionados (Fig. 5), aqui também podemos observar que o algoritmo MSQD possui um desempenho superior ao SQD, convergindo mais rapidamente.

Utilizando o pré-codificador  $f_2$ , o resultado é mostrado na Fig. 10. Neste caso, independente do kernel utilizado, o MSQD não conseguiu atingir um bom desempenho. Ainda assim, o MSQD-Epa conseguiu convergir para um patamar de IIS residual cerca de 2 dB abaixo do atingido pelo MSQD-G.

## V. CONCLUSÃO

Algoritmos de equalização não supervisionados baseados em ITL têm sido bastante utilizados para tratar situações nas quais as técnicas clássicas falham, como no caso da transmissão de sinais correlacionados. Neste trabalho, abordamos o algoritmo SQD, cujo critério busca igualar as funções densidade de probabilidade da saída do equalizador

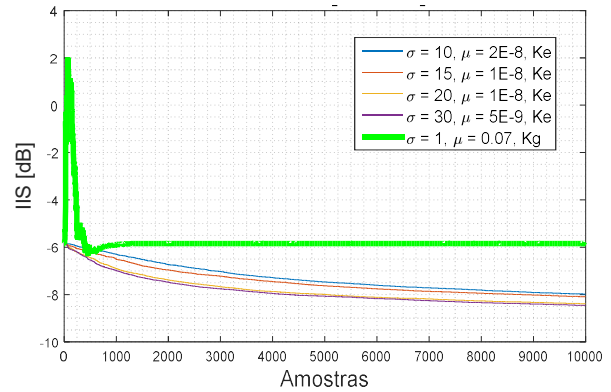


Fig. 10. Desempenho do algoritmo MSQD para diversos valores de  $\sigma$  para um sinal BPSK pré-filtrado  $f_2 = 1 + 0.5z^{-1} + 0.2z^{-2}$ , utilizando o canal  $C_2(z) = 1 + 0.5z^{-1} + 0.1z^{-2}$ .

com a do sinal transmitido. Para isso, é necessário se obter uma estimativa de tais pdfs em tempo real. Estimadores baseados em kernel têm sido muito utilizados com este fim, apresentando bons desempenhos. No entanto, dentro do contexto de equalização, a literatura apresenta somente o uso de kernels gaussianos. Como tal função não é ótima para a estimação de pdfs, neste trabalho, propomos o uso do kernel Epanechnikov. Deduzimos novamente os algoritmos SQD e MSQD a partir desta nova função kernel e mostramos, dentro de alguns contextos com sinais correlacionados e decorrelacionados, como o uso desta nova função pode melhorar o desempenho dos algoritmos.

## AGRADECIMENTOS

Os autores agradecem à FAPESP pelo apoio financeiro.

## REFERÊNCIAS

- [1] M. Lázaro, I. Santamaría and C. Pantaleón, “Matched PDF-Based Blind Equalization”, in IEEE International Conference on Acoustics, Speech and Signal Processing, 2003.
- [2] D. W. Scott, “Multivariate Density Estimation and Non Parametric Methods”. Willey, 2015.
- [3] E. Kraszczuk and A. Neves, “Análise de Critérios de Equalização Cega Baseados em Aprendizagem por Teoria da Informação” in XXXI Simpósio Brasileiro de Telecomunicações – SBTr2013, 2013.
- [4] E. T. S. Martins, “Análise do Desempenho de Algoritmos de Equalização Cega Baseados em ITL com Kernel Epanechnikov” in Dissertação apresentada ao Programa de Pós-graduação em Engenharia da Informação da Universidade Federal do ABC, 2018.
- [5] D. Fantinato;A. Neves; R. Attux. Analysis of a Novel Density Matching Criterion Within the ITL Framework for Blind Channel Equalization., Circuits Systems and Signal Processing, v. 37, p. 203-231, 2018.
- [6] M. Lázaro, I. Santamaría, D. Erdogmus, K. Hild, C. Pantaleón, J.C. Principe, Stochastic blind equalization based on PDF fitting using parzen estimator. IEEE Trans. Signal Process. 53(2), 696–704, 2005
- [7]J.C. Principe, Information Theoretic Learning Rényi’s Entropy and Kernel Perspectives, Springer, Berlin, 2010
- [8] M. Lázaro, I. Santamaría, C. Pantaleón, D. Erdogmus, K. Hild, J.C. Principe, Blind equalization by sampled PDF fitting, in Proceedings of the Fourth International Symposium Independent Component Analysis Blind Equalization, Nara, Japan, 2003, pp. 1041–1046

## Anexo II

GOIS, L.; FANTINATO, D. ; SUYAMA, R. ; NEVES, A. "Relações entre Critérios baseados na correntropia e Estatísticas de Segunda Ordem para Equalização de Canais de Comunicação", 2020 Brazilian Technology Symposium, 2020, Campinas - SP, Proceedings of the 2020 Brazilian Technology Symposium, 2020.

# Relações entre Critérios baseados na Correntropia e Estatísticas de Segunda Ordem para Equalização de Canais de Comunicação

Lucas Gois

Centro de Engenharia, Modelagem e Ciências Sociais Aplicadas  
Universidade Federal do ABC  
Santo André, SP, Brasil  
lucas.gois@aluno.ufabc.edu.br

Ricardo Suyama

Centro de Engenharia, Modelagem e Ciências Sociais Aplicadas  
Universidade Federal do ABC  
Santo André, SP, Brasil  
ricardo.suyama@ufabc.edu.br

Denis Fantinato

Centro de Matemática, Computação e Cognição  
Universidade Federal do ABC  
Santo André, SP, Brasil  
denis.fantinato@ufabc.edu.br

Aline Neves

Centro de Engenharia, Modelagem e Ciências Sociais Aplicadas  
Universidade Federal do ABC  
Santo André, SP, Brasil  
aline.neves@ufabc.edu.br

**Resumo**— A correntropia vem sendo utilizada no processamento de sinais em uma ampla gama de aplicações. Normalmente, a correntropia utiliza um kernel gaussiano, o que faz com que a medida leve em conta todos os momentos estatísticos de ordem par dos sinais envolvidos. Neste artigo, analisamos a correntropia implementada com o kernel Epanechnikov. Visto que este kernel é dado por um polinômio de segunda ordem, comparamos a correntropia aplicada ao problema de equalização com o critério que busca recuperar a correlação entre sinais (*correlation retrieval*). Discutimos então as semelhanças entre os dois, incluindo também a correntropia com o kernel gaussiano na análise.

**Palavras-Chave**— Correntropia, kernel Epanechnikov, Correlation Retrieval, Estatísticas de Segunda Ordem.

**Abstract**— Correntropy has been recently used for signal processing in a large variety of applications. Usually, correntropy uses a Gaussian kernel, which leads to a measure that takes into account all statistical moments even order of the signal. In this paper we analyze correntropy implemented with the Epanechnikov kernel, which is given by a second order polynomial. Considering an equalization scenario, we compare such criterion with the correlation retrieval. We discuss similarities between both criteria, taking also into account correntropy with Gaussian kernel.

**Keywords**— Correntropy, Epanechnikov kernel, Correlation Retrieval, Second order Statistics

## I. INTRODUÇÃO

No contexto de aprendizado de máquina baseado em Teoria da Informação (ITL, do inglês Information Theoretic

Learning), o conceito de correntropia pode ser definido como uma medida de correlação generalizada [1]. Esta medida possui como vantagem uma caracterização estatística dupla: é capaz de extrair diferentes momentos estatísticos sobre os sinais envolvidos e também de sua estrutura temporal [1]. Como na correlação, é possível calcular a correntropia considerando-se diferentes atrasos entre as amostras. Além disso, faz uso de funções kernel, que realizam o mapeamento dos dados para um espaço de dimensão maior (os chamados Reproducing kernel Hilbert space – RKHS) e permitem que as operações sejam feitas nesse espaço, sendo o resultado retornado diretamente no espaço original dos dados [1]. Nesse sentido, a escolha da função kernel é uma etapa fundamental que definirá o espaço de maior dimensão e também quais os momentos estatísticos (podendo ser de ordem superior) serão levados em conta [2].

Devido a estas características, a medida de correntropia tem sido utilizada com sucesso para tratar diversos problemas envolvendo processamento de sinais. Por exemplo, aplicada ao problema de equalização [1], é possível definir um critério para equalização não-supervisionada de sinais com correlação temporal, cenário no qual métodos clássicos como o algoritmo do módulo constante falham [3,10]. Além disso, a correntropia apresenta uma boa robustez na presença de ruído impulsivo [8]. Podem ser encontradas também aplicações para análise de instrumentos musicais [4], reconhecimento de padrões [5], processamento de sinais biomédicos [6,7], separação de fontes [18] além de outros, incluindo contextos não-lineares [9]. Em todos esses casos, escolhe-se o kernel gaussiano, geralmente, pelo fato deste conseguir representar de forma satisfatória as características dos sinais envolvidos. Mas, considerando a

---

Agradecemos à FAPESP (Projeto no 2018/17678-3), e ao Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq, pelo financiamento deste projeto.

existência de outras funções kernel, seria esta uma escolha eficiente?

Em [11,12], foi proposto o uso da correntropia com um kernel não-gaussiano. Mais especificamente, explorou-se o kernel Epanechnikov, que é dado por uma função polinomial quadrática limitada, mostrando que é possível se obter bons desempenhos com a correntropia aplicada ao problema de equalização utilizando-se este kernel. No entanto, um estudo mais detalhado da medida da correntropia dentro deste contexto não foi abordado.

É interessante ressaltar que, pelo fato do kernel Epanechnikov ser uma função polinomial quadrática, utilizando-se este kernel com a correntropia, a medida passa a explorar basicamente estatísticas de segunda ordem, algo que analisaremos mais detalhadamente neste artigo. Dessa forma, tendo em vista o trabalho realizado em [10] e [13], surgiu o interesse em comparar a correntropia calculada com o kernel Epanechnikov com outros critérios que são baseados em estatísticas de segunda ordem como o *Correlation Retrieval* (CR), proposto em [10] e analisado em [13], critério este que, aplicado ao problema de equalização, busca igualar as correlações do sinal transmitido e do sinal recuperado na saída do equalizador. Tanto a correntropia como o critério CR buscam explorar a estrutura temporal do sinal, ou seja, o sinal transmitido precisa possuir uma certa correlação entre as amostras.

O objetivo deste artigo portanto é analisar o desempenho da correntropia utilizando-se o kernel Epanechnikov no problema de equalização de canais e realizar uma comparação com o uso do kernel gaussiano e o critério CR. Para este fim, o trabalho foi estruturado da seguinte forma: o problema de equalização considerado neste trabalho é descrito na Seção II. O critério e os algoritmos baseados na medida de correntropia utilizando o kernel gaussiano e Epanechnikov são apresentados na Seção III. A seção IV apresenta o critério CR. As seções V e VI mostram as relações entre os critérios baseados na correntropia e CR considerando seus momentos estatísticos e suas superfícies de erro. Os desempenhos dos algoritmos na equalização de diferentes cenários são mostrados na Seção VII. Finalmente, as conclusões do trabalho são apresentadas na Seção VIII.

## II. O PROBLEMA DE EQUALIZAÇÃO DE CANAIS

Na Fig. 1 é apresentado o diagrama de blocos de um sistema de comunicação. A fim de recuperar o sinal inicialmente transmitido, utilizamos um equalizador. A saída deste filtro é dada por:

$$y(n) = \mathbf{w}^T(n)\mathbf{x}(n) \quad (1)$$

onde  $\mathbf{w}(n)=[w_0 \ w_1 \ \dots \ w_{L_{eq}}]$  é o vetor com os  $L_{eq}+1$  coeficientes do equalizador e  $\mathbf{x}(n) = \mathbf{h}^T(n)\mathbf{s}(n) = [x(n) \ x(n-1) \ \dots \ x(n-L_{eq})]$  é um vetor com amostras do sinal de saída do canal, com  $\mathbf{h}(n)$  sendo a resposta impulsiva do canal e  $\mathbf{s}(n)$  é o vetor com as amostras do sinal transmitido.

A fim de adaptar os coeficientes do equalizador  $\mathbf{w}$ , iremos considerar métodos autodidatas, i.e., que não necessitam de uma sequência de treinamento, baseados na medida de correntropia, o que será detalhado na próxima seção.

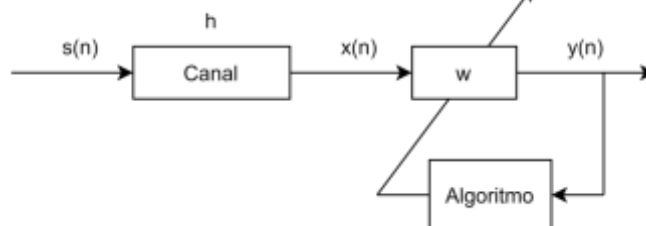


Fig.1: Diagrama de blocos do sistema de comunicação.

## III. MÉTODOS BASEADOS NA CORRENTROPIA

A correntropia, proposta pela primeira vez em [1], pode ser vista como uma nova medida de correlação generalizada, a qual está relacionada à estimativa da entropia quadrática de Renyi através da janela de Parzen [17]. Esta nova medida vem sendo aplicada na equalização cega de canais [1], tendo em vista que a correntropia também é capaz de explorar as características temporais do sinal, assim tornando-a adequada para lidar com sinais correlacionados.

Dentro deste contexto, o critério baseado na medida de correntropia busca minimizar a diferença entre as medidas de correntropia do sinal transmitido e a do sinal estimado, na saída do equalizador, chegando a seguinte função custo:

$$J_{COR}(\mathbf{w}) = \sum_{m=1}^P (V_s[m] - V_y[m])^2 \quad (2)$$

onde  $P$  é o valor de atraso máximo considerado entre as amostras,  $V_s$  é a correntropia do sinal transmitido  $s(n)$  e  $V_y$  é a correntropia do sinal de saída do equalizador  $y(n)$ . A correntropia de um processo estocástico qualquer  $A(n)$  é calculada fazendo-se:

$$V_A(m) = E[K(A(n) - A(n-m))] \quad (3)$$

onde  $E[.]$  denota a esperança matemática,  $K(.)$  é a função kernel,  $A(n)$  é um processo estocástico qualquer e  $m$  é o intervalo entre as amostras. Considerando-se um processo estocástico de tempo discreto, a medida dada em (3) pode ser estimada através de uma média temporal, i.e.:

$$V_A(m) \cong \frac{1}{N-m+1} \sum_{i=0}^{N-m} K(A(n-i) - A(n-m-i)) \quad (4)$$

onde  $N$  é a janela de amostras que usamos para estimar a correntropia.

### A. Correntropia com o Kernel Gaussiano

Em [1,2] e na maioria dos trabalhos encontrados na literatura, a correntropia é calculada utilizando o *kernel* Gaussiano, dado por:

$$K_G(A) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{A^2}{2\sigma^2}}. \quad (5)$$

O desvio padrão  $\sigma$ , também conhecido como tamanho ou largura do *kernel*, é um dos parâmetros fundamentais no cálculo da correntropia, influenciando fortemente no desempenho do critério dado por (2), como será apresentado posteriormente.

A adaptação dos coeficientes do equalizador baseada no critério dado por (2) pode ser efetuada pelo algoritmo do gradiente descendente, que resulta na seguinte regra de atualização dos parâmetros do equalizador:

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu \sum_{m=1}^P (V_s^G[m] - V_y^G[m]) \frac{\partial V_y^G[m]}{\partial \mathbf{w}} \quad (6)$$

onde  $\mu$  representa o passo de adaptação,  $V_s^G$  e  $V_y^G$  representam as correntropias de  $s(n)$  e  $y(n)$  respectivamente, ambas calculadas utilizando-se o *kernel* gaussiano e:

$$\begin{aligned} \frac{\partial V_y^G[m]}{\partial \mathbf{w}} &= \frac{1}{N-m} \sum_{i=n-N-m+1}^n K_G(y(i) - y(i-m)) \times \\ &\times (y(i) - y(i-m))(\mathbf{x}(i) - \mathbf{x}(i-m)) \quad (7) \end{aligned}$$

Este algoritmo será denominado COR-Gau, por utilizar o *kernel* gaussiano.

### B. Correntropia com o Kernel Epanechnikov

O *kernel* Epanechnikov é dado por [11,12]:

$$K_E(A) = \frac{3}{4\sigma} \left(1 - \left(\frac{A}{\sigma}\right)^2\right), \quad |A| < \sigma \quad (8)$$

onde  $\sigma$  é a largura do *kernel*.

Mantendo o mesmo critério dado por (2) e considerando o algoritmo do gradiente descendente para a otimização dado por (6), podemos simplesmente substituir o *kernel* Gaussiano pelo *kernel* Epanechnikov, dado por (8), o que resulta em:

$$\begin{aligned} \frac{\partial V_y^E[m]}{\partial \mathbf{w}} &= \frac{1}{N-m} \sum_{i=n-N-m+1}^n \frac{3}{2\sigma} \left(\frac{y(n-i) - y(n-m-i)}{\sigma^2}\right) \times \\ &\times (\mathbf{x}(n-i) - \mathbf{x}(n-m-i)) \quad (9) \end{aligned}$$

Por conta do *kernel* Epanechnikov, este algoritmo será denominado COR-Epa.

## IV. EQUALIZAÇÃO BASEADA NA CORRELAÇÃO

Proposto em [10], o *Correlation Retrieval* (CR) faz uso exclusivo de momentos estatísticos de segunda ordem, explorando a estrutura temporal do sinal. O critério CR foi proposto inspirado em técnicas utilizadas no problema de separação cega de fontes, baseadas exclusivamente em estatísticas de segunda ordem como o AMUSE ou o SOBI [14, 15, 16]. Tais métodos exploram a estrutura temporal do sinal para conseguir separar as fontes sem usar estatísticas de ordem superior. O mesmo acontece com o critério CR: explorando-se a correlação dos sinais para vários atrasos entre amostras diferentes, tem-se informação suficiente para equalizar o canal. A função custo busca medir a diferença entre as autocorrelações do sinal transmitido e o sinal obtido na saída do equalizador, para diferentes atrasos entre amostras:

$$J_{CR}(\mathbf{w}) = \sum_{m=0}^P (r_y(m) - r_s(m))^2 \quad (10)$$

onde  $r_s(m) = E[s(n)s(n-m)]$  e  $r_y(m) = E[y(n)y(n-m)]$ , sendo  $P$  o valor máximo do atraso e considerando sinais reais. As correlações necessárias em (10) podem ser estimadas a partir de médias temporais:

$$\hat{r}_y(m) = \frac{1}{N} \sum_{i=0}^{N-1} y(n-i)y(n-m-i) \quad (11)$$

onde  $N$  denota o número amostras utilizadas para estimação.

A função custo pode ser minimizada utilizando-se o método do gradiente descendente:

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu_{CR} \sum_{m=0}^P (r_y(m) - r_s(m)) \frac{\partial r_y(m)}{\partial \mathbf{w}} \quad (12)$$

onde  $\mu_{CR}$  é o passo de adaptação e

$$\frac{\partial r_y(m)}{\partial \mathbf{w}} = \frac{1}{N-m} \sum_{i=n-N-m+1}^n y(n-i)\mathbf{x}(\beta) - y(\beta)\mathbf{x}(n-i)$$

com  $\beta = n - m - i$ .

## V. COMPARAÇÃO ENTRE OS CRITÉRIOS

Podemos iniciar comparando o critério baseado na correntropia implementada com o *kernel* Epanechnikov (COR-Epa) e o critério CR. Através da equação (2), podemos notar que o primeiro busca igualar as correntropias do sinal transmitido com a do sinal de saída do equalizador. Utilizando (3) e (8), a correntropia com o *kernel* Epanechnikov é definida como:

$$\begin{aligned} V_A^E(m) &= E \left[ \frac{3}{4\sigma} \left(1 - \left(\frac{A(n) - A(n-m)}{\sigma}\right)^2\right) \right] \\ &= \frac{3}{4\sigma} - \frac{3}{2\sigma^3} E[A^2(n)] + \frac{3}{2\sigma^3} E[A(n)A(n-m)] \end{aligned}$$

$$= \frac{3}{4\sigma} - \frac{3}{2\sigma^3}(\sigma_A^2 - r_A(m)) \quad (13)$$

ou seja, a medida leva em conta a variância e a autocorrelação do sinal com atraso  $m$ .

Assim, ao utilizar a correntropia com o kernel Epanechnikov na definição da função custo em (2), basicamente estamos comparando as variâncias e as autocorrelações dos sinais  $s(n)$  e  $y(n)$ , considerando diversos atrasos entre as amostras,  $m$ .

Como podemos entender o termo constante em (13) (termo  $\frac{3}{4\sigma}$ ) e a variância do sinal ( $\sigma_A^2$ ) como sendo simplesmente uma restrição de amplitude, podemos considerar a seguinte aproximação:

$$V_y^E(m) \approx r_y(m)$$

Considerando que esta mesma aproximação vale também para  $V_s^E(m) \approx r_s(m)$ , a função custo do COR-Epa (2) acaba sendo aproximadamente equivalente à função custo CR, dada por (10).

Já com relação à correntropia utilizando o *kernel* gaussiano, conforme análise realizada em [1] e [2], através da expansão em série de Taylor de (5), é possível se verificar que a medida leva em conta todos os momentos de ordem par do sinal em questão [1,2].

$$V_A^G = \frac{1}{\sqrt{2\pi}\sigma} \sum_{k=0}^{\infty} \frac{(-1)^k}{2^k \sigma^{2k} k!} E[A^{2k}] \quad (14)$$

Por outro lado, considerando só a expansão até segunda ordem, em torno de zero, obtém-se [10]:

$$V_A^G(m) \approx \frac{1}{\sqrt{2\pi}\sigma} \left( 1 - \frac{1}{\sigma^2}(\sigma_A^2 - r_A(m)) \right) \quad (15)$$

A semelhança de (15) com (13) é bastante direta e mostra que, independente do *kernel* escolhido para o cálculo das correntropias em (2), as funções custo resultantes terão algumas similaridades. No entanto, como (15) representa uma aproximação até segunda ordem para  $V_y^G(m)$ , as superfícies de erro serão diferentes e os pontos de mínimo não necessariamente serão os mesmos entre os critérios COR-Gau e COR-Epa. Além disso, a partir de (14) é possível perceber que quanto maior o valor da largura do *kernel*  $\sigma$ , menor é a contribuição dos momentos de ordem superior em  $V_A^G$ . Sendo assim, é de se esperar que aumentando o valor de  $\sigma$  a semelhança entre os critérios COR-Gau e COR-Epa aumente. Isto poderá ser visto na seção VI através da análise das superfícies de erro.

## VI. SUPERFÍCIES DE ERRO

Após a discussão sobre as semelhanças entre os critérios sendo estudados aqui, buscaremos ilustrar as relações discutidas na Seção V através das superfícies de erro dos critérios COR-Gau, COR-Epa e CR.

Iniciamos o nosso estudo considerando um sinal  $s(n)$  independente e identicamente distribuído, ou seja, sem estrutura temporal, dado por um sinal BPSK (*Binary Phase Shift Keying*) no qual os símbolos variam entre  $-1$  e  $+1$  com igual probabilidade. Para podermos visualizar mais facilmente a superfície de erro, foi considerado um equalizador com dois coeficientes e o canal  $H(z) = 1 + 0.6z^{-1}$ . As figuras foram obtidas usando-se  $P=5$ .

Considerando, primeiramente, os critérios COR-Epa e CR, a Fig. 2 mostra a superfície de erro dos dois, considerando, no caso do COR-Epa,  $\sigma=1$ . Podemos observar a grande semelhança entre as duas superfícies com relação à posição dos mínimos globais e locais. No entanto, também fica clara a diferença no formato das superfícies.

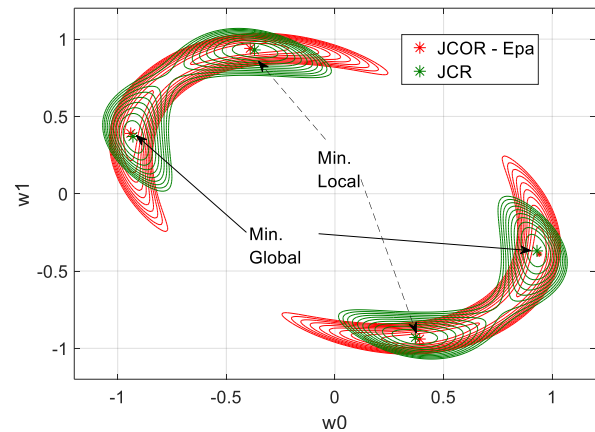


Fig. 2. Superfícies de erro COR-Epa e CR, BPSK,  $\sigma = 1$

Já comparando o COR-Epa com o COR-Gau, para  $\sigma=1$ , conforme mostrado na Fig. 3, vemos como existe uma diferença na posição dos mínimos. Por outro lado, como mostra a equação (14), conforme se aumenta o valor de  $\sigma$ , espera-se que a semelhança entre os dois critérios aumente. Isto pode ser visualizado comparando-se as Figuras 3 e 4.

Na Fig. 4 foi utilizado  $\sigma=5$ . Aqui mostramos só a parte superior da superfície de erro, para facilitar a visualização. Neste caso os pontos de mínimo das duas superfícies praticamente se sobrepõem.

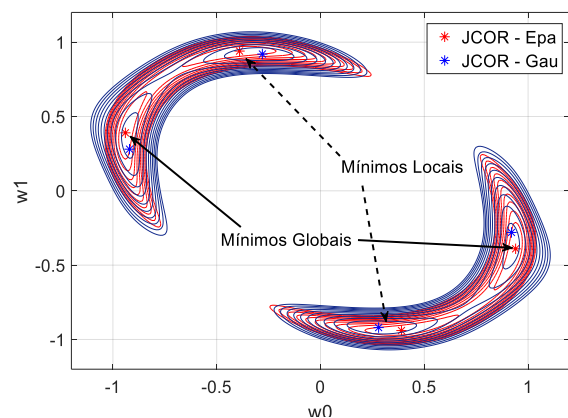
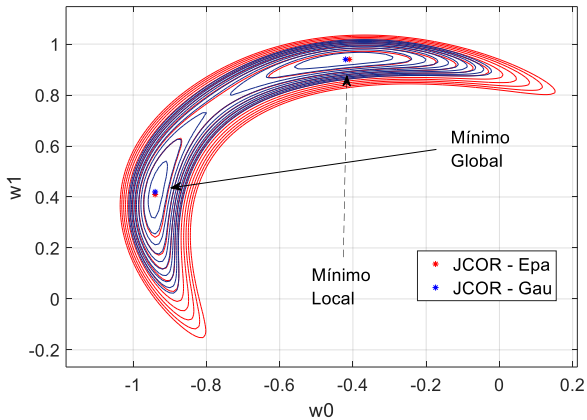
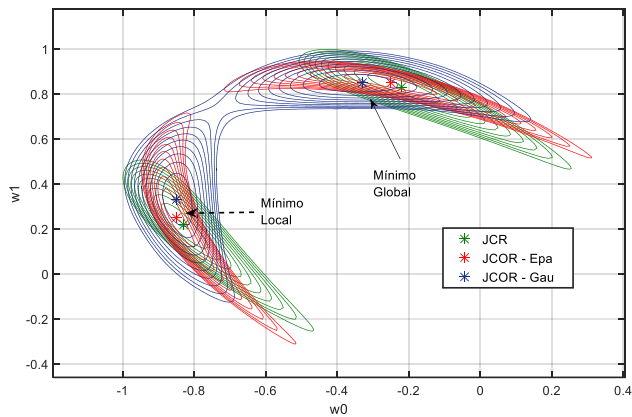
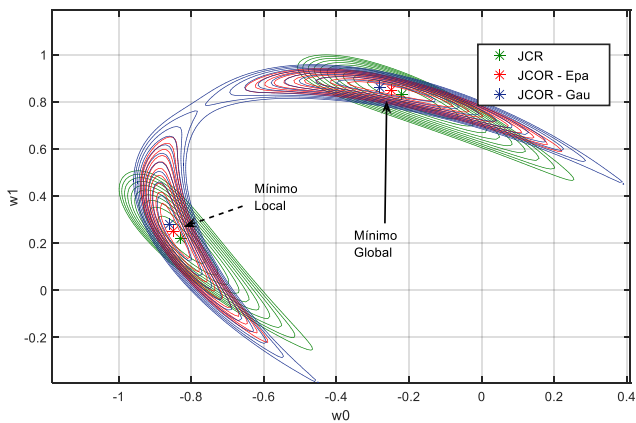


Fig. 3. Superfícies de erro COR-Epa e COR-Gau, BPSK,  $\sigma = 1$ .




 Fig. 4: Superfícies de erro COR-Epa e COR-Gau, BPSK,  $\sigma = 5$ .

Considerando agora o caso em que a fonte é um sinal correlacionado, consideramos novamente um sinal BPSK e filtramos por um pré-codificador dado por  $F(z) = 1 + 0.5z^{-1}$ . O resultado deste processo de filtragem será o sinal  $s(n)$ . As Figs. 5 ( $\sigma=1$ ) e 6 ( $\sigma=5$ ) mostram os resultados obtidos. Neste caso a diferença entre as posições dos mínimos é maior do que no caso dos sinais descorrelacionados. Ainda assim, aumentando o valor da largura do *kernel*  $\sigma$ , vemos que as posições dos mínimos se aproximam.


 Fig. 5: Superfícies de erro, sinal correlacionado,  $\sigma = 1$ .

 Fig. 6: Superfícies de erro, sinal correlacionado,  $\sigma = 5$ .

## VII. RESULTADOS DE SIMULAÇÃO

Nesta seção, analisaremos o desempenho do algoritmo COR-Epa, variando os seus parâmetros e comparando com o algoritmo baseado no CR e no COR-Gau. A interferência intersimbólica residual (IIS), medida de desempenho utilizada, é definida como:

$$IIS(n) = \frac{\sum_j |\vartheta_j(n)|^2 - \max_j (|\vartheta_j(n)|^2)}{\max_j (|\vartheta_j(n)|^2)} \quad (16)$$

onde  $\vartheta(n) = [\vartheta_0(n) \ \vartheta_1(n) \ \dots \ \vartheta_M(n)]$  é a resposta combinada entre o canal e o equalizador (obtida através da convolução de  $h(n)$  com  $w(n)$  no instante de tempo  $n$ ).

Utilizaremos novamente o sinal  $s(n)$  dado por um sinal BPSK filtrado por  $F(z) = 1 + 0.5z^{-1}$ . O canal é dado por  $H(z) = 0.2 + 1z^{-1} + 0.4z^{-2} + 0.1z^{-3}$ . O equalizador será um vetor  $w$  de tamanho 21, iniciado como um vetor *center spike*, ou seja, o vetor é todo nulo a menos do elemento central (posição 11) que é definido como sendo igual a 1. As figuras mostram uma média de 30 simulações.

Na Fig. 7, vemos os resultados obtidos utilizando-se  $N=50$ ,  $P=10$  e  $P=20$  para os algoritmos COR-Epa, COR-Gau e CR. Os passos de adaptação utilizados foram: para  $P=10$ ,  $\mu_{CE}=0.02$  (COR-Epa),  $\mu_{CR}=1e-5$  e  $\mu_{CG}=0.05$  (COR-Gau). Já para  $P=20$ , foram utilizados  $\mu_{CE}=0.0007$ ,  $\mu_{CR}=1e-5$  e  $\mu_{CG}=0.004$ . Com um valor de  $\sigma=5$  tanto para o COR-Epa como para o COR-Gau, vemos que os dois algoritmos convergem para patamares de IIS similares, o que já era esperado visto que os pontos de mínimos das funções custo, conforme visto as seções III, se aproximam conforme o valor de  $\sigma$  aumenta. Por outro lado, as superfícies de erro possuem formatos diferentes, o que faz com que os dois algoritmos tenham comportamentos diferentes: o COR-EPA converge mais rápido considerando  $P=10$ , mas a situação se inverte quando consideramos  $P=20$ . Os passos de adaptação foram escolhidos de forma que os algoritmos convergissem para o mesmo patamar de IIS. Além disso, podemos observar que o aumento no valor de  $P$  favorece mais o desempenho do COR-Gau do que o do COR-Epa.

Já o algoritmo CR converge para patamares de erro bem menores do que os outros dois algoritmos, chegando a uma diferença de cerca de 3 dB no caso de  $P=20$ . Neste caso, o passo de adaptação foi escolhido de forma que o algoritmo apresentasse a mesma velocidade de convergência obtida pelos algoritmos baseados na correntropia. Como pode ser visto na Fig. 7, a solução encontrada pelo algoritmo CR se mostra melhor para resolver o problema em questão.

Na Fig.8, o desempenho dos algoritmos foi obtido usando-se  $N=100$ ,  $P=10$ ,  $\sigma=5$ ,  $\mu_{CE}=0.002$ ,  $\mu_{CR}=1e-5$  e  $\mu_{CG}=0.3$ . Neste caso vemos como o COR-Epa e o CR convergem para o mesmo patamar de IIS, ainda que o segundo convirja mais rápido, mostrando como, neste caso, os dois algoritmos convergem para a mesma solução, conforme discutido na seção III. Neste caso, não foi possível obter um desempenho

melhor para o algoritmo COR-Gau, que acabou convergindo para um patamar de IIS superior ao dos demais algoritmos.

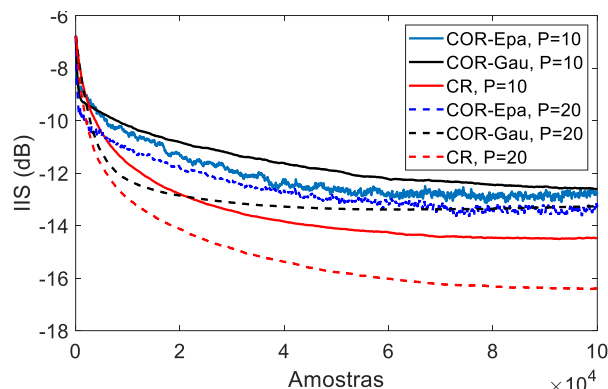


Fig. 7: Desempenho dos algoritmos,  $\sigma=5$ ,  $N=50$

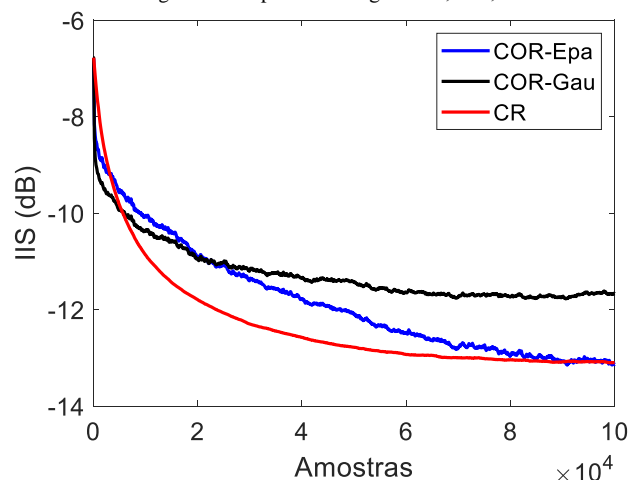


Fig. 8: Desempenho dos algoritmos,  $\sigma=5$ ,  $N=100$ ,  $P=10$

### VIII. CONCLUSÃO

Dentro do contexto de equalização cega, algoritmos baseados em ITL são utilizados para alcançar resultados que técnicas clássicas não conseguem. Um destes critérios de equalização utiliza uma nova medida de correlação generalizada, chamada de correntropia. Na literatura é comum encontrarmos a correntropia sendo calculada com o uso de um *kernel* gaussiano. Neste trabalho, analisamos as características da correntropia calculada utilizando-se o *kernel* Epanechnikov. Como este é dado por um polinômio de segunda ordem, vimos como, neste caso, a correntropia passa a explorar basicamente as estatísticas de segunda ordem dos sinais envolvidos. Por este motivo, comparamos o critério resultante com o *correlation retrieval*, critério que é baseado somente nas correlações e também explora a estrutura temporal dos sinais. Mostramos como realmente os dois critérios possuem soluções bastante próximas, mas as superfícies de erro são diferentes, o que leva o desempenho dos algoritmos baseados no gradiente descendente deste dois critérios a também serem diferentes. No caso da correntropia, a superfície depende fortemente do valor da largura do *kernel*.

Por outro lado, vimos que, dependendo do valor de  $\sigma$ , estes critérios também se aproximam das soluções apresentadas pela correntropia obtida com o *kernel* gaussiano.

### REFERÊNCIAS

- [1] Santamaria, P. Puskal e J. Principe, "Generalized Correlation Function: Definition, Properties and Application to Blind Equalization," IEEE Transactions on Signal Processing, vol. 54, n. 6, pp. 2187-2197, 2006.
- [2] Z. Yang, A. Walden e E. McCoy, "Correntropy: Implications of nonGaussianity for the moment expansion and deconvolution," Signal Processing, n. 91, pp. 864-876, 2011.
- [3] Neves, A.; Wada, C.; Suyama, R.; Attux, R.; ROMANO, J. M. T. . An Analysis of Unsupervised Signal Processing Methods in the Context of Correlated Sources. Lecture Notes in Computer Science, v. 5441, p. 82-89, 2009.
- [4] M. E. Özbek e F. A. Savaci, "Correntropy function for fundamental frequency determination of musical instrument samples," Expert Systems with Applications, n. 38, pp. 10025-10030, 2011.
- [5] H. Yan, X. Yuan, S. Yan e J. Yang, "Correntropy based features election using binary projection," Pattern Recognition, n. 44, pp. 2834-2842, 2011.
- [6] A. Garde, L. Sömmo, R. Jané e B. F. Giraldo, "Correntropy-based Nonlinearity Test Applied to Patients with Chronic Heart Failure," em 32nd Annual International Conference of the IEEE EMBS, Buenos Aires, 2010.
- [7] M. Hasson, J. Terrien, C. Marque e B. Karlsson, "Comparison between approximate entropy, correntropy and time reversibility: Application to uterine electromyogram signals," Medical Engineering & Physics, vol. 33, n. 8, pp. 980-986, 2011.
- [8] Liu, Weifeng & Principe, Jose., "Correntropy: Properties and Applications in Non-Gaussian Signal Processing". Signal Processing, IEEE Transactions on. 55. 5286 – 5298, 2007. 10.1109/TSP.2007.896065.
- [9] Liu, Weifeng & Principe, Jose., "Correntropy: A Localized Similarity Measure". 4919 – 4924, 2006 10.1109/IJCNN.2006.247192.
- [10] D. Fantinato, R.R. Attux, A. Neves, J.M. Romano, "Blind Deconvolution of Correlated Sources Based on Second-Order Statistics". In: XXXI Simpósio Brasileiro de Telecomunicações, Fortaleza, 2013.
- [11] E.T. Martins, A. Neves, "Análise da Eficiência da Utilização do Kernel Epanechnikov Aplicado à Correntropia". In: VII Simpósio de Processamento de Sinais, São Bernardo do Campo – SP, 2017.
- [12] E.T. Martins. Análise do Desempenho de Algoritmos de Equalização Cega Baseados em ITL com Kernel Epanechnikov, Mestrado, Engenharia da Informação, Universidade Federal do ABC, 2018.
- [13] D. Fantinato, "New Methods for Adaptive Equalization Based on Information Theoretic Learning", Tese de doutorado, Faculdade de Engenharia Elétrica e de Computação, UNICAMP, 2017.
- [14] L. Tong, V.C. Soon, Y. Huang, R. Liu, "AMUSE: a new blind identification algorithm", Proceedings of IEEE ISCAS, pp. 1784-1787, Vol. 3, New Orleans, LA, 1990.
- [15] A. Belouchrani, K.K. Abed-Meraim, J.F. Cardoso, E. Moulines, "Second order blind separation of correlated sources", Proceedings of International Conference on Digital Signal Processing, pp. 346-351, 1993.
- [16] A. Belouchrani, K. A. Meraim, J.F. Cardoso, E. Moulines, A blind source separation technique based on second order statistics, IEEE Trans. on Signal Processing, v. 45, n. 2, pp. 434-444, 1997.
- [17] J. C. Principe, D. Xu e J. W. Fisher III, "Information-Theoretic Learning" em Unsupervised Adaptive, Nova York, Wiley, 2000.
- [18] R. Li, W. Liu, J.C. Principe, A unifying criterion for instantaneous blind source separation based on correntropy, Signal Processing 87 (2007) 1872–1881.