Adaptive Kalman Filter Digital Pre-Distorter for Non-Linear Power Amplifiers

A Kalman Filter Digital Pre-distorter Algorithm

Jonathan Foley

Centre for Mathematical Sciences
Faculty of Science
Lund University
Sweden
June 16 2021

Abstract

The aim of the thesis is to develop an adaptive Kalman Filter algorithm for Digital Pre-Distortion (DPD) and to study its relevance as a Power Amplifier (PA) DPD method. The purpose of a DPD is to counter the non-linearity effects of a PA. The Kalman Filter is used as an optimizer for a General Memory Polynomial (GMP) model. With help from the Kalman Filter optimizer, the GMP model can numerically approximate the inverse of the non-linear behavior of a PA and apply it to the signal as Pre-Distortion before it passes through the PA, thereby counteracting the non-linearity effects of the PA. The Kalman Filter algorithm is compared with a Least Squares adaptive algorithm as a benchmark. The results show that the Kalman Filter algorithm has potential to be a low memory cost adaptive algorithm with more stability and better performance than the benchmark.

Acknowledgements

I would like to express my gratitude to all those who have helped me in the process of completing this thesis. Continuous feedback and discussion has helped me hugely in completing the thesis. I would like to give thanks to my supervisors at Ericsson Dimitar Nikolov and Sergio Gutiérrez who always expressed interest in the progress of the project and supported the work throughout. I would also like to give thanks to Ericsson for the opportunity of writing the thesis and specifically Stefano Olivotto for hiring me as a masters student for this project at Ericsson.

Contents

1 Introduction and Previous Works

The aim of this thesis is to develop an adaptive Kalman Filter (KF) algorithm for Digital Pre-Distortion (DPD) and to study its performance, versatility and relevance as a DPD method. In this thesis the KF algorithm is used to estimate and optimize the coefficients of a General Memory Polynomial (GMP) which is one way to model the non-linear behaviour of a Power Amplifier (PA). The choice of a GMP over other models was due to its ability to consider cross term dependencies of the signal together with its relatively high performance shown in previous work [6]. This thesis also looks at whether or not there are benefits in considering said cross term dependencies in conjunction with the KF algorithm. The GMP coefficients can be estimated and optimized in a few different ways, each way with its trade-offs between performance and complexity. This thesis includes a comparison with a less complex algorithm, an adaptive Least Squares (LS) algorithm. The LS algorithm is developed to have the same data intake per iteration and a similar learning rate as the KF algorithm in order to make the performance comparison as fair as possible.

The goal of DPD is to increase the functionality and effectiveness of a PA. A PA is a device in a communications system in charge of increasing the power of a signal before it is transmitted via an antenna. The higher the power of the signal, the farther the range of the antenna i.e. the information can be transmitted at larger distances. In order to attain higher signal power, the input power of the signal to the PA is raised, but when the input power is too high, the PA starts to exhibit non-linear behaviour. Non-linear behaviour leads to distortion and power leakage, which are intolerable consequences in accordance with regulatory laws. The DPD counteracts the non-linear behaviour, and allows for linear behaviour at higher input power and therefore higher output power, which in turn is a more energy efficient use of the PA.

Many DPD methods have been developed in recent years. These methods have grown in complexity and performance as hardware has improved. A neural network with two hidden layers, using a combination of linear and non-linear activation functions was developed in 2011 [5]. A 'Low Complexity Extended Kalman Filter' was developed in 2017 [10] which used an Extended Kalman Filter to train a neural network. The idea of using support vector machines for non-linearity compensation in PA's has been around a while [1] and more recently in 2019 a twin support vector machine approach showed promising results [11].

This thesis is structured so that the results and discussion are as intelligible as possible. Part 2 is about why PA's are used and what problems arise by using them. Section 2.1 gives an overview of the mathematical implications of a PA. It should become relatively apparent in section 2.1.1 why a PA is necessary, what the amplification of a signal entails, what a non-linear system is, and why it affects the signal. Then section 2.1.2 discusses how the non-linearity of a PA is countered by DPD. Section 2.2 contains an overview of some relevant DPD performance metrics, namely Adjacent Channel Leakage Ratio in section 2.2.1, Saturation Rate in 2.2.2, and Amplitude and Phase Modulation in section 2.2.3. Part 3 introduces what the General Memory Polynomial is and why it is relevant to use as part of DPD, it includes the mathematical definition of a the GMP and discussions of why it is defined the way it is and how it is optimized. The KF algorithm is introduced in part 4, the mathematical definition is thoroughly described along with some intuitive ways of interpreting the algorithm. Section 4.1 goes into detail about how the Kalman Filter is applied to optimize the coefficients of the GMP model. Part 5 contains a multitude of performance results of the KF algorithm as well as some discussion of the interpretation of the results. Results include testing different parameters for the GMP and testing different sample sizes per iteration of the KF algorithm. Analysis of the results and discussions of future research are given in part 6 as well as some general reflection on choices made during the creation of the KF algorithm. The PA model and data simulation is written in MATLAB [4] as are all the algorithms developed in this paper.

2 About the Power Amplifier

2.1 Radio Frequency Power Amplifiers

The Power Amplifier (PA) is a vital component in wireless communication systems. It amplifies signals so that they are more easily detectable by the receiver. In essence, a strengthened signal reduces information loss between sender and receiver. The instantaneous power of a signal is measured as the squared magnitude of the signal[1]

$$P_{x_t} = |x_t|^2 = |I_{x_t} + iQ_{x_t}|^2 = I_{x_t}^2 + Q_{x_t}^2 \ , \tag{1}$$

where x_t represents a sample value of the signal at instance t, and the signal is in I/Q form, a rectangular representation of polar coordinates.

A PA generally behaves quite well on its own for relatively low power input signals. Behaving well in the sense that increasing the input power linearly increases the output power of the PA. It is often the case for a modern PA that higher efficiency is achieved at a higher signal input power [8], a common metric for efficiency in this case is Power-Added Efficiency (PAE) which is expressed as

$$PAE = \frac{P_{out} - P_{in}}{P_{dc}} \ ,$$

where P_{dc} represents the electrical power required to run the PA. Since the PA is one of the components in a wireless communication system that consumes a relatively large amount of electrical power, it is crucial that it is as efficient as possible.
When the PA is well behaved; at its ideal performance, its function could be described as

$$g(x_t) = c_1 x_t \ , \tag{2}$$

where $c_1 > 1$ is some constant [2] and x_t again represents some sample value at instance t of the signal. In other words, the output of the PA is a scalar multiple of the input. This would then result in an increased power output in accordance with (1):

$$P_{g(x_t)} = |g(x_t)|^2 = c_1^2(I_{x_t}^2 + Q_{x_t}^2) \ .$$

Naturally, the solution of increasing efficiency then lays in increasing the input power of the signal. However, PA's reach an input power threshold - often referred to as a

[1] In this thesis, power related measurements will be represented in decibel-milliwatts (dBm) but the conversion will be omitted in equations.

[2] Representative of the Gain of the PA and therefore assumed to be larger than 1

compression point - shown in Figure 1 at which the relation between input and output power of the PA loses its linearity. Despite the non-linearity, the input domain succeeding the compression point is valuable real-estate because higher energy efficiency is still achieved with higher input power after this point. Hence, the objective becomes to counteract the side-effects of non-linearity past the compression point.

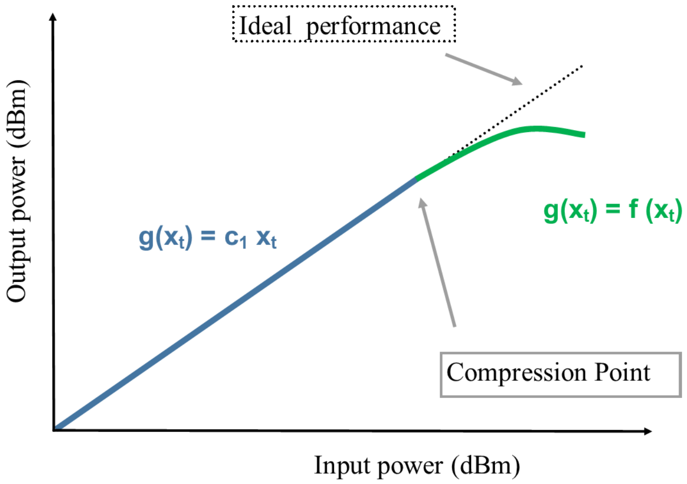

Figure 1: *Illustration of Compression point and the relationship between Input power and Output power of the a Power Amplifier.*

2.1.1 Non-Linearity Impediments

The PA's effect on the power of the signal becomes non-linear with input power greater than at the aforementioned compression point. This leads to intermodulation distortion and spectral regrowth. Spectral regrowth is essentially power leakage into the side-lobes of the power spectrum of the signal. This leakage may corrupt neighbouring signal channels, regulatory laws therefore limit how much power leakage is tolerated.

Past the compression point, it can no longer be assumed that the PA behaves as in eq. (2) but rather that

$$g(x_t) = \begin{cases} c_1 x_t, & for\ P_{in} \leq P_{cp} \\ f(x_t), & for\ P_{in} > P_{cp} \end{cases} \tag{3}$$

where P_{cp} is the compression point. In general $f(x_t)$ differs from PA to PA, and before trying to figure out what it is and how to counter its effect on the signal, it is important to know how it affects the signal.

The function $f(x_t)$ is unknown. It represents the behaviour of a PA after the compression point but this thesis is void of speculation as to its exact form. However, to illustrate how this unknown function might affect the signal, a simple polynomial can be used as an example. Let it be assumed that:

$$f(x_t) = c_1 x_t + c_2 x_t^2 + c_3 x_t^3 \ ,$$

where c_2 and c_3 are two coefficients such that $f(x_t)$ reasonably represents the non-linear effects of a PA[3]. Then, noting that the I/Q signal is a rectangular representation of polar coordinates, i.e. $I = A \cdot \cos \omega t$, $Q = A \cdot \sin \omega t$, the signal can be rewritten in Eulers form to

$$I + iQ = A(\cos \omega t + i \sin \omega t) = Ae^{i\omega t} \ . \tag{4}$$

It can then clearly be seen that when the signal is put through the PA, the output would show

$$g(x_t) = c_1 A e^{i\omega t} + c_2 A^2 e^{i(2\cdot\omega t)} + c_3 A^3 e^{i(3\cdot\omega t)} \ , \tag{5}$$

which contains two new angles $2 \cdot \omega$ and $3 \cdot \omega$ which are the antagonists of the PA that contribute to power leakage into neighbouring frequencies.

2.1.2 Counteracting Non-Linear Impediments with Digital Pre-Distortion

The general idea of Digital Pre-Distortion (DPD) is to influence the signal before it passes through the PA in order to achieve the desired - scalar multiple of input - output. In essence, it is an attempt to find the inverse of the PA function and apply it to the signal before it enters the PA. If an exact PA function existed and was known and if it was continuous and injective around the input sample value, then an inverse could be found and applied to the signal if the input power surpasses the compression point, i.e.

$$g[f^{-1}(x_t)] = x_t \ ,$$

though the gain (c_1) would be excluded from the inverse so as to maintain the effect of amplifying the signal, which is fine if:

[3] This type of polynomial is actually quite a good representation of a PA's non-linear behaviour given the correct values of c_2 and c_3

$$f = \frac{g}{c_1} \Rightarrow g[f^{-1}(x_t)] = c_1 x_t \ . \tag{6}$$

Though, as was established, a function g that truly represents the PA is not assumed to be known. It is clear nonetheless that eq. (6) shows the ideal outcome of a DPD, and so that is what a DPD aims to achieve. This is illustrated in Figure 2

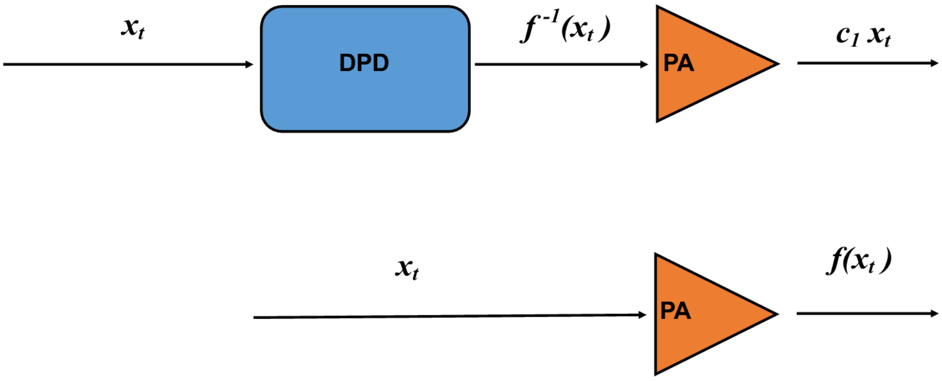

Figure 2: *Illustration of DPD counteracting the PA effects.*

In order to formulate a strategy for counteracting the non-linear impediments, it is helpful to first understand what assumptions can be made about the PA and to know what type of information is at hand. In general, it is quite reasonable to say that at the very least, some sequential segment of the input signal to the PA and the corresponding output signal from the PA is available[4]. Furthermore, the PA is a piece of hardware, and it is known to exhibit memory effects from thermal - electrical influences. It then follows that it is reasonable to find a numerical approximation of the inverse effect of the PA. To do so, a general function which has the ability to mimic the behaviour of the PA must be established. In this thesis, said function will take the form of a General Memory Polynomial (GMP) which will be further discussed in part 3.

[4] Depending on the sample size of the system or what memory is available

2.2 Performance Measures

2.2.1 Adjacent Channel Leakage Ratio

One of the main issues with non-linearity is power leakage into neighbouring frequencies. The Adjacent Channel Leakage Ratio (ACLR) is considered a good performance measure for whether or not power leakage has been reduced. If successful, DPD will cause the ACLR to decrease. The ACLR is defined as

$$ACLR = \frac{\int_{AdjC} |G(x_t)|^2 df}{\int_{MC} |G(x_t)|^2 df} \tag{7}$$

where $G(x_t)$ represents the Fourier transform of the PA output $g(x_t) \xrightarrow{\mathfrak{F}} G(x_t)$ and is normally rescaled to dBm. $AdjC$ and MC are the domains of integration, namely the adjacent channels and the main channel respectively. The squared magnitude of the Fourier transform of a time dependent signal is generally referred to as the Power Spectral Density (PSD) of a signal, which shows the distribution of power over frequencies. In essence, the ACLR compares the band-limited spectrum of the adjacent channels to the main channel, and if the power leakage is reduced, the ACLR will be proportionally reduced. An illustration of ACLR on a signals PSD can be seen in Figure 3.

2.2.2 Rate of Saturation

Adaptive algorithms have the capability to change parameter estimations given new behaviour in the information received. A more detailed description can be found in part 4, but importantly, when an algorithm is adaptive, an essential performance metric is the rate at which the algorithm can adapt, i.e. its saturation rate. There are a multitude of things that will affect the learning rate of the an algorithm. Most significantly, if the model used to emulate the PA is underspecified or incapable of representing the actual behaviour of the PA, the algorithm will simply keep changing its estimated parameters with each update, since the model never quite fits the data. For instance, if the model proposed to emulate the PA is a linear function, the algorithm would not be able to find a stable coefficient parameter if the input power is higher than the compression point threshold, since the behaviour of the PA after this point is non-linear.

The rate of saturation of the algorithm is seen as Mean-Squared Error (MSE) per signal iteration. The MSE is calculated as

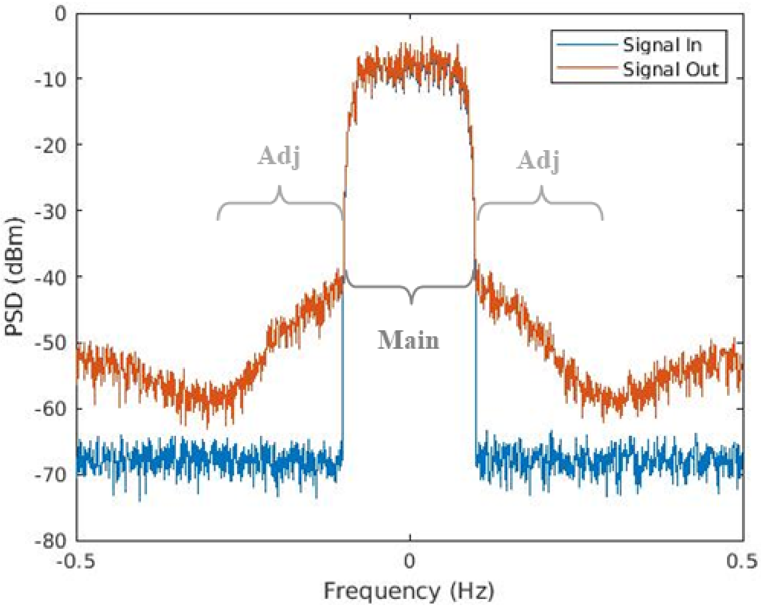

Figure 3: *Power Spectral Densities of a simulated input signal and the same signal when put through a simulated Power Amplifier. Also highlighting the main and adjacent channels.*

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(x_i - \frac{g[f^{-1}(x_i)]}{c_1}\right)^2 , \qquad (8)$$

where $g[f^{-1}(x_i)]$ represents the i^{th} data sample that has been digitally pre-distorted and passed through the PA[5].

The MSE is not only an important metric when measuring the rate of saturation, but the final MSE also determines how dissimilar the pre distorted signal is to the true signal. If the MSE is too high, the information in the amplified signal would have lost information.

2.2.3 Amplitude and Phase Modulation

Input power and its relation to output power is a characteristic of the PA which perhaps most obviously illustrates non-linearity, however, non-linearity affects more than just the power of the signal. In almost all cases, PA's will be driven by modu-

[5] The MSE will also be represented in dBm for consistency in units.

lated[6] signals, a better understanding of the PA behaviour can be found by studying the phase modulation and amplitude modulation of the output signal with varying input power. Generally, two more characteristics [3] are used to describe the PA[7], namely amplitude modulation to amplitude modulation (AM/AM) and amplitude modulation to phase modulation (AM/PM). The AM/AM characteristic can be seen by looking at the magnitude of the instantaneous gain against the input power as shown in Figure 4. The magnitude of the instantaneous gain is defined as

$$|Gain_{inst}| = \frac{|x_{out}|^2}{|x_{in}|^2} = \frac{I_{out}^2 + Q_{out}^2}{I_{in}^2 + Q_{in}^2} \ , \tag{9}$$

where $x_{out} = I_{out} + iQ_{out}$, $x_{in} = I_{in} + iQ_{in}$ and will generally be represented in dBm.

The AM/PM characteristic can be seen by looking at the phase of the instantaneous gain against the input power, seen in Figure 5. The phase can be derived by recalling Eulers form from eq. (4)

$$I = Acos\theta \ , \ Q = Asin\theta \ \Rightarrow \frac{Q}{I} = \frac{sin\theta}{cos\theta} = tan\theta \Rightarrow \theta = tan^{-1}\left(\frac{Q}{I}\right) \ ,$$

then the phase of the instantaneous gain is found as

$$\underline{/Gain_{inst}} = \underline{/x_{out}} - \underline{/x_{in}} = tan^{-1}\left(\frac{Q_{out}}{I_{out}}\right) - tan^{-1}\left(\frac{Q_{in}}{I_{in}}\right) \ . \tag{10}$$

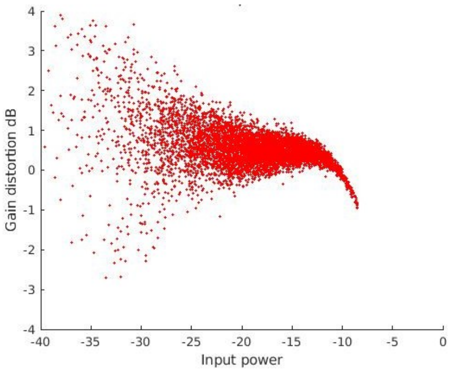

Figure 4: *AM/AM characteristic of the PA*

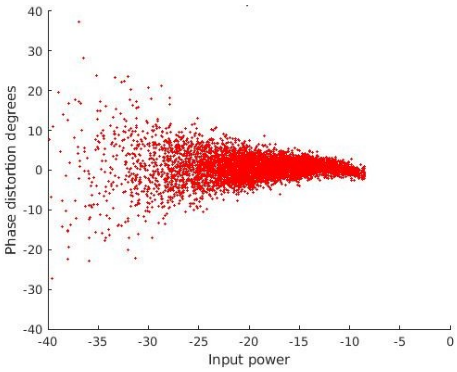

Figure 5: *AM/PM characteristic of the PA*

[6] Modulation is the process of varying the properties of a carrier signal. Benefits include increasing range of communication and multiplexing, among others.

[7] There are PM/PM and PM/AM characteristics as well but since the PA's intent is to affect the amplitude of a signal, these characteristics do not provide much information.

The dispersion of the AM/AM and AM/PM plots in Figures 4 and 5 indicate how severe the memory effects of the PA are. Had there been no memory effects, there wouldn't have been differences in distortions at the same input power. It can be seen in Figure 4 for instance, that at -30 dBm input power, the gain distortion is anywhere between $+4$ dBm and -3 dBm, so the distortion of the samples depend on their position in the signal sequence i.e. they depend on memory. If a DPD performs well, the dispersion should be reduced, and distortions should decrease.

3 Memory Polynomials

A Memory Polynomial (MP) model can be convenient to use when a system input has some dependence on previous inputs, as implied in the name, it considers 'memory'. The MP model is capable of modelling non-linear behaviour, it splits the input into weighted summands of inputs at different lags with the option of summands having dependence on the amplitude[8] to the k^{th} power at said lag. The MP model is described as

$$y_{MP}(n) = \sum_{k \in K_a} \sum_{l \in L_a} a_{kl} x(n-l) |x(n-l)|^k \ , \tag{11}$$

where K_a and L_a are sets of polynomial order and memory respectively more about these sets is discussed in the following section.

The coefficients a_{kl} are to be found as a solution to a set of linear equations such that the model reflects the desired behaviour. It is important here to have an idea of where dependencies of the system lay, it is easy to have an over-fitted model with too much memory. It is a sensible assumption in the case of a PA system that much of the memory is inherited from nearby lags and then tapers off since the physical (thermal) memory of a PA is likely monotone and gradual. This entails that, likely, much of the behaviour of the PA will be contingent on nearby lags.

3.1 General Memory Polynomial

The General Memory Polynomial (GMP) model is essentially an extension of the MP, it differs by including cross terms which allows for considering dependencies on amplitudes of leading and lagging input terms. The output of a GMP model is described by its input as

$$\begin{aligned} y_{GMP}(n) = &\sum_{k \in K_a} \sum_{l \in L_a} a_{kl} x(n-l) |x(n-l)|^k + \\ &\sum_{k \in K_b} \sum_{l \in L_b} \sum_{m \in M_b} b_{klm} x(n-l) |x(n-l-m)|^k + \\ &\sum_{k \in K_c} \sum_{l \in L_c} \sum_{m \in M_c} c_{klm} x(n-l) |x(n-l+m)|^k \ , \end{aligned} \tag{12}$$

[8] In this case amplitude, but the MP is applicable when inputs do not represent signals or waves

where K_b and L_b should be seen as sets of cross term polynomial order and memory at lag up to M_b, respectively. Similarly K_c and L_c should be seen as sets of cross term polynomial order and memory at forward lag up to M_c. a_{kl} , b_{klm} and c_{klm} are coefficients to be found as a solution to the system and will be represented in a coefficient vector, $\boldsymbol{\alpha}$, whose elements are compiled by the union of the following sets.

$$\begin{aligned} A = \{a_{k,l} \mid k \in K_a,\ l \in L_a\} &\cup \{b_{k,l,m} \mid k \in K_b,\ l \in L_b,\ m \in M_b\} \\ &\cup \{c_{k,l,m} \mid k \in K_c,\ l \in L_c,\ m \in M_c\}\ , \end{aligned} \tag{13}$$

where $K_a \cup L_a \cup L_b \cup L_c \in \mathbb{N}_0$, $K_b \cup K_c \cup M_b \cup M_c \in \mathbb{N}$. Then the elements of A are brought into the complex vector space[9] $\mathbb{C}^{|A|\times 1}$ and compiled in the vector $\boldsymbol{\alpha}$. The GMP output will, for the sake of compactness be represented in this thesis as

$$y_{GMP}(n) = x_{k,l,m}^T(n)\boldsymbol{\alpha}\ , \tag{14}$$

where $x_{k,l,m}(n) \in \mathbb{C}^{|A|\times 1}$ is a vector of the unique summands in (12). The following is an example of what the model looks like when $K_a = \{0, 1, 2\}, K_b = K_c = \{1, 2\}$, $L_a = \{0, 1\}$, $L_b = L_c = \{0\}$, $M_b = M_c = \{1\}$

$$\begin{aligned} y_{GMP}(n) = a_{0,0}x(n) + a_{1,0}x(n)|x(n)| + a_{2,0}x(n)|x(n)|^2 + a_{0,1}x(n-1) + \\ a_{1,1}x(n-1)|x(n-1)| + a_{2,1}x(n-1)|x(n-1)|^2 + b_{1,0,1}x(n)|x(n-1)| + \\ b_{2,0,1}x(n)|x(n-1)|^2 + c_{1,0,1}x(n)|x(n+1)| + c_{2,0,1}x(n)|x(n+1)|^2\ . \end{aligned} \tag{15}$$

It should be noted that the summations are over sets and not lists. Often the partial sums of the GMP are defined as sums from $1 : L_b$ for instance, which would include each integer from 1 to L_b. Including each of these terms is not necessarily beneficial, since more summands contribute to complexity and summands may or may not contribute very much to minimizing the error of the model [6]. Hence, the algorithm allows choosing partial sums such as perhaps only odd lags $L_b = [1, 3, 5]$ which may turn out to be more significant summands. Examples of this are shown in part 5. If there are enough summands that contribute considerably to the accuracy of the model, suitable coefficients need to be found so that the model represent the PAs behaviour well enough to use its inverse as a pre-distortion method.

[9] Complex vector space due to the nature of the problem at hand

3.2 Least Squares as an Optimizer

As far as optimizers go, Least Squares Method (LS) is a relatively simple and often reliable method to use. It finds the coefficients $\boldsymbol{\alpha}$ as a solution to a linear system minimizing the square error. Here, to illustrate how the LS method works, let X be some arbitrary input with weights $\boldsymbol{\alpha}$ with the target output Y, the aim is to minimize the square error:

$$\begin{aligned} ||X\boldsymbol{\alpha} - Y||^2 &= (X\boldsymbol{\alpha} - Y)^T (X\boldsymbol{\alpha} - Y) \\ &= Y^TY - Y^TX\boldsymbol{\alpha} - \boldsymbol{\alpha}^TX^TY + \boldsymbol{\alpha}^TX^TX\boldsymbol{\alpha} \; . \end{aligned}$$

The next step is to take the partial derivative w.r.t $\boldsymbol{\alpha}$ and set it to zero

$$\begin{aligned} &\frac{\partial}{\partial \boldsymbol{\alpha}} \left(Y^TY - Y^TX\boldsymbol{\alpha} - \boldsymbol{\alpha}^TX^TY + \boldsymbol{\alpha}^TX^TX\boldsymbol{\alpha} \right) = 0 \\ &\Rightarrow -2X^TY + 2X^TX\boldsymbol{\alpha} = 0 \; , \end{aligned}$$

since $Y^TX\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}^TX^TY$ are an equivalent scalar. Then solving for $\boldsymbol{\alpha}$

$$2X^TX\boldsymbol{\alpha} = -2X^TY \Rightarrow \boldsymbol{\alpha} = (X^TX)^{-1}X^TY \; . \tag{16}$$

Eq. (16) is the solution for least squares. However, when finding the inverse of the PA function, the coefficients should be found by mapping the outputs to the inputs via the GMP model rather than inputs to outputs. In order to symbolize that, X is replaced with Y and Y with X, an appropriate LS solution for the inverse PA function is then:

$$\boldsymbol{\alpha} = (Y^TY)^{-1}Y^TX \; . \tag{17}$$

A somewhat adaptive LS method algorithm is used in this paper to compare performance with the KF algorithm. In order to do so with as low bias as possible, the algorithm was designed such that it would take in the same amount of information per iteration as the KF algorithm, and share similar stability and saturation rate. The $\boldsymbol{\alpha}$ is then adaptively updated with new information depending on the size of the error of the previous estimate:

$$\boldsymbol{\alpha}_t = \boldsymbol{\alpha}_{t-1} + \lambda(Y^TY)^{-1}Y^TX\boldsymbol{e}_{t-1} \; , \tag{18}$$

where $\boldsymbol{e} = x(N) - y^T_{k,l,m}(N)\boldsymbol{\alpha}_{t-1}$ and λ is a user-determined parameter often referred to as learning rate.

4 Kalman Filter

The Kalman Filter (KF) is a state estimation algorithm widely used in dynamic systems where state changes are sequentially dependent. It is an algorithm which combines two correlated but separate noisy sequences to estimate a hidden state sequence. It has the potential to be competitive as an optimizer for the GMP since it is an adaptive algorithm that does not require relatively[10] much physical memory. It does not need to store any information about past iteration estimates, but as the covariance matrix (introduced later in this section) is updated, important information about the dependencies between coefficients is considered in each iteration of the algorithm.

In order to estimate the hidden state sequence, the standard KF combines a linear state transition model and an observation model, they are generally described in the following way:

$$\begin{aligned} x_t &= A_t x_{t-1} + B_t u_t + w_t \\ y_t &= C_t x_t + v_t \ , \end{aligned} \tag{19}$$

where:

- x_t is some state vector, which are inputs to a system whose output is y_t
- A_t is a matrix describing the natural development of the state from time $t-1$ to t in the specified system
- B_t is a matrix describing the behaviour of the controlled input u_t to the system
- w_t is the process noise , v_t is the observation noise
- C_t is a mapping from input to output of the system

As can be seen in eq. (19), x_t is prevalent in both the state transition and the observation model, and both models contain uncertainty in w_t and v_t. The KF combines the two models to make an estimate of the true state x_t with reduced uncertainty[11].

The KF algorithm is often described as having two steps, a prediction step and an update step. The prediction step uses the state transition model to predict the

[10] This will of course depend on the amount of summands in the GMP model

[11] Assuming that the algorithm converges, i.e. the observation and predicted observation difference decreases

state x_t as well as the covariance matrix Σ_t (representing the preciseness of the state estimation) in the following way:

$$\begin{aligned} \hat{x}_t &= A_t x_{t-1} + B_t u_t \\ \hat{\Sigma}_t &= A_t \Sigma_{t-1} A_t^T + R_t \ , \end{aligned} \tag{20}$$

where $R_t = E[w_t w_t^T]$ is a covariance matrix of the process noise. $\hat{x}_t$ and $\hat{\Sigma}$ are often referred to as the *a priori* state and covariance estimate. The update step is when the algorithm combines the information from the prediction and the observation. It contains the following calculations,

$$\begin{aligned} K_t &= \frac{\hat{\Sigma}_t C_t^T}{C_t \hat{\Sigma}_t C_t^T + Q_t} \\ x_t &= \hat{x}_t + K_t(y_t - C_t \hat{x}_t) \\ \Sigma_t &= (I - K_t C_t)\hat{\Sigma}_t \ , \end{aligned} \tag{21}$$

where K_t is the Kalman Gain, $Q_t = E[v_t v_t^T]$ is the observation covariance matrix, x_t is the *a posteriori* state estimate and Σ_t is the a posteriori covariance of the state estimate. The outputs of the KF x_t and Σ_t are the predictions updated with information from the observation y_t. The derivations of K_t and the covariance matrix Σ_t update step are omitted in this thesis as they are quite lengthy and do not necessarily contribute to the understanding of the algorithm. For the intents and purposes of this thesis, with regards to their derivation, it suffices to say that K_t is derived as a minimizer of the estimate covariance.

Taking a closer look at K_t a helpful interpretation of how the observation and prediction are combined can be found. K_t is in some sense a trust-weighting of the two contributors of information (the prediction and observation) about the true state, where the contributor with least variance gets most trust as shown in Figure 6. To illustrate this, one can look at the extreme cases for K_t. If

$$\hat{\Sigma}_t \neq 0 \ , \ Q_t = 0 \rightarrow K_t = \frac{\hat{\Sigma}_t C_t^T}{C_t \hat{\Sigma}_t C_t^T + 0} = \frac{1}{C_t} \rightarrow x_t = \frac{y_t}{C_t} \ ,$$

meaning that the KF algorithm estimate is solely estimated with information from the observation if the observation covariance matrix is equivalently zero. Whereas if

$$\hat{\Sigma}_t = 0 \ , \ Q_t \neq 0 \rightarrow K_t = \frac{0}{0 + Q_t} = 0 \rightarrow x_t = \hat{x}_t \ ,$$

meaning that the KF algorithm estimate is instead solely estimated with information from the predicted state when the state prediction covariance is zero. When the KF is applied to a reasonable system, K_t in a one dimensional problem is such that $K_t \in [0, 1]$ and similarly in the multivariate case, every element in the K_t matrix is non-negative and less than or equal to 1.
Recalling the observation model in eq. (19), the *a posteriori* state estimation can be rewritten as

$$x_t = \hat{x}_t + K_t(y_t - \hat{y}_t) \ , \tag{22}$$

since C_t is a mapping from the state space to the observation space, $C_t\hat{x}_t$ would map a predicted state to a predicted observation. Eq. (22) shows that if the difference between the observation and the predicted observation becomes smaller, then the update to the predicted state $\hat{x}_t$ becomes smaller as well.

To give a simple example of a single iteration estimation, consider the location of a moving object: If an object is moving in a direction with a certain velocity, a prediction for the next observation would be the direction of the object, multiplied by the velocity and time until the next observation. However, unbeknownst to the predictor, the object may have changed direction or velocity. The observation of the location of the object would then likely be different than what was predicted. Bearing in mind that both observation and prediction contain uncertainty, the Kalman Filter would estimate the true location to be at the intersection of probable locations between the predicted and observed location giving a more likely estimate of the location.

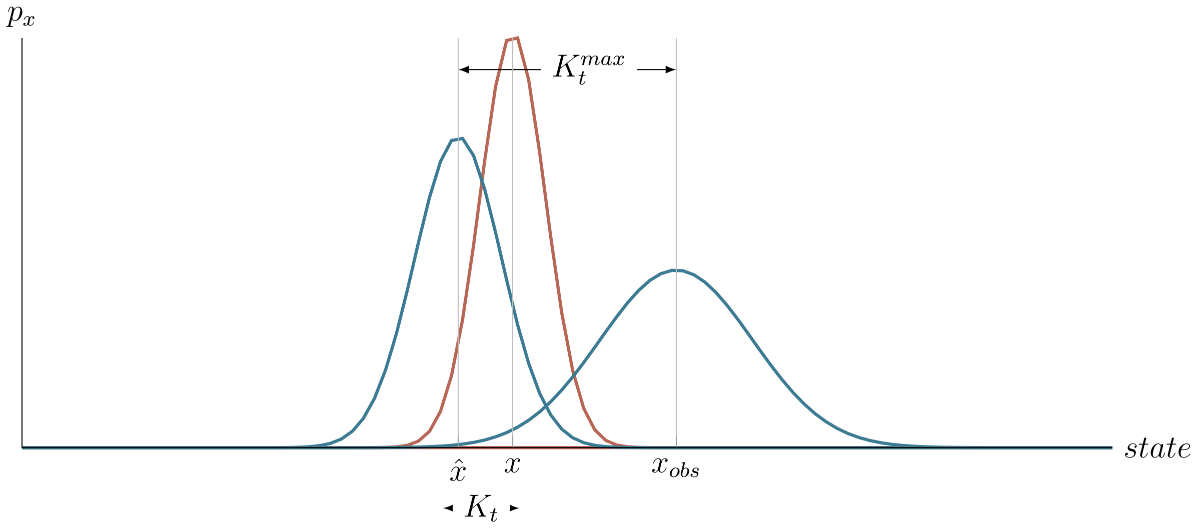

Figure 6: *One-dimensional illustration of joint distribution.*

Figure 6 is a pedagogical illustration of how two Gaussian distributed variables create a joint Gaussian distribution with lower variance. The standard Kalman filter considers the case where both the process and observation noise is Gaussian, which entails that the joint distribution of the two random variables is also Gaussian.

4.1 Applying the Kalman Filter

It may not be entirely obvious how to optimize the GMP model with the KF, this sections goes into detail about how to apply the KF in the GMP optimization. The hidden states to be estimated by the KF are the coefficients $\boldsymbol{\alpha}$ of the GMP, then the states transition model can be defined as

$$\boldsymbol{\alpha}_t = A_t \boldsymbol{\alpha}_{t-1} + B_t u_t + w_t \ , \tag{23}$$

where $\boldsymbol{\alpha}_t$ is a vector of the coefficients of the memory polynomials as in eq. (14), $A_t = I$ since the optimal coefficients are assumed to stay constant[12], $B_t = 0$, $u_t = 0$ since there is no controlled input altering the development of the coefficients, and $w_t = 0$ since there is no state transition process that noise can be attributed to. So in this case the state transition model is

$$\boldsymbol{\alpha}_t = \boldsymbol{\alpha}_{t-1} \ , \tag{24}$$

[12] For one particular PA

Now to consider the observation model. The mapping is explained in the GMP model itself, however, the way to go about finding an inverse to the PA function, is to do an inverse mapping. As in section 3.2, the GMP is used to map the outputs to the input in order to find the coefficients which correctly depict the behaviour of the inverse of the PA. Once these coefficients have been found, the GMP should be applied to the input before entering the PA, such that after having gone through the PA, the signal remains the same. Recalling eq. (12) but substituting n with t for continuity of notation and physical representation of signal values being a time dependent sequence:

$$\begin{aligned} y_t = &\sum_{k\in K_a}\sum_{l\in L_a} a_{kl}x_{t-l}|x_{t-l}|^k + \\ &\sum_{k\in K_b}\sum_{l\in L_b}\sum_{m\in M_b} b_{klm}x_{t-l}|x_{t-l-m}|^k + \\ &\sum_{k\in K_c}\sum_{l\in L_c}\sum_{m\in M_c} c_{klm}x_{t-l}|x_{t-l+m}|^k \ , \end{aligned}$$

which can be expressed as in eq. (14), while interchanging x for y to signify the inverse

$$x_t = y_{k,l,m,t}^T \boldsymbol{\alpha}_t + v_t \ , \tag{25}$$

where v_t is the observation noise and $\boldsymbol{\alpha}$ is the aforementioned coefficient vector. The prediction and update steps can then be defined for this system as

$$\begin{aligned} \hat{\boldsymbol{\alpha}}_t &= \boldsymbol{\alpha}_{t-1} \quad \in \mathbb{C}^{|A|\times 1} \\ \hat{\Sigma}_t &= \Sigma_{t-1} \quad \in \mathbb{C}^{|A|\times |A|} \ , \end{aligned} \tag{26}$$

where Σ_0 is initiated at some reasonable value for the system and updated in the update step which in this system is defined as

$$\begin{aligned} K_t &= \frac{\hat{\Sigma}_t y_{k,l,m,t}}{y_{k,l,m,t}^T \hat{\Sigma}_t y_{k,l,m,t} + Q_t} \quad \in \mathbb{C}^{|A|\times q} \\ \boldsymbol{\alpha}_t &= \hat{\boldsymbol{\alpha}}_t + K_t(x_t - y_{k,l,m,t}^T \hat{\boldsymbol{\alpha}}_t) \quad \in \mathbb{C}^{|A|\times 1} \\ \Sigma_t &= (I_{|A|} - K_t y_{k,l,m,t}^T)\hat{\Sigma}_t \quad \in \mathbb{C}^{|A|\times |A|} \ , \end{aligned} \tag{27}$$

where q is the number of samples per iteration and $y_{k,l,m,t} \in \mathbb{C}^{|A|\times q}$ is the mapping from output to input or C_t^{-1} from eq. (19).

Complex Covariance

When using a Kalman Filter on a complex system it should be mentioned that the covariance matrix Σ_t does not entirely represent the second order statistics of the complex variables in $\boldsymbol{\alpha}$ [7]. When considering complex random variables, there is a second order statistic called pseudo covariance which must be considered as well as the usual covariance, in order to fully characterize the second order statistic of the complex variable.

The type of Kalman Filter which does not include pseudo covariance, is referred to as a Conventional Complex Kalman Filter (CCKF) [9] and is what is used in this thesis. Though there are methods for including the pseudo covariance, in this thesis it has been sufficient not to. This is perhaps due to the nature of the problem. It is quite reasonable to assume - since the PA's main function is to affect the amplitude of the signal, as opposed to the phase - that the summands of the GMP contain enough variations of phase to represent the inverse of the PA as a linear combination of complex summands with real valued coefficients.

4.2 Kalman Filter Algorithm Step-by-Step

Before discussing the results, this section clarifies step-by-step how one iteration of the KF algorithm works. The assumption here is that some section of the signal is ready to enter the PA. The length of the section of the signal will be denoted by q. So an input signal of length q samples will be annotated as x_t^q where t represents the t^{th} sample of length q. Then the output of the PA is correspondingly denoted y_t^q, i.e. using the function g as in eq. (3)

$$g(x_t^q) = y_t^q \ . \tag{28}$$

As mentioned in the previous section, the GMP model is at this point applied to the output so that it can be mapped to the input as a linear combination of the GMP[13] summands as in eq. 29.

[13] The parameters of the GMP K_a, L_a, K_b, L_b, M_b, K_c, L_c, M_c are pre-determined by the user

$$\begin{aligned}x_t^q \approx &\sum_{k\in K_a}\sum_{l\in L_a} a_{kl} y_{t-l}^q |y_{t-l}^q|^k + \\ &\sum_{k\in K_b}\sum_{l\in L_b}\sum_{m\in M_b} b_{klm} y_{t-l}^q |y_{t-l-m}^q|^k + \\ &\sum_{k\in K_c}\sum_{l\in L_c}\sum_{m\in M_c} c_{klm} y_{t-l}^q |y_{t-l+m}^q|^k \ ,\end{aligned} \tag{29}$$

where the coefficients $a_{kl},\ b_{klm},\ c_k lm\ \in \boldsymbol{\alpha}$ are either random numbers in $(0,1]$ if $t=0$ at initialization, or if $t>0$, $\boldsymbol{\alpha}$ is the previous estimation of the algorithm. The error of this mapping is then found as

$$\begin{aligned}x_t^q - &\sum_{k\in K_a}\sum_{l\in L_a} a_{kl} y_{t-l}^q |y_{t-l}^q|^k + \\ \sum_{k\in K_b}\sum_{l\in L_b}\sum_{m\in M_b} &b_{klm} y_{t-l}^q |y_{t-l-m}^q|^k + \\ \sum_{k\in K_c}\sum_{l\in L_c}\sum_{m\in M_c} &c_{klm} y_{t-l}^q |y_{t-l+m}^q|^k = e_t^q \ .\end{aligned}$$

The Kalman Gain K_t is at this point also calculated, in order to establish the aforementioned 'trust weighting' of prediction vs. observation. It is found as

$$K_t^q = \frac{\hat{\Sigma}_t y_{k,l,m,t}^q}{(y_{k,l,m,t}^q)^T \hat{\Sigma}_t y_{k,l,m,t}^q + Q_t} \ , \tag{30}$$

where if the covariance matrix of the prediction $\hat{\Sigma}_t$ is low relative to the observation covariance matrix Q_t then the predicted state estimates $\hat{\boldsymbol{\alpha}}$ will be trusted more, and be updated with smaller changes in the next step of the algorithm, namely the update step of the coefficients $\hat{\boldsymbol{\alpha}}$

$$\boldsymbol{\alpha}_t = \hat{\boldsymbol{\alpha}}_t + K_t^q(e_t^q) \ . \tag{31}$$

Then the covariance matrix $\hat{\Sigma}_t$ is updated, again in accordance with the Kalman Gain

$$\Sigma_t = (I_{|A|} - K_t^q (y_{k,l,m,t}^q)^T)\hat{\Sigma}_t \ . \tag{32}$$

Equations (31) and (32) are now the updated coefficients and covariance respectively and are used as predictors in the next iteration of the algorithm as shown in eq. (26)

The goal of the algorithm is to reduce the error e_t^q and the covariance matrix Σ_t^q. If either of the two are large, the KF algorithm will update the estimated coefficients in correspondingly large increments, and vice versa. With good coefficient estimates, the updates will be smaller and converge to optimal coefficients.

When coefficients for the GMP that maps the output of the PA to the input are found, the GMP now represents the inverse function of the PA, which then is applied to the input signal before it enters the PA as shown in eq. (6). Figure 7 is an illustration of the flow of a signal section x_t^q through the DPD and how the input and output is processed by the KF algorithm in order to update the GMP coefficients in the DPD.

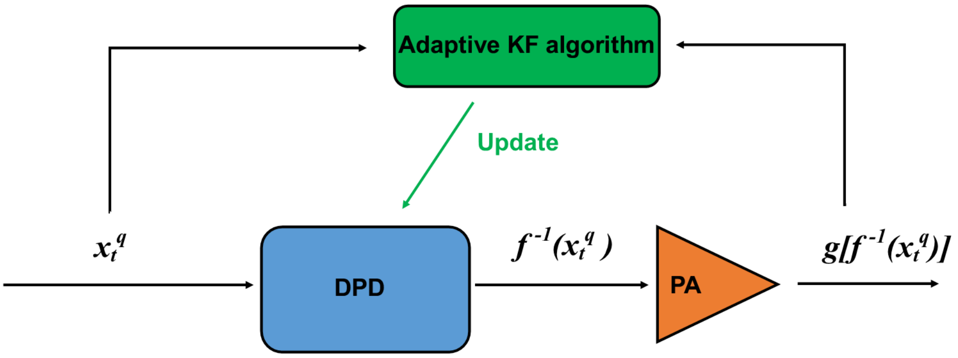

Figure 7: *Illustration of the flow of the PA and DPD part of a communications system with an adaptive step.*

5 Results

All of the signal data used in this thesis follows the Wideband Code Division Multiple Access (WCDMA) modulation scheme and is simulated and provided by Ericsson AB. A simulated PA is also used in this thesis. The PA model was developed in previous work by [2] and is based on a Mini-circuits System In Package (MSiP) surface mount IC (model no: YSF-322+) at $-2dBm$ input power, made of GaAs with E-PHEMT.

For the table of results in Table 1 the initial coefficients of the GMP model $\boldsymbol{\alpha}_0$ were sampled from a uniform distribution in $[0, 1]$. Σ_0 was selected as the identity matrix scaled down by a factor 10: $I \cdot 10^{-1}$, and the observation covariance matrix Q_t was chosen to be the same: $I \cdot 10^{-1}$. The sample section length q was 16. There are definitively options when choosing the initial values. In particular, changing Σ_0 will affect convergence rate but also stability, it is chosen here to have good stability such that the algorithm consistently converges to optimal coefficients. When Σ_0 was chosen to be larger, the results would converge faster but not necessarily be stable enough to stay at the lowest MSE.

5.1 Varying the General Memory Polynomial parameters

The parameters of the GMP were chosen to show how their variation would affect the MSE and ACLR. When the number of coefficients increases, the saturation rate generally decreases. This means that the GMP variants with less coefficients will reach their saturation quicker, i.e. the coefficients stabilize and there is no need for further iterations.

Test 0 in the table of Table 1 is the MSE and ACLR of the non-processed signal i.e. when no DPD has been applied. Tests 1 : 6 are meant to illustrate the effects of increasing lag parameter L_a (tests 1 : 3) and polynomial order parameter K_a (tests 4 : 6) without varying other parameters. It is clear that while L_a parameters allow for lowering the MSE of the pre-distorted signal output, it has very little affect on the ACLR, whereas the K_a parameters seem to have the complementary affect of lowering the ACLR more than the MSE. When combined, in tests 7 : 9 better results are achieved in MSE, while still maintaining a lowered ACLR.

In essence, tests 1 : 9 are the standard Memory Polynomial model since only L_a and K_a are non-empty sets. Tests 10 : 14 then serve to show if there is a benefit to employing the GMP at all. These tests show that while there does not seem to be any great improvement is MSE, the ACLR is lowered, particularly while utilizing the lag envelope parameters (L_c, K_c, M_c). Test 14 which contains non-empty sets for all parameters, shows the best result compared to the previous test, though it also consists of most summands - each with a unique coefficient. This entails that, with the adaptive Kalman Filter algorithm as optimizer, utilizing the GMP seems to be beneficial.

Tests 15 : 17 are conducted with the same parameters as in tests 18 : 20 but with 100 rather than 500 signal iterations, they strengthen the notion that the GMP outperforms the standard Memory Polynomial even further. Test 15 considered only odd numbers contained in the polynomial order parameter sets K_a, K_b, and K_c while test 16 was similar but with even numbers, it seems that the odd number polynomial order parameters could contribute more to minimizing MSE while even number polynomial orders contributed more to ACLR decrease (shown in tests 18 and 19 with 500 iterations). Notably, both the MSEs and ACLRs from tests 15 : 17 improved when increasing the iteration count to 500 in tests 18 : 20. It is also clear from Figure 12 and 13 that many of the coefficient values have not yet stabilized, but their gradient is far lower at 500 than at 100 iterations.

Tests 21 and 22 are examples of the algorithm performing quite well despite not having nearly as many summands as in test 15 : 20. Finally tests 23 : 25 show that simply increasing the amount of summands does not directly lead to improvement in MSE and ACLR, even when iterating up to 1500 times.

It can be seen in Figures 9 and 11 that, though quite a few of the coefficient estimates seem stable, a handful are still either increasing or decreasing. It is reasonable to assume that when one coefficient increases, others may very well be negatively correlated and decrease as a reaction. Clearly when looking at the respective saturation graphs, this behaviour is decreasing the MSE, quite slowly however, and perhaps tweaking the variance Σ_0 would allow these to converge faster to their optimal saturated values.

Table 1: *Results of the Kalman Filter Algorithm while varying GMP parameters and signal iterations. MSE and ACLR results are in dBm*

Test	La	Ka	Lb	Kb	Mb	Lc	Kc	Mc	MSE	ACLR	Coefs	Iter
0									-39.72	-37.68		
1	[0,1]	[0]							-45.55	-38.98	2	100
2	[0:2]	[0]							-45.57	-38.99	3	100
3	[0:3]	[0]							-45.63	-38.98	4	100
4	[0]	[0,1]							-43.42	-40.43	2	100
5	[0]	[0:2]							-43.65	-41.37	3	100
6	[0]	[0:3]							-43.69	-41.44	4	100
7	[0,1]	[0,1]							-46.86	-40.25	4	100
8	[0,1]	[0:2]							-47.45	-41.13	6	100
9	[0:2]	[0:2]							-47.48	-41.16	9	100
10	[0,1]	[0,1]	[0,1]	[1]	[1]				-46.87	-40.25	6	100
11	[0,1]	[0,1]	[0,1]	[1,2]	[1]				-47.22	-40.68	8	100
12	[0,1]	[0,1]				[0,1]	[1]	[1]	-46.87	-40.26	6	100
13	[0,1]	[0,1]				[0,1]	[1,2]	[1]	-47.26	-40.72	8	100
14	[0,1]	[0,1]	[0,1]	[1,2]	[1]	[0,1]	[1,2]	[1]	-47.32	-40.97	12	100
15	[0:3]	[0,1,3,5]	[0:2]	[1,3,5]	[1,2]	[0:2]	[1,3,5]	[1,2]	-47.84	-41.77	52	100
16	[0:3]	[0,2,4,6]	[0:2]	[2,4,6]	[1,2]	[0:2]	[2,4,6]	[1,2]	-47.66	-41.43	52	100
17	[0:6]	[0:5]							-47.77	-41.51	42	100
18	[0:3]	[0,1,3,5]	[0:2]	[1,3,5]	[1,2]	[0:2]	[1,3,5]	[1,2]	-47.87	**-42.17**	52	500
19	[0:3]	[0,2,4,6]	[0:2]	[2,4,6]	[1,2]	[0:2]	[2,4,6]	[1,2]	**-47.92**	-42.13	52	500
20	[0:6]	[0:5]							-47.86	-41.89	42	500
21	[0:2]	[0:2]	[0,1]	[1,3]	[2]	[0,1]	[1,3]	[2]	-47.63	-41.57	17	100
22	[0:2]		[0,1]	[1,3]	[2]	[0,1]	[1,3]	[2]	-47.73	-41.94	17	500
23	[0:6]	[0:5]	[0:6]	[1:3]	[1:3]	[0:6]	[1:3]	[1:3]	-47.62	-41.63	168	100
24	[0:6]	[0:5]	[0:6]	[1:3]	[1:3]	[0:6]	[1:3]	[1:3]	-47.66	-42.05	168	500
25	[0:6]	[0:5]	[0:6]	[1:3]	[1:3]	[0:6]	[1:3]	[1:3]	-47.66	-42.06	168	1500

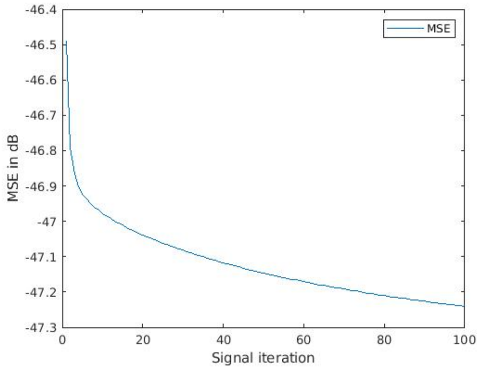

Figure 8: *MSE vs iterations test 13 from Table 1*

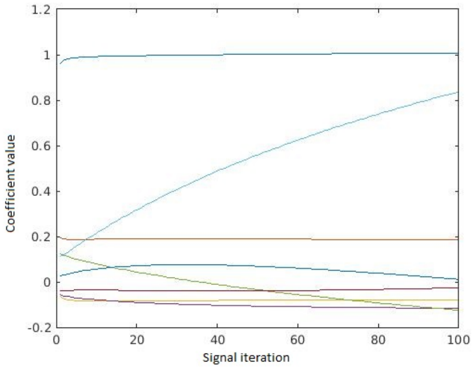

Figure 9: *Real coefficients vs iterations test 13 from Table 1*

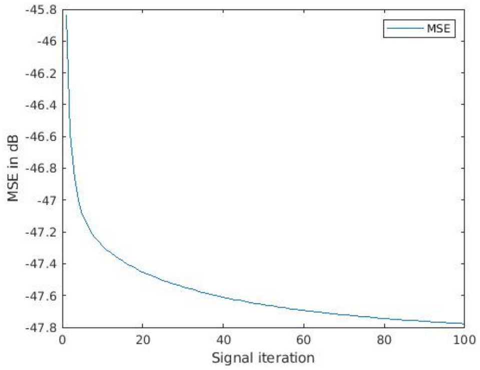

Figure 10: *MSE vs iterations test 17 from Table 1*

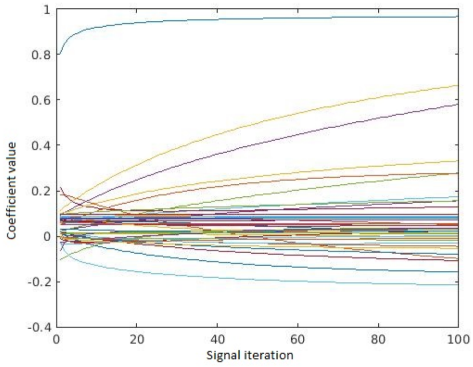

Figure 11: *Real coefficients vs iterations test 17 from Table 1*

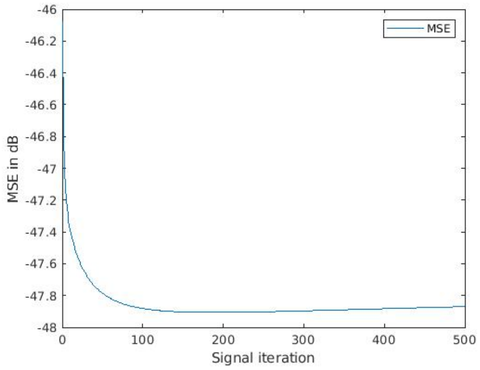

Figure 12: *MSE vs iterations test 18 from Table 1*

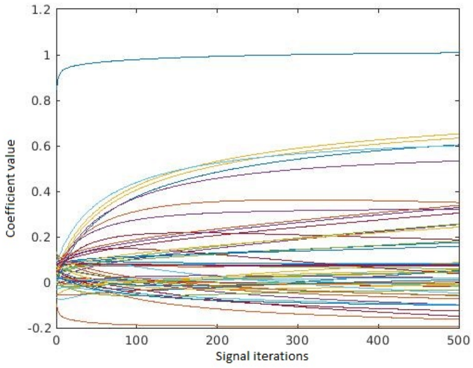

Figure 13: *Real coefficients vs iterations test 18 from Table 1*

5.2 Least Squares Adaptive Algorithm Results

Table 2 shows the results of tests 9, 13, 14 and 17 when the LS algorithm is used instead of the KF algorithm. In general, the LS algorithm seemed to be less stable and less consistent. It was also far less reliable and saturated slower when the amount of summands increased. Clearly test 17 had at this point not reached a good saturation which ended up increasing ACLR rather than decreasing it. The performance was quite good in test 13 but still performed worse than the KF algorithm.

Table 2: *Some comparative results of the Least Squares Filter Algorithm labelled in accordance with the results from Table 1. MSE and ACLR results are in dBm*

Test	La	Ka	Lb	Kb	Mb	Lc	Kc	Mc	MSE	ACLR	Coefs	Iter
9	[0:2]	[0:2]							-44.18	-40.79	9	100
13	[0,1]	[0,1]				[0,1]	[1,2]	[1]	-46.36	-40.28	8	100
14	[0,1]	[0,1]	[0,1]	[1,2]	[1]	[0,1]	[1,2]	[1]	-40.47	-36.45	12	100
17	[0:6]	[0:5]							-39.45	-36.86	42	100

Figures 15 and 17 show that the development of the coefficients in the LS algorithm do not show the same consistency or stability as in Figures 9 and 11 which represent the same GMP models but with the KF algorithm. There are quite clear signs that the KF algorithm is comparatively more consistent in the dispersion of coefficient estimates, and they seem stay within the interval $[-0.5, 1]$ or so. The KF algorithm also handles large GMP models far better than the LS algorithm, Test 17 from Table 1 has 42 GMP summands, the LS algorithm was almost 8 *dBm* higher in MSE after 100 signal iterations.

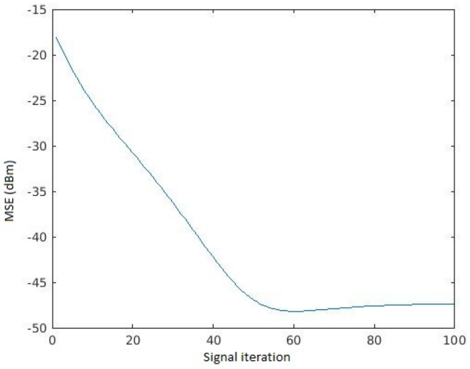

Figure 14: *MSE vs iterations LS test 13 from Table 2*

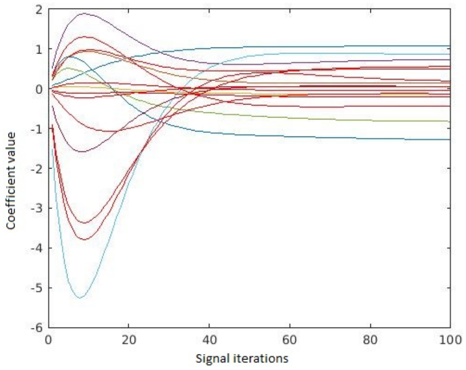

Figure 15: *Real coefficients vs iterations LS test 13 from Table 2*

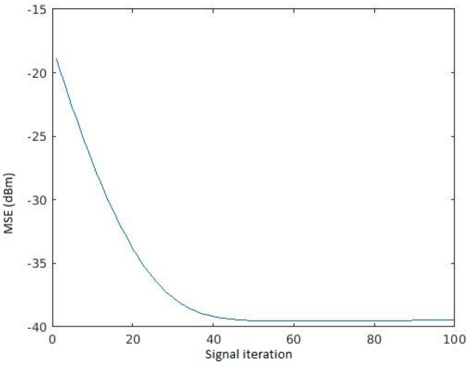

Figure 16: *MSE vs iterations LS test 17 from Table 2*

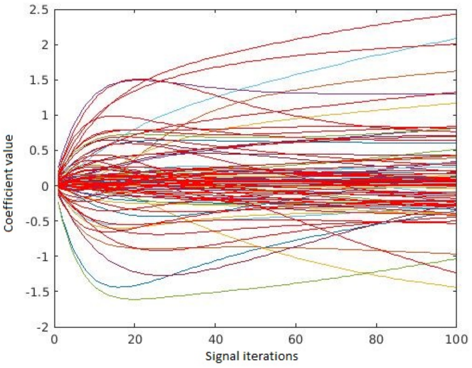

Figure 17: *Real coefficients vs iterations LS test 17 from Table 2*

5.3 Kalman Filter Algorithm Performance Metrics

The KF algorithm Test 19 performed the best of the few GMP models that were optimized, Figures 18 and 19 show the reduced dispersion of the AM/AM characteristic and the AM/PM characteristic. This entails that the memory effects of the PA have been reduces by the DPD. Figure 20 illustrates a clear decrease in ACLR in the Pre-Distorted signal PSD. The reduction in ACLR may not seem incredibly drastic, but on the left and right hand sides of the main channel, the edges of the signal[14] there is a PSD value decrease of $-8\ dBm$ and $-10\ dBm$ respectively. In general, after this much reduction of the ACLR, the error between true signal and Pre-Distorted output is so small that it has a hard time driving the algorithm to update its estimates.

[14] Which is most vital since due to antenna behaviour

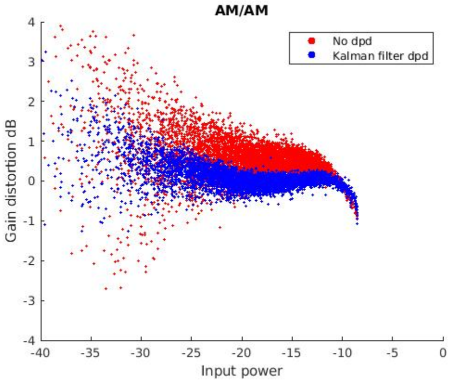

Figure 18: *AM/AM characteristics before and after KF DPD with GMP model from Test 19*

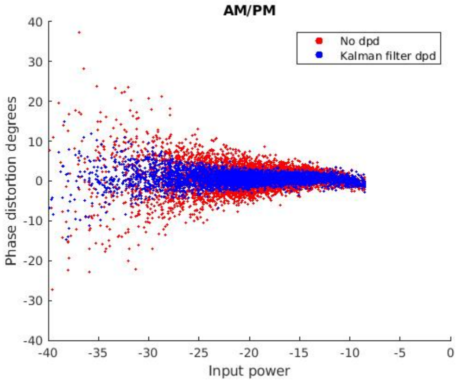

Figure 19: *AM/PM characteristics before and after KF DPD with GMP model from Test 19*

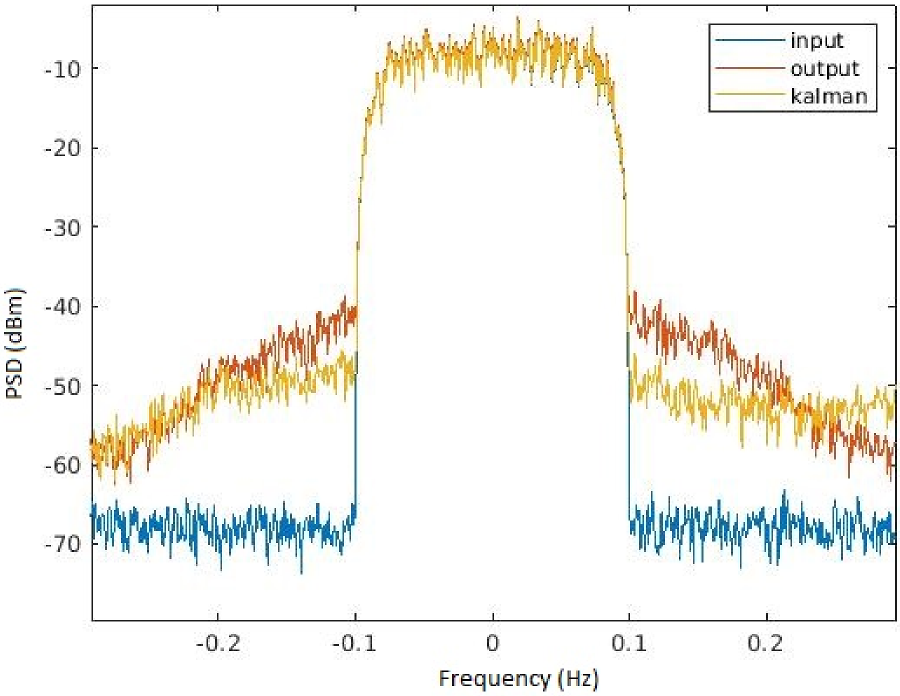

Figure 20: *Power Spectral Densities of KF DPD output (Yellow), non-DPD output (Red), and true signal (Blue)*

6 Conclusion

The performance of the KF algorithm was better in MSE and ACLR in each comparative test than the benchmark, but this does not necessarily mean that it is a suitable replacement algorithm, there are other things to consider such as memory and computational cost.

The computational cost of the KF algorithm may or may not be too high to effectively implement in a communications system, this is difficult to comment on in general. In terms of memory however, the only additional memory that the KF algorithm needs when compared to the LS algorithm, is the Σ_t matrix. According to the results, it is very arguable that Σ_t is worth the memory cost. Even quite a small Σ_t matrix (for a relatively small GMP model) gives a considerable performance improvement in MSE, ACLR, and reduction in memory effects.

The Σ_t matrix is a covariance matrix of the estimated hidden states $\boldsymbol{\alpha}$ which seems to contribute hugely to the stability and consistency of the development of the coefficients. Not only did the KF algorithm show more consistency and stability in coefficient development, but it also considers potential correlation between coefficients. This allows for peculiar behaviour of coefficient optimization. It entails that for instance, two coefficients can be negatively correlated i.e. as one decreases, the other increases. When looking at the Rate of Saturation and the development of coefficients in Figures 8 and 9 it is clear that the MSE is still on a downwards slope, while some coefficients are not stabilized. In fact, it is quite clear that there is one coefficient which is quite drastically still increasing and two are still decreasing at a lower rate. Combined, if these three coefficients are correlated then the three summands may contribute to a sequence which is converging, then the sequence in turn has its own optimal combined contribution to the optimization of the GMP model.

There may be a discussion to be had about the use of the Conventional Complex Kalman Filter [9] in this thesis. Not including the pseudo covariance of the complex variables, may very well have been what caused the KF algorithm to drive the complex parts of the coefficients to zero. Though as previously mentioned, it could also be due to the nature of the problem. Since the PA's main function is to increase the power of the signal, which should affect the amplitude of the signal rather than the phase. then the summands of the GMP could very well contain enough variations of the signal to represent the inverse of the PA as a linear combination of complex summands with real valued coefficients. This could very well be an oversight, though having real rather than complex coefficients minimizes memory and computational costs, which could outweigh the value of a potential performance increase when including the pseudo variance.

6.1 Future Adaptations

There are possible adjustments to improve performance in the choice of Q_t and Σ_0 which affect the convergence rate of the algorithm. Of course there is then a trade off between stability and convergence rate. There are also potential combinations of a Kalman Filter and a Neural Network, as mentioned in the introduction, which show promising results, but again at the cost of complexity, there is always a balance between performance and hardware requirement that must be considered.

In the spirit of trying to make this KF algorithm even more efficient, in terms of using less memory, and computing smaller matrices, there was an attempt during this project, to systematically reduce the amount of summands in large GMP models that seemed to not contribute much. In this case, the 'contribution' to optimizing the GMP model was simply seen as how large the coefficient of the summand was. For instance, if a coefficient was less than 0.05 after 100 iterations perhaps, it would be omitted and the algorithm would continue to optimize the remaining coefficients. This showed poor results however. The problem seemed to stem from the fact that a low valued coefficient could still be strongly correlated to another coefficient, and then the contribution of its correlated coefficient would be reduced if the low valued partner was removed.

For future expansions of this project, a similar attempt to find an optimal GMP could be tried, but rather than removing summands with low valued coefficients, add the criteria that they must also be relatively uncorrelated to other coefficients. This information can be found from the empirical data obtained by the KF in the Σ_t matrix, it contains all the information about covariances between the coefficients.

References

[1] Toshiyuki Eda, Takanori Ito, Hiromitsu Ohmori, and Akira Sano. Adaptive compensation of nonlinearity in high power amplifier by support vector machine. *IFAC Proceedings Volumes*, 34(14):243–248, 2001. IFAC Workshop on Adaptation and Learning in Control and Signal Processing (ALCOSP 2001), Cernobbio-Como, Italy, 29-31 August 2001.

[2] Himanshu Gaur and Md Zahidul Islam Shahin. Efficient dpd coefficient extraction for compensating antenna crosstalk and mismatch effects in advanced antenna system, 2018. Student Paper.

[3] Fadhel M. Ghannouchi, Oualid Hammi, and Mohamed Helaoui. *Behavioral Modelling and Predistortion of Wideband Wireless Transmitters*. John Wiley & Sons, 2015.

[4] MATLAB. *version 9.7.0 (R2010b)*. The MathWorks Inc., Natick, Massachusetts, 2019.

[5] Farouk Mkadem and Slim Boumaiza. Physically inspired neural network model for rf power amplifier behavioral modeling and digital predistortion. *IEEE Transactions on Microwave Theory and Techniques*, 59(4):913–923, 2011.

[6] Dennis R. Morgan, Zhengxiang Ma, Jaehyeong Kim, Mike G. Zierdt, and John Pastalan. A generalized memory polynomial model for digital predistortion of rf power amplifiers. *IEEE Transactions on Signal Processing*, 54(10):3852–3860, 2006.

[7] Bernard Picinbono and Pascal Bondon. Second-order statistics of complex signals. *IEEE Transactions on Signal Processing*, 45(2):411–420, 1997.

[8] Frederick Raab, P.M. Asbeck, S.C. Cripps, Peter Kenington, Z.B. Popovic, Nick Pothecary, John Sevic, and Nathan Sokal. Power amplifiers and transmitters for rf and microwave. *Microwave Theory and Techniques, IEEE Transactions on*, 50:814 – 826, 04 2002.

[9] Gang Wang, Shuzhi Sam Ge, Rui Xue, Ji Zhao, and Chao Li. Complex-valued kalman filters based on gaussian entropy. *Signal Processing*, 160:178–189, 2019.

[10] Linhuang Wu, Kaixiong Su, Zhifeng Chen, and Pingping Chen. A low complexity extended kalman filter algorithm for neural network digital predistortion of power amplifier. In *2017 International Conference on Green Informatics (ICGI)*, pages 17–24, 2017.

[11] Jin Xu, Weiliang Jiang, Linhua Ma, Mingyu Li, Zhiqiang Yu, and Zhen Geng. Augmented time-delay twin support vector regression-based behavioral modeling for digital predistortion of rf power amplifier. *IEEE Access*, 7:59832–59843, 2019.