Kernel RLS Equalizer for Orthogonal Time Frequency Space Signaling (OTFS)

MAHA GEORGE ZIA
Department of Electrical Engineering
Salahaddin University
Kurdistan region- Erbil
IRAQ

Abstract: - Orthogonal Time Frequency Space (OTFS) modulation has emerged as a promising waveform for high-mobility wireless communications because it exploits the delay-Doppler domain to improve robustness over time-varying multipath channels. Nevertheless, nonlinear distortion and multipath fading can degrade orthogonality and lead to poor bit error rate (BER) performance. In this paper, a kernel recursive least squares (KRLS)-based nonlinear equalizer is integrated with the overlap-save method (OSM) to mitigate channel impairments in OTFS systems operating over the Extended Vehicular A (EVA) channel model. A modified KRLS algorithm is also introduced to improve numerical stability and reliability. Simulation results show that the proposed OTFS system using modified KRLS with OSM achieves better BER performance than kernel least mean square (KLMS), kernel affine projection (KAP), and traditional KRLS equalizers, demonstrating its effectiveness for high-mobility scenarios.

Key-Words: - OTFS modulation, kernel recursive least (KRLS), nonlinear adaptive filter, frequency domain equalizer, adaptive equalizer, overlap-save method.

Received: July 16, 2025. Revised: March 19, 2026. Accepted: April 11, 2025. Published: May 28, 2026.

1 Introduction

Orthogonal Time Frequency Space (OTFS) modulation is one of promising techniques in 5G and 6G, because it is more robust to high Doppler spread, phase noise and provides high diversity order [1,2,3]. OTFS loses orthogonality because of fading over multipath fading channel, therefore it is coupled with a suitable equalizer [4] to exploit the full channel diversity over both time and frequency [5].

Several channel equalization methods were presented for OTFS modulation for single user and multi users. First, a Low-complexity minimum mean square error (MMSE) and zero-forcing (ZF) equalizers for OTFS signal detection were proposed in [6]. Their proposed approach achieves exact MMSE and ZF solutions at a much lower complexity compared to the matrix inversion approach by recognizing a certain structure in the effective delay-Doppler channel matrix in OTFS modulation.

A time domain low complexity linear minimum mean square error (MMSE) equalization and successive interference cancellation (SIC) receiver for LDPC (low density parity check) coded cyclic prefix (CP)-OTFS were described in [7]. It was presented that CP-OTFS has significantly large tolerance to residual carrier frequency offset (CFO), however it is not completely immune to it. Also, it was found that SIC can bring the performance close to linear minimum mean square (LMMSE) with ideal channel estimates.

In [8], a doubly iterative sparsified minimum mean square error (DI-S-MMSE) turbo equalizer for OTFS modulation was proposed, it iteratively exchanges the extrinsic information between a soft-input-soft-output (SISO) MMSE estimator and a SISO decoder. This equalizer does not suffer from short loops, and its performance is comparable to that of the near-optimal symbol-wise maximum a posteriori (MAP) algorithm, but it maintains a linear complexity.

In this work, the integration of a nonlinear time domain equalizer-based kernel recursive least square (KRLS) algorithm with the overlap-save method (OSM) represents a significant advancement for Orthogonal Time Frequency Space (OTFS) modulation within the Extended Vehicular A (EVA) channel model. The modified KRLS algorithm, effectively offers more reliable performance compared with Kernel least mean square (KLMS) algorithm, Kernel affine projection (KAP) algorithm and traditional KRLS algorithm. Also, it enhances the bit error rate (BER) performance and ensures more reliable performance for OTFS modulation. The remainder of this paper is organized as follows: section 2 is dedicated to the proposed OTFS modulation system, incorporating a Kernel Recursive Least Squares (KRLS) equalizer and overlap-save method (OSM). Section 3 describes a nonlinear adaptive filtering, where Kernel least mean square (KLMS) algorithm, Kernel affine projection (KAP) algorithm, kernel recursive least (KRLS) algorithm and modified kernel recursive least (KRLS) are presented. The bit error rate performance for the proposed OTFS modulation system is analysed through simulations and is presented in section 4. Finally, section 5 draws the conclusion. *Notation*: scalars are represented as small italic letters, vectors are represented as small bold letters, and matrices are represented as capital bold letters.

2 Proposed OTFS Modulation System

Consider OTFS modulation system [6]. The proposed OTFS modulation incorporating adaptive equalizer (Kernel Recursive Least Squares (KRLS)) and overlap-save method (OSM) is shown in Fig, 1.
The information symbols $x[k,l]$ are treated as points on the 2D $N \times M$ delay-Doppler grid, where M are the sub-carriers, each of Δf Hz bandwidth, and N are the symbols of duration $T = {}^{1}/_{\Delta f}$. The bandwidth is $B = M\Delta f$.
These symbols are mapped to time-frequency plane $X[n,m]$ using inverse symplectic finite Fourier transform (ISFFT), as in [6].

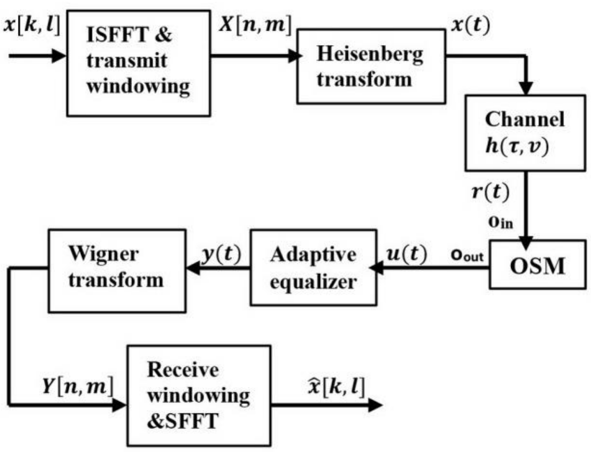

Fig. 1. Block diagram of the proposed OTFS modulation System with OSM and KRLS equalization.

$$X[n,m] = \frac{1}{NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} x[k,l] e^{j2\pi\left(\frac{nk}{N} - \frac{ml}{M}\right)}. \quad (1)$$

$X[n,m]$ is converted to a time domain $x(t)$ for transmission using Heisenberg transform, as in [6].

$$x(t) = \sum_{n-0}^{N-1} \sum_{m=0}^{M-1} X[n,m]\, g(l) e^{j2\pi m \Delta f(l)}. \quad (2)$$

Where the transmit pulse shape is denoted by $g(l)$, and $l = t - nT$. The $x(t)$ signal is then transmitted through the time varying wireless channel [6].
The received time domain signal $r(t)$ enters overlap-save method (OSM) block. In Fig. 2 the time domain vector at point (o_{in}) is divided into a sequence of B-symbol blocks($B \leq N_B$) with certain overlap of length N_B [9]. N_B is the FFT size used to convert these blocks to a frequency domain. The frequency domain signal block is given below [10]:

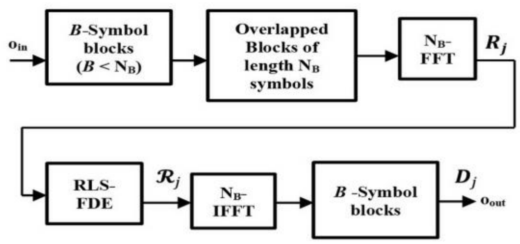

Fig. 2. Block diagram of the overlap-save method (OSM).

$$\boldsymbol{R_j}(k) = \mathbf{H}_{i,j}(k)\, \mathbf{S}(k) + \boldsymbol{\Delta_j}(k), \;\; k = 0,1, \dots, N_B. \quad (3)$$

Where $\mathbf{H}_{i,j}(k)$, $\mathbf{S}(k)$, and $\boldsymbol{\Delta_j}(k)$ are respectively the channel gain, the signal component, and noise component due to AWGN. Now, each frequency component $\boldsymbol{R_j}(k)$ is multiplied by the adaptive recursive least square (RLS) frequency domain equalizer (RLS-FDE) weight $\mathbf{w}_j(k)$, is given by [10]:

$$\boldsymbol{\mathcal{R}}_j(k) = \sum_j \mathbf{w}_j(k)\, \boldsymbol{R}_j(k). \quad (4)$$

After RLS-FDE the central B-symbol block in the equalized N_B- symbol block is picked up to suppress the residual inter block interference (IBI) [9]. Next, the equalized symbol blocks $\boldsymbol{\mathcal{R}}_j(k)$ are transformed into time domain by using N_B −IFFT where the processed data blocks are combined to form the time domain signal $\boldsymbol{D}_j$ at point (o_{out}). The time domain signal $\boldsymbol{D}_j$ is now entered the time domain kernel recursive least square (KRLS) equalizer as $u(t)$ signal.
The output of a nonlinear adaptive equalizer is given by [11, 12]:

$$y(t) = f\big(u(t)\big). \quad (5)$$

In this work, $y(t)$ is the equalized time domain signal of Kernal adaptive equalizer used to reduce ISI. The signal $y(t)$ is transformed into a time-frequency signal using matched filter (Wigner transform) [6], given by:

$$Y[n,m] = H_{eq}[n,m]\, X[n,m] + V[n,m]. \quad (6)$$

Where $H_{eq}[n,m]$ is the equivalent response of the channel, RLS-FDE and KRLS equalizers in frequency domain.
Finally, the estimated signal is mapped back to the delay-Doppler domain using symplectic finite Fourier transform (SFFT), given by [6]:

$$\hat{x}[k,l] = \sum_{n=0}^{N-1}\sum_{m=0}^{M-1} y[n,m] e^{-j2\pi(\frac{nk}{N}-\frac{ml}{M})}. \quad (7)$$

3 Definition of a nonlinear adaptive filter

Figure 3 shows the basic structure of a nonlinear adaptive filter [11,12]. The input signal vector, $u(i)$, applied to the filter at time i , produces the actual response $y(i)$. This actual response is compared with an externally supplied desired response $d(i)$ to produce the error signal $e(i)$. This error signal is, in turn, used to produce an adjustment to the filter weights.

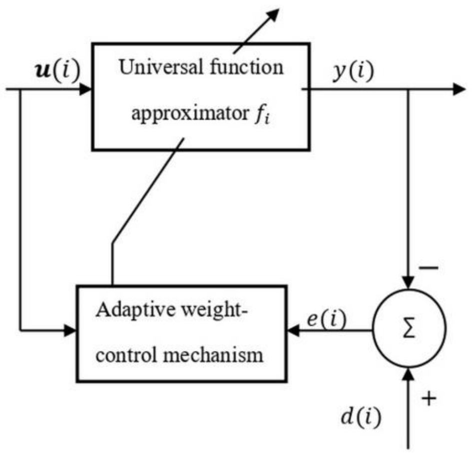

Fig. 3 Basic structure of a nonlinear adaptive filter

A nonlinear adaptive filter shown in Fig. 3 is built that possesses the ability of modelling any continuous input – output mapping $y = f\,(u(i))$ and obeys the following sequential learning rule [11]:

$$f_i = f_{i-1} + \boldsymbol{Gain}(i)e(i). \quad (8)$$

where f_i denotes the estimate input-output mapping at time i and $\boldsymbol{Gain}(i)$ is a function in general. The current estimate consists of two additive parts, namely, the previous estimate and a correction term proportional to the prediction error on new data.
Kernal adaptive algorithms transform the input data into a high - dimensional feature space via a reproducing kernel such that the inner product operation in the feature space can be computed efficiently through the kernel evaluations [11, 12]. Then, appropriate linear methods are subsequently applied on the transformed data. Because high-dimensional feature space is linear, kernel adaptive filters can be thought of as a generalization of linear adaptive filters. Kernel least mean square (KLMS) algorithm, Kernel affine projection (KAP) algorithm, kernel recursive least (KRLS) algorithm and modified kernel recursive least (KRLS) are presented below.

3.1 Kernel Least Mean Square (KLMS) algorithm

KLMS allocates a new kernel unit for the new training data with input $\boldsymbol{u}(i)$ as the center and $\eta\, e\,(i)$ as the coefficient. $\boldsymbol{a}(i)$ is the coefficient vector at iteration i , $\boldsymbol{a}_j(i)$ is its jth component, and $C(i)$ is the corresponding set of centers. At iteration i, given a test input point $\boldsymbol{u}_{tp}$, the output of the system is [12]:

$$f_{\boldsymbol{u}_{tp}} = \eta \sum_{j=1}^{i} e\,(j)\kappa\big(\boldsymbol{u}(j), \boldsymbol{u}_{tp}\big). \quad (9)$$

The step - size parameter is η, and the Kernel function $\kappa\,(\boldsymbol{u}, \boldsymbol{u}')$ used is the Gaussian kernel. The Gaussian kernel is used to quantify similarity between a pair of objects $\boldsymbol{u}$ and $\boldsymbol{u}'$ in $\mathbb{U}$ and the most important feature is that it reaches its maximum of one only when $\boldsymbol{u} = \boldsymbol{u}'$ and grows closer to zero as $\boldsymbol{u}$ and $\boldsymbol{u}'$ become more distant. The Gaussian kernel is given by [11,12]:

$$\kappa\,(\boldsymbol{u}, \boldsymbol{u}') = exp\left(\frac{-\|\boldsymbol{u}-\boldsymbol{u}'\|^2}{2{h_{KB}}^2}\right). \quad (10)$$

where h_{KB} is the kernel bandwidth.

3.1.1 KLMS algorithm

Initialization: choose step - size parameter η and kernel κ , let: $\boldsymbol{a}_1\,(1) = \eta\, d(1), C(1) = \{\, \boldsymbol{u}(1)\}$,
$f_1 = \boldsymbol{a}_1\,(1)\,\kappa\,(\,\boldsymbol{u}(1), .)$
while $\{\,\boldsymbol{u}\,(i), d\,(i)\}$ available
Compute the output:
$f_{i-1}(\boldsymbol{u}(i)) = \sum_{j=1}^{i-1} \boldsymbol{a}_j(i-1)\,\kappa\,(\boldsymbol{u}(i), \boldsymbol{u}(j)\,)$
Compute the error: $e\,(i) = d(i) - f_{i-1}(\boldsymbol{u}(i))$
Store the new center: $C(i) = \{\,C(i-1), \boldsymbol{u}(i)\,\}$
Compute and store the coefficient: $\boldsymbol{a}_i(i) = \eta\, e(i)$

end while

If f_i is denoted as the estimate of the input – output nonlinear mapping at time i, then the following sequential learning rule represents KLMS algorithm in summery and is summarized by [11, 12]:

$$f_{i-1} = \eta \sum_{j=1}^{i-1} e_j\ \kappa\ (\boldsymbol{u}(j), .). \tag{11}$$

$$f_{i-1}(\boldsymbol{u}(i)) = \eta \sum_{j=1}^{i-1} e_j\ \kappa\ \big(\boldsymbol{u}(j), \boldsymbol{u}(i)\big). \tag{12}$$

$$e\ (i) = d(i) - f_{i-1}\big(\boldsymbol{u}(i)\big). \tag{13}$$

$$f_i = f_{i-1} + \eta\ e(i)\ \kappa\ (\boldsymbol{u}(i), .). \tag{14}$$

3.2 Kernel Affine Projection (KAP) algorithm

Initialization: choose step - size parameter η, $\boldsymbol{a}_1(1) = \eta\ d(1)$.
while $\{\boldsymbol{u}\ (i), d\ (i)\}$ available
Allocate a new unit: $\boldsymbol{a}_i(i-1) = 0$.
for $k = \max(1,\ i - K + 1)$ to i
Evaluate outputs: $y(i;k) = \sum_{j=1}^{i-1} a_j(i-1)\ \kappa_{k,j}$
Compute the error: $e(i;k) = d(k) - y(i;k)$
update the $min(\ i\ , K\ )$ most recent units:
$\boldsymbol{a}_k(i) = \boldsymbol{a}_k(i-1) + \eta\ e(i;k)$
end for
if $i > K$ then
keep the remaining:
for $k = 1$ to $i - K$ do
$\boldsymbol{a}_k(i) = \boldsymbol{a}_k(i-1)$
end for
end if
end while

where $\kappa_{k,j} = \kappa(\boldsymbol{u}(i), \boldsymbol{u}(j))$. If f_i is considered as the estimate of the input - output mapping at time i , then the sequential learning rule for KAP is given by [12]:

$$f_i = f_{i-1} + \eta\ \sum_{j=i-K+1}^{i} e(i;j)\kappa(\boldsymbol{u}(j), .). \tag{15}$$

The updates needed for KAP at time i are [11]:

$$\boldsymbol{a}_i(i) = \eta\ e(i;i) \tag{16}$$

$$\boldsymbol{a}_i(i) = \boldsymbol{a}_j(i-1) + \eta\ e(i;j),\quad where\ \ j = i-K+1, \ldots, i-1 \tag{17}$$

$$\boldsymbol{a}_i(i) = \boldsymbol{a}_j(i-1), j = 1, \ldots, i-K \tag{18}$$

$$C(i) = \{C(i-1), \boldsymbol{u}(i)\} \tag{19}$$

At iteration i , given a test point input $\boldsymbol{u}_{tp}$, the system output is computed as [12]:

$$f_{\boldsymbol{u}_{tp}} = \sum_{j=1}^{i} \boldsymbol{a}_j\ (i)\kappa\big(\boldsymbol{u}(j), \boldsymbol{u}_{tp}\big). \tag{20}$$

3.3Traditional KRLS and Modified KRLS Algorithms

KRLS filter allocates a new kernel unit [12, 13, 14] for the new training data with input $\boldsymbol{u}(i)$ as the center and $(b_i)^{-1}e\ (i)$ as the coefficient. $\boldsymbol{a}(i)$ is the coefficient vector at iteration i , $\boldsymbol{a}_j(i)$ is its jth component, and $C(i)$ is the corresponding set of centers. $e\ (i)$ is the prediction (estimated) error between the desired signal $d(i)$ and the prediction $f_{i-1}(\boldsymbol{u}(i))$:

$$e\ (i) = d(i) - f_{i-1}\big(\boldsymbol{u}(i)\big). \tag{21}$$

At iteration i, with an input $\boldsymbol{u}$, the output of the system is [12, 13]:

$$f_i(\boldsymbol{u}) = \sum_{j=1}^{i} \boldsymbol{a}(i)\ \ \kappa(\boldsymbol{u}(j), \boldsymbol{u}). \tag{22}$$

The Kernel function $\kappa\ (\boldsymbol{u}, \boldsymbol{u}')$ used is the Gaussian kernel.
The traditional KRLS algorithm and its modification are listed below where the scalars are small italic letters. Vectors and matrices are respectively small and capital bold letters.

3.3.1 Algorithm 1 Traditional KRLS

Initialization:$\boldsymbol{RR}(1) = [\lambda + \kappa(\boldsymbol{u}(1), \boldsymbol{u}(1))]^{\boldsymbol{-1}}$,
$\boldsymbol{a}(1) = \boldsymbol{RR}(1)d(1)$. Choosing the regularization parameter λ.
If $i > 1$ then
$\boldsymbol{p}(i) = [\kappa(\boldsymbol{u}(i), \boldsymbol{u}(1)), \ldots., \kappa(\boldsymbol{u}(i), \boldsymbol{u}(i-1))]^T$
$\boldsymbol{q}(i) = \boldsymbol{RR}(i-1)\boldsymbol{p}(i)$
$b(i) = \lambda + \kappa(\boldsymbol{u}(i), \boldsymbol{u}(i)) - \boldsymbol{q}(i)^T\boldsymbol{p}(i)$

$$\boldsymbol{RR}(i) = b(i)^{-1}\begin{bmatrix} \boldsymbol{RR}(i-1)b(i) + \boldsymbol{q}(i)\boldsymbol{q}(i)^T & -\boldsymbol{q}(i) \\ -\boldsymbol{q}(i)^{-1} & 1 \end{bmatrix}$$

$e(i) = d(i) - \boldsymbol{p}(i)^T\boldsymbol{a}(i-1)$

$$\boldsymbol{a}(i) = \begin{bmatrix} \boldsymbol{a}(i-1) - \boldsymbol{q}(i)b(i)^{-1}e(i) \\ b(i)^{-1}e(i) \end{bmatrix}$$

end if

3.3.2 Algorithm 2 Modified KRLS

Initialization: $\lambda > 0$
$\boldsymbol{RR}(1) = [\lambda + \kappa(\boldsymbol{u}(1), \boldsymbol{u}(1))]^{\boldsymbol{-1}}$, $\boldsymbol{a}(1) = \boldsymbol{RR}(1)d(1)$.
for $i = 2$**:** nn
$\boldsymbol{p}(i) = [\kappa(\boldsymbol{u}(i), \boldsymbol{u}(1)), \ldots., \kappa(\boldsymbol{u}(i), \boldsymbol{u}(i-1))]^T$
$\boldsymbol{q}(i) = \boldsymbol{RR}(i-1)\boldsymbol{p}(i)$
if $b(i) = \lambda + \kappa(\boldsymbol{u}(i), \boldsymbol{u}(i)) - \boldsymbol{q}(i)^T\boldsymbol{p}(i)$ then

$$\boldsymbol{R}(i) = b(i)^{-1}\begin{bmatrix}\boldsymbol{RR}(i-1)b(i)+\boldsymbol{q}(i)\boldsymbol{q}(i)^T & -\boldsymbol{q}(i) \\ -\boldsymbol{q}(i)^{-1} & 1\end{bmatrix}$$

$e(i) = d(i) - \boldsymbol{p}(i)^T\boldsymbol{a}(i-1)$

$$\boldsymbol{a}(i) = \begin{bmatrix}\boldsymbol{a}(i-1) - \boldsymbol{q}(i)b(i)^{-1}e(i) \\ b(i)^{-1}e(i)\end{bmatrix}$$

if $b(i) << \lambda$ OR $b(i) \approx 0$ then

Discard $\boldsymbol{u}(i)$ and choose another data point

Choose $b(i+1) = \lambda + \kappa(\boldsymbol{u}(i+1), \boldsymbol{u}(i+1))$

$$\boldsymbol{RR}(i+1) = b(i+1)^{-1} \cdot \begin{bmatrix}\boldsymbol{RR}(i)b(i+1)+\boldsymbol{q}(i+1)\boldsymbol{q}(i+1)^T & -\boldsymbol{q}(i+1) \\ -\boldsymbol{q}(i+1)^{-1} & 1\end{bmatrix}$$

$e(i+1) = d(i+1) - \boldsymbol{p}(i+1)^T\boldsymbol{a}(i)$

$$\boldsymbol{a}(i+1) = \begin{bmatrix}\boldsymbol{a}(i) - \boldsymbol{q}(i+1)b(i+1)^{-1}e(i+1) \\ b(i+1)^{-1}e(i+1)\end{bmatrix}$$

end if

end if

end for

For both algorithms, if f_i is considered as the estimate of the input - output mapping at time i , then the sequential learning rule for KRLS is given by [12]:

$$f_i = f_{i-1} + r(i)^{-1}\left[\kappa(\boldsymbol{u}(i),.)) \sum_{j=1}^{i-1} \boldsymbol{z}_j\,(i)\kappa(\boldsymbol{u}(j),.)\right] \tag{23}$$

The updates needed for KRLS at time i are [12, 13]:

$$\boldsymbol{a}_i(i) = r(i)^{-1}e(i). \tag{24}$$

$$\boldsymbol{a}_j(i) = \boldsymbol{a}_j(i-1) - r(i)^{-1}e(i)\,\boldsymbol{z}_j(i), \quad j = 1, \dots, i-1 \tag{25}$$

$$C(i) = \{C(i-1), u(i)\} \tag{26}$$

Table 1 lists complexity comparison of KLMS, KAP, KRLS, and modified KRLS algorithms

Table 1 complexity comparison of KLMS, KAP, KRLS, and modified KRLS algorithms

Algorithm	**Complexity**	**Memory**
KLMS	$O(i)$	$O(i)$
KAP	$O(i+k^2)$	$O(i+k)$
KRLS	$O(i^2)$	$O(i^2)$
Modified KRLS	$O(i^2)$	$O(i^2)$

Although the complexity of KRLS, and modified KRLS are the same, the modified KRLS algorithm solves the problem of system singularity when $b(i)$ becomes too small (≈ 0), where the data point $\boldsymbol{u}(i)$ will be discarded and choosing another data point. Therefore, the modified KRLS algorithm offers a more stable implementation without increasing the overall computational order.

4. Results and Discussion

Fig. 4 shows the BER performance comparison of OTFS modulation with OSM using KLMS, KAP, KRLS and modified KRLS algorithms in Extended Vehicular A model (EVA) model [15] with maximum Doppler spread for each path generated according to a maximum user equipment (UE) speed of 120 km/h. The OTFS modulation grid with $M = 16$ sub-carriers, each of $\Delta f = 15$ KHz bandwidth, and $N = 8$ symbols. QPSK modulation is considered. The complexity of the system is more increased due to OSM by *O (N log N)* per block. The parameters used in simulations are a=1.2, $\eta = 0.05$, $\kappa\left(u_i, u_j\right) = exp\left(-0.1\left\|u_i - u_j\right\|^2\right)$, and λ = 1. Although KRLS is more complex compared to KAP and KLMS algorithms, its BER performance outperforms KAP and KLMS, respectively.

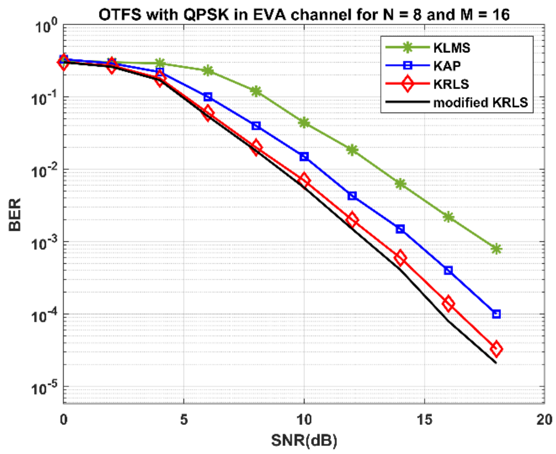

Fig.4 BER performance of the OTFS system using KLMS, KAP, KRLS, and modified KRLS equalizers with OSM in the EVA channel model

Fig.5 illustrates the efficacy of OTFS in a high mobility scenario, utilizing a Kernel Recursive Least Squares (KRLS) equalizer. The comparison is made between the performance with and without the overlap-save method (OSM) in an Extended Vehicular A (EVA) channel model [15]. The simulation parameters are tailored to a high-speed environment, with a Doppler spread calibrated for a user equipment (UE) speed of 120 km/h. The OTFS grid is defined as 16 sub-carriers and 8 symbols, each

$\Delta f = 15$ KHz, and employs Quadrature Phase Shift Keying (QPSK) modulation. The kernel function κ, which is a measure of similarity between two points in the feature space, is set to an exponential decay function based on the Euclidean distance, and the regularization parameter λ is set to 1. The results indicate a significant improvement in Bit Error Rate (BER) when the KRLS equalizer is paired with the OSM, attributing to the latter's ability to mitigate Inter Block Interference (IBI), while the equalizer effectively reduces Inter Symbol Interference (ISI), leading to a more robust communication in high-mobility environments.

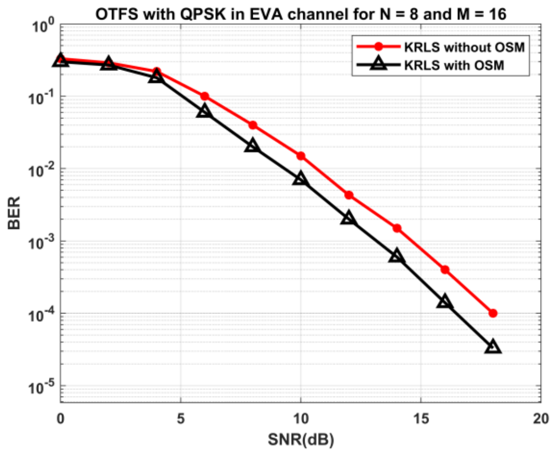

Fig.5 BER performance of the OTFS system using the KRLS equalizer without and with OSM in the EVA channel model.

With the same simulation parameters used in Fig.4, the BER performance of the OTFS schemes with overlap-save method for KRLS and modified KRLS algorithms are illustrated in Fig.6. The results indicate a better improvement in Bit Error Rate (BER) when modified KRLS equalizer is used because it solves the problems of system singularity and instability compared to the traditional KRLS equalizer.

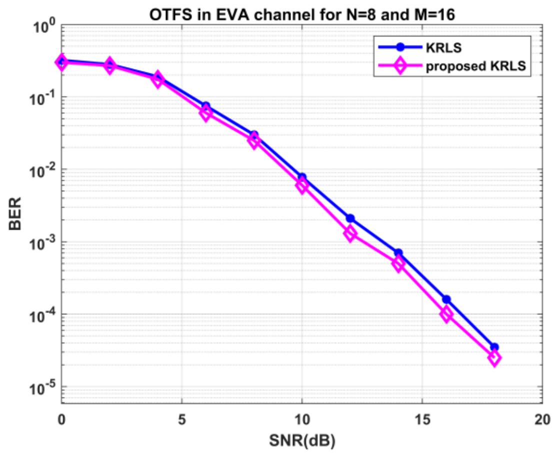

Fig.6. BER performance of the OTFS system using traditional KRLS and modified KRLS equalizers with OSM in the EVA channel model.

5. Conclusion

This paper investigated four nonlinear adaptive equalizers for OTFS modulation: KLMS, KAP, traditional KRLS, and modified KRLS. KLMS offers the lowest complexity but the weakest BER performance, whereas KAP provides improved performance at moderate additional cost. KRLS and modified KRLS achieve the best BER results, although they are computationally more demanding. The results demonstrate that combining a KRLS-based nonlinear time-domain equalizer with the overlap-save method is an effective approach for OTFS reception in EVA channels. In particular, the modified KRLS algorithm improves stability and alleviates singularity-related limitations of the traditional KRLS method, leading to better BER performance. Overall, the proposed OTFS receiver with modified KRLS and OSM provides a more robust solution for high-mobility wireless environments.

References:

[1] K. R. Murali and A. Chockalingam, ‘‘On OTFS modulation for high Doppler fading channels,’ in *Proc. Inf. Theory Appl. Workshop (ITA)*, San Diego, CA, USA, Feb. 2018, pp. 1–10.

[2] P. Raviteja, E. Viterbo, and Y. Hong,” OTFS Performance on Static Multipath Channels,” *IEEE Wireless Communications Letters*, vol. 8, no. 3, pp. 745-748, June 2019, doi:10.1109/LWC.2018.2890643.

[3] Z Wei et al., "Orthogonal Time-Frequency Space Modulation: A Promising Next-Generation Waveform," *IEEE Wireless Communications*, vol. 28, no. 4, pp. 136-144, Aug.2021, doi: 10.1109/MWC.001.2000408.

[4] W. Shin, K. Kim and Y. -J. Ko, "Performance Comparison of Equalization Schemes for OTFS over Time-Varying Multipath Channels," International Conference on Information and Communication Technology Convergence (ICTC), (Jeju, Korea (South), 2020), pp. 1433-1438, doi: 10.1109/ICTC49870.2020.9289461.

[5] P. Raviteja, Y. Hong, E. Viterbo, and E. Biglieri, "Effective diversity of OTFS modulation," *IEEE Wireless Communications Letters*, vol. 9, no. 2, pp. 249-253, Feb.2020, doi: 10.1109/LWC.20192951758.

[6] G. D. Surabhi and A. Chockalingam," Low-Complexity Linear Equalization for OTFS Modulation", *IEEE Communications Letters*, vol. 24, no. 2, pp. 330-334, Feb. 2020, doi: 10.1109/LCOMM.2019.2956709.

[7] S. S. Das, V. Rangamgari, S. Tiwari, and S. C. Mondal," Time Domain Channel Estimation and Equalization of CP-OTFS Under Multiple Fractional Dopplers and Residual Synchronization Errors" *IEEE Access*, vol.9, pp. 10561-10576, Dec. 2021, doi: 10.1109/ACCESS.2020.3046487.

[8] H. Li, and Q. Yu," Doubly-Iterative Sparsified MMSE Turbo Equalization for OTFS Modulation", *IEEE Trans. Communications*, vol. 71, no. 3, pp. 1336-135, March 2023, doi:10.1109/TCOMM.2023.3237243.

[9] H. TOMEBA, K. TAKEDA, and F. ADACHI," Overlap MMSE-frequency domain equalization for multi-carrier signal transmission", WPMC, San Diego, USA, Sept. 2006, pp. 17-20.

[10] Maha George Zia, " Adaptive Frequency Domain Affine Projection Equalizer of MIMO SC-FDMA System**,"** *IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)* Vol. 17, no. 5, Ser.II (Sep.-Oct. 2022), PP 33-39, doi: 10.9790/2834-1705023339.

[11] D. Comminiello, J. C. Principe, Adaptive Learning Methods for Nonlinear System Modeling, Butterworth-Heinemann, 2018.

[12] H. Zhao, B. Chen, Efficient Nonlinear Adaptive Filters: Design, Analysis and Applications, Springer Nature, 2023

[13] Y. Engel, S. Mannor and R. Meir, "The kernel recursive least-squares algorithm," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2275-2285, Aug. 2004, doi: 10.1109/TSP.2004.830985.

[14] X. Guo, S. Ou, M. Jiang, Y. Gao, J. Xu and Z. Cai, "A New Spares Kernel RLS Algorithm for Identification of Nonlinear Systems, " *IEEE Access*, vol. 9, pp. 163165-163177, Dec. 2021, doi: 10.1109/ACCESS.2021.31.33012.

[15] ETSI TS 136 116 V11.4.0 (2015-04) LTE; Evolved Universal Terrestrial Radio Access (E-UTRA), Relay radio transmission and reception (3GPP TS 36.116 version 11.4.0 Release 11).