RAPID GAIN ESTIMATION FOR MULTI-USER SOFTWARE DEFINED RADIO APPLICATIONS

Viraj Gajjar and Kurt Kosbar
Missouri University of Science and Technology
Rolla, MO, 65409
vgf4c@mst.edu, kosbar@mst.edu

ABSTRACT

This paper proposes a machine learning algorithm to estimate the peak of a signal generated as a sum of modulated signals. Each signal in the sum may have different modulation format, carrier frequency, data rate, and power level. A dataset of summed signals with varying parameters of individual signals was simulated to train an artificial neural network. The neural network estimates the peak voltage, central moments, variance, skewness, and kurtosis of the summed signal. Once trained, the neural network can rapidly and accurately predict the peak voltage, and statistics of the summed signal, based on the individual signal parameters, and does not have to generate or observe the signal itself. This can be used in an automatic gain control system, to prevent clipping when multiple modulators share a digital to analog converter or amplifier chain.

INTRODUCTION

A software defined radio (SDR) can be used to generate multiple signals with varying modulation formats, data rates, power levels, and carrier frequency [1]. The multiple baseband modulated signals can be added in the SDR, and the summed signal sent to a digital-to-analog converter (DAC) for transmission. To minimize quantization noise, the summed signal needs to be as large as possible. However, exceeding the range of DAC will cause distortion due to clipping. It is possible to look for the peak voltage during transmission, and then adjust the gain based on the peak voltage. But using such a system may lead to distortion in first part of the transmission, and the problem gets worse if the number of transmitters allotted to the SDR changes. This paper proposes a way to estimate gain based on signal parameters such as modulation formats, data rates, power levels, and carrier frequencies of the summed signals and the number of signals added; without having to look at the summed signal.

Estimating statistics of the summed signal such as mean, variance, and standardized moments [2] can help determine how frequently any particular peak voltage occurs in the summed signal. If the

peak is an outlier, then the system may consider clipping it for better efficiency, and adjust the gain based on some smaller peak which occurs more frequently. Estimating the peak voltage and the statistics of the summed signal based solely on the signal parameters can be considered as a function approximation problem.

A class of machine learning algorithms, supervised learning, can learn function that maps an input to the output based on available data, which consists of input-output pairs [3]. Supervised learning has been used in applications such as estimating the channel state information for a MIMO communication link based on atmospheric conditions [4], to estimate channel noise statistics in MIMO wireless network [5], to provide optimal handover solutions by learning a mobile terminal's usage pattern [6], and to improve pilot contamination in massive MIMO by learning channel parameters of the desired link in target cell and undesired link in the adjacent cells [7]. Supervised machine learning algorithms such as artificial neural networks (ANN) can solve function fitting problems, and can approximate nearly any continuous function under certain assumptions [8, 9].

This paper demonstrates a way to use ANN for estimating peak voltage and statistics of a signal for gain estimation using simulated data. Specifically, multiple signals with varying modulation formats, power levels, data rates, and carrier frequency are summed, and the peak voltage and the statistics of this summed signal are measured. The parameters of the signals summed, and the number of signals, form the input of the ANN, and the peak voltage and the statistics form the output. A dataset using 125,000 summed signals is generated, of which 70% of the data is used to train, and the remaining 30% is used to validate and test the efficiency of the trained ANN.

THE DATASET

The summed signal is generated by adding multiple signals with varying modulation formats, data rates, power levels, and carrier frequencies. Table 1 shows the range of parameters used to generate the summed signals for the dataset.

Table1 Dataset Parameter Range

Parameter	Range
Modulation technique	BPSK, QPSK, 8-QAM, 16-QAM, 32-QAM, 64-QAM, 128-QAM, and 256-QAM [10]
Bit rate	1000 bps to 10000 bps
Power level	-30 dBm to 0 dBm
Normalized Carrier frequency	-60 kHz to 60 kHz

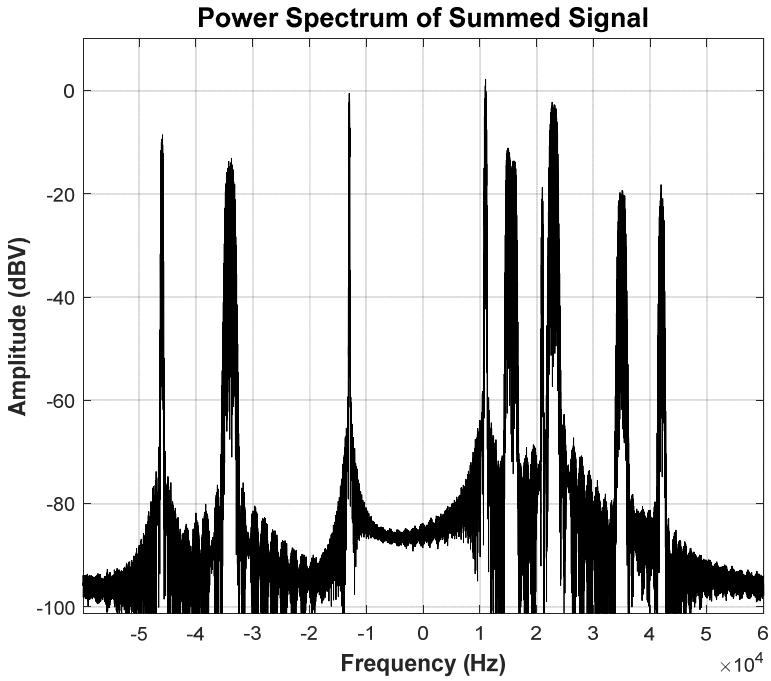

Figure 1 Power spectrum of the summed signal

Figure 1 shows the power spectrum of the summed signal, where ten signals are added. To generate a summed signal, multiple signals are generated by randomly varying the parameters within the range of each parameter mentioned in Table 1.

The following statistics of the summed signal were considered for simulating the dataset and for estimation: mean, variance, skewness (third standardized moment), kurtosis (fourth standardized moment), fifth standardized moment, and sixth standardized moment. Standardized moments can be found using the equation 1.

$$\zeta_k = \frac{\mu_k}{\sigma^k} = \frac{E[(X-\mu)^k]}{(\sqrt{E[(X-\mu)^2]})^k} \tag{1}$$

Where, ζ_k is the kth standardized moment, μ_k is the kth moment about the mean, σ is the standard deviation, X is a random variable, and μ is the mean. Figure 2 shows how the statistics change as more samples of the summed signal are considered. The mean converges more rapidly than any other statistic. As the order of the moments increase, more samples of the signal are required for the statistic to converge. These statistics are calculated for a summed signal with 10 modulated signals, which is the maximum number of signals considered for dataset simulation. Figure 3 shows that peak voltage takes the largest number of samples to converge when compared to the statistics.

A dataset is generated using 125000 summed signals, where each summed signal is generated by randomly selecting parameters from Table 1. The number of modulated signals used to generate a summed signal can vary between 2 and 10.

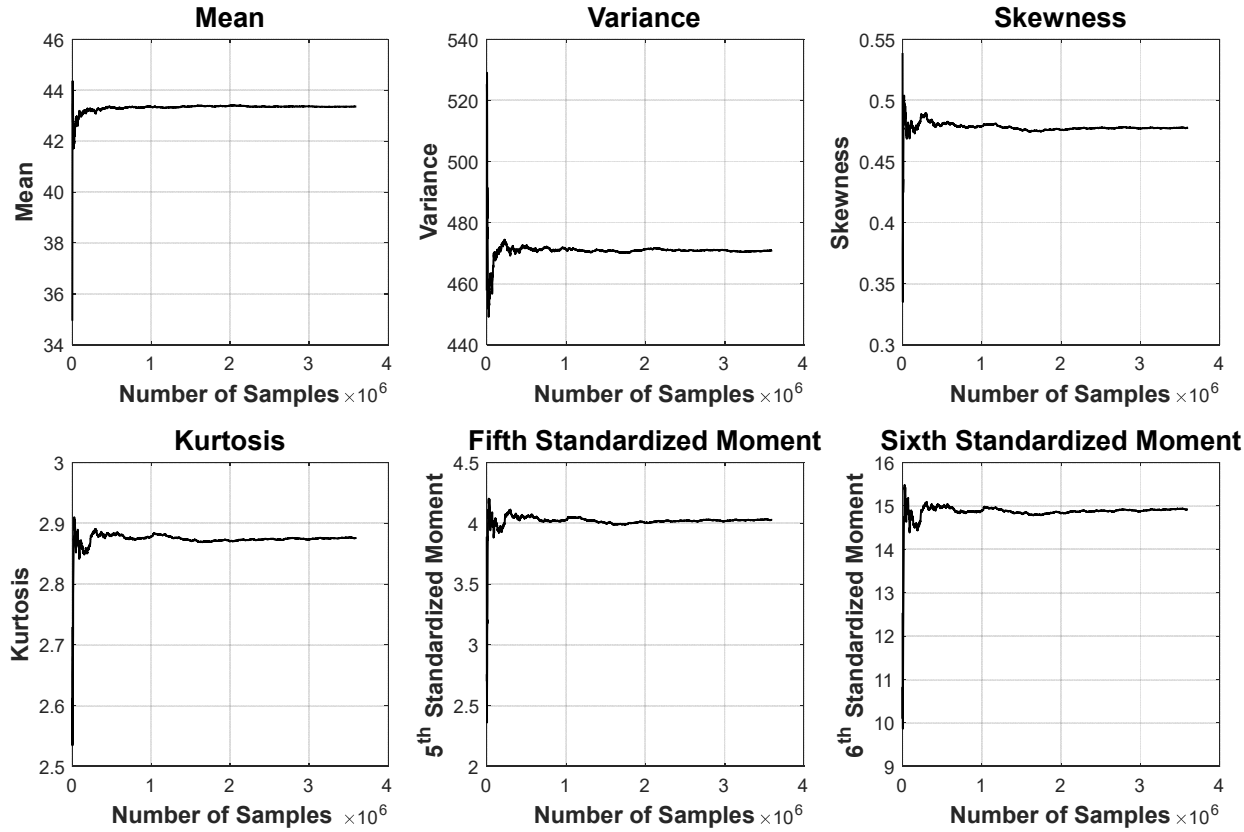

Figure 2 Convergence of statistics as the number of samples increase

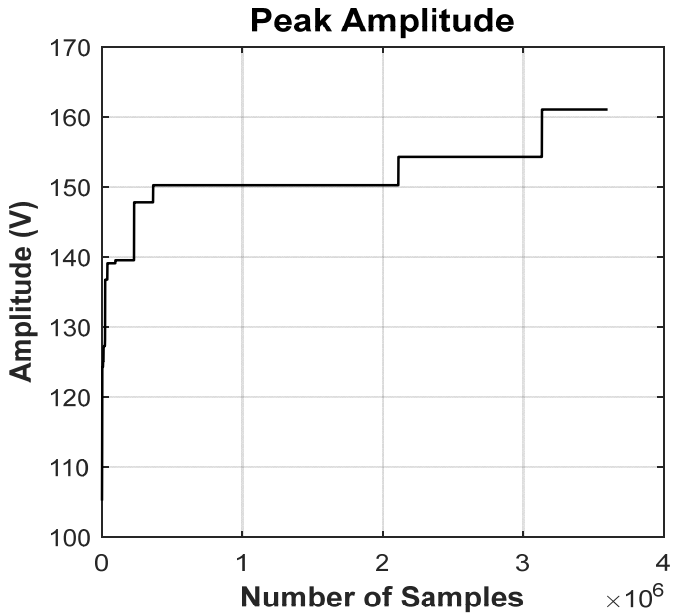

Figure 3 Convergence of peak amplitude as the number of samples increase

ARTIFICIAL NEURAL NETWORK

The generated dataset consists of two matrices, an input feature matrix, and an output matrix of statistics and peak amplitudes. The input feature matrix consists of parameters: modulation order, bitrate, power level, and carrier frequency. The parameters are specified for each component signal used to generate the summed signal. The number of signals in the sum is also part of the feature matrix. The output matrix consists of the six statistics and the peak amplitude of the summed signal. Therefore, the input feature matrix is 125,000-by-41, where 125,000 are the number of data samples and 41 represents the number features for each data sample (4 features for each component signal for a maximum of 10 added signals, and one feature for the number of signals added). If fewer than ten signals are summed, the input features for unused signals are set to zero. Similarly, the output feature matrix is 125,000-by-7, where 7 represents the peak amplitude and the six different statistics. Thus, the neural network should have 41 neurons at the input which takes a vector of features as input, and 7 neurons at the output which estimates the peak voltage and the statistics of the summed signal.

Before training the ANN, the input feature matrix is normalized, and both the input and the output datasets are divided into a training set, a validation set, and a test set. Approximately 70% of the samples of the dataset are used to train the ANN. The validation set forms 15% of the dataset, and are used to provide an unbiased evaluation of the training data fit while tuning model parameters. The remaining 15% of the dataset is the test set, which is used to evaluate the final model.

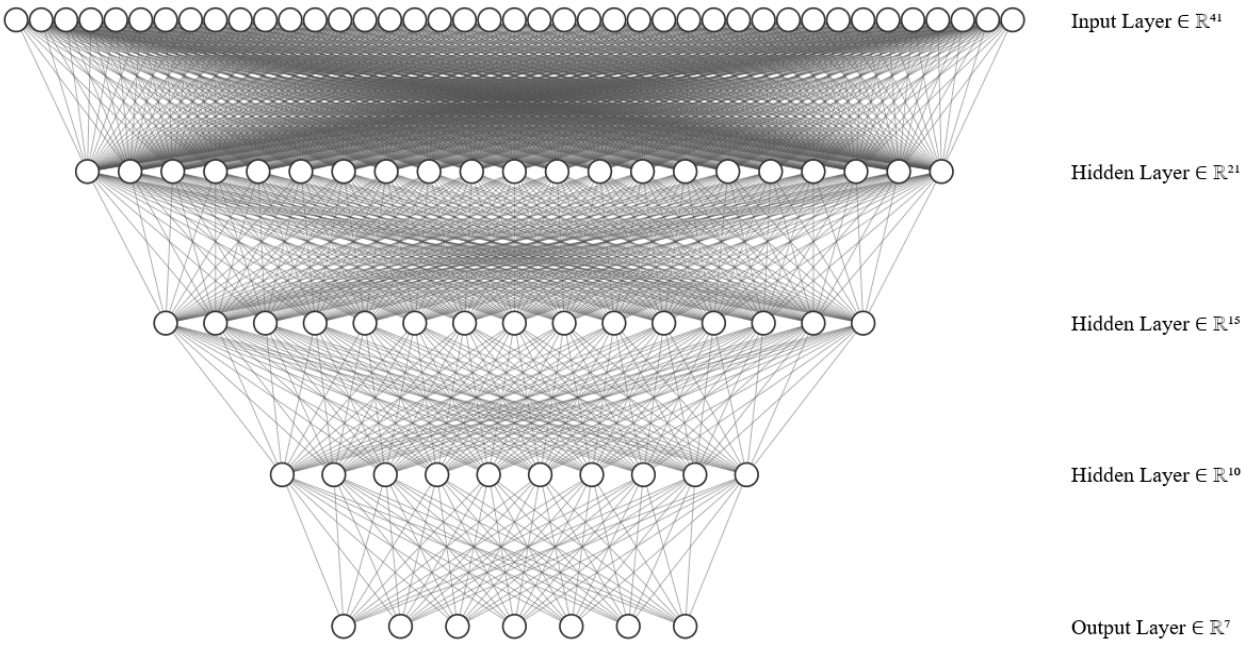

Figure 4 ANN architecture

The overall architecture of the ANN trained is shown in Figure 4. The network consists of five layers, the input layer with 41 neurons for the input features, three hidden layers with 21, 15, and 10 neurons respectively, and an output layer with 7 neurons for the peak voltage and statistics. All the weights (connections between each neuron) are randomly initialized before the training process starts. The training process starts by feeding a feature vector to the input layer and based on this input the activations of the next layer are calculated. The network calculates the activations of each subsequent layer using the following equation:

$$A^{(1)} = \sigma(WA^{(0)} + b) \tag{2}$$

where $A^{(1)}$ and $A^{(0)}$ are activations of the output layer and the input layer respectively, W is the matrix of weights connecting the input and the subsequent layer, b is the bias vector, and σ is the activation function. Equation (3) and Equation (4) are the activation functions used for the hidden layers and the output layer respectively. The linear function (Eq. 4) at the output layer is better at fitting a function compared to the tan-sigmoid function (Eq. 3).

After calculating the activations of each neuron, the cost function of the entire network is calculated by comparing the network outputs and actual outputs. This cost function is then used to calculate the gradient using Bayesian regularization backpropagation [11, 12], then the calculated gradient is used to perform gradient descent to update the weights. The network iterates through the entire dataset several times until either expected accuracy is achieved or the performance plateaus.

$$f(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{3}$$

$$f(x) = x \tag{4}$$

RESULTS AND DISCUSSIONS

The ANN was trained for 500 epochs, and the performance of the network was tested on the test set which included samples that the ANN was not exposed to during the training stage. Figure 5 shows the performance of the network on the training set and the test set in terms of mean squared error (MSE). It can be observed that the network starts with a high MSE as the weights are randomly initialized for the first epoch. Once the network starts training, the MSE drops rapidly. Over the first 100 epochs of training, the MSE drops from 50.29 to 0.1757. The performance of the ANN does not improve substantially after the first 100 epochs, and the drop in the MSE is 0.0238 over the next hundred epochs. After that point, the drop in the MSE plateaus, and the improvement in performance is negligible over the last 300 epochs. The final MSE of 0.1384 over the entire training set shows that the ANN was able to closely fit a function that is able to map the parameters of the input modulated signals, to the peak amplitude and the statistics of the summed

signal. The performance plot shows the training set slightly outperforms the test set. This is expected, as the test set consists of samples that the ANN has not seen and is using the trained weights and the trained activations of the neurons to estimate the output.

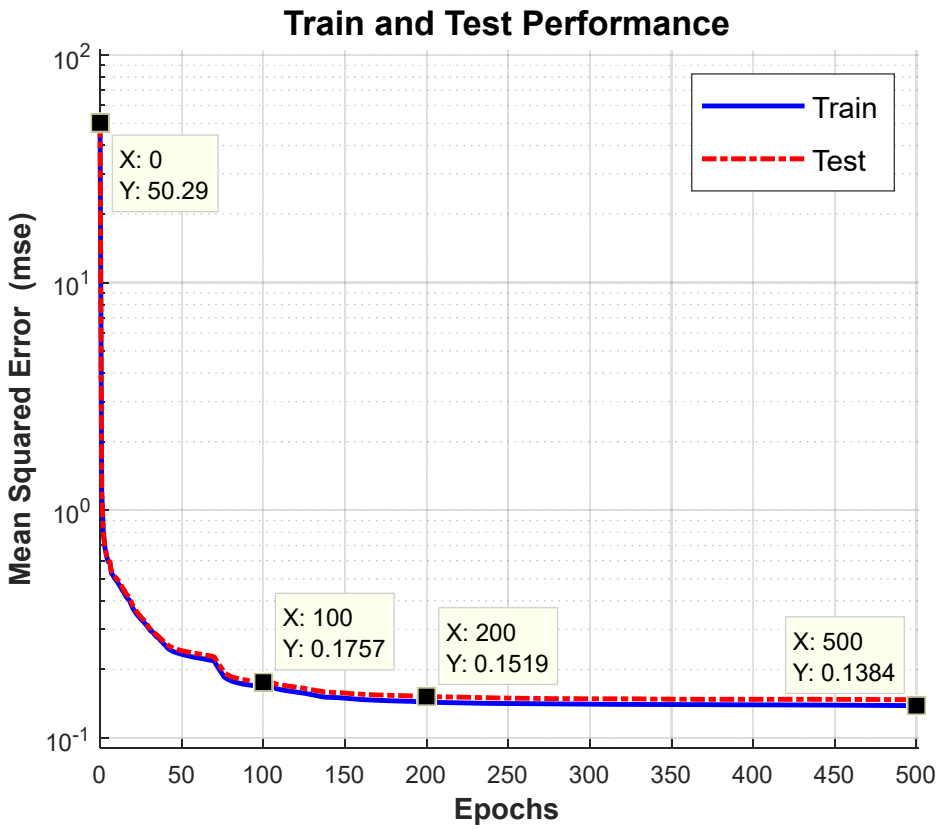

Figure 5 MSE of the Train and Test set plotted against number of epochs

Figure 4 shows the error histograms with 20 bins for errors after 1 epoch (left) and for errors after 500 epochs (right). The error histogram illustrates how far the outputs of the ANN deviate from the target values. When the ANN is trained for 1 epoch, the histogram is spread out, compared to when the ANN is trained for 500 epochs. There are approximately 10^5 error instances close to 1 when the neural network is trained only for 1 epoch, but when the network is trained for 500 epochs the number of error instances close to 1 reduces to 20,000.

The ANN was successfully able to learn a function that maps the parameters of the added signals and the number of signals added to the statistics and the peak amplitude of the summed signal. The trained ANN would be able to estimate the peak amplitude and the statistics of the summed signal based on the parameters of the component signals. The ANN would be able to do this without having to generate the summed signal.

Figure 6 Error histograms after 1 epoch (left) and 500 epochs (right)

CONCLUSIONS

This paper demonstrated a way to estimate gain of a summed signal before it goes to a DAC, so that the full range of DAC is used. To accomplish this, a dataset of parameters of the signals added to generate summed signal and corresponding statistics and the peak amplitude of the summed signal was generated. This dataset was used to train an ANN to fit a function that maps the parameters to the statistics and the peak amplitude. This ANN was evaluated on a test set, and was able to closely estimate of the statistics and the peak amplitude. These estimates can be used to decide the amount of gain required for the SDR for optimal performance. Once trained, an ANN would be able to estimate gain based just on the parameters of the signals summed, without having to look at the entire summed signal.

REFERENCES

[1] Jondral, F.K. Software-Defined Radio—Basics and Evolution to Cognitive Radio. *J Wireless Com Network* **2005,** 652784 (2005).

[2] Kenney, J. F. and Keeping, E. S. "Moments in Standard Units." §7.8 in *Mathematics of Statistics, Pt. 1, 3rd ed.* Princeton, NJ: Van Nostrand, pp. 98-99, 1962.

[3] Stuart J. Russel, Peter Norvig (2010) *artificial Intelligence: A Modern Approach, Third Edition, Prentice Hall.*

[4] V. Gajjar and K. Kosbar, "CSI Estimation Using Artificial Neural Network," in *Proc. Int. Telemetering Conf.*, (Las Vegas, NV), pp. 294–303, Nov. 2019.

[5] P. Zhou, Y. Chang, and J. A. Copeland, "Determination of Wireless Networks Parameters through Parallel Hierarchical Support Vector Machines," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 3, Mar. 2012, pp. 505–12.

[6] B. K. Donohoo *et al.*, "Context-Aware Energy Enhancementsfor Smart Mobile Devices, *IEEE Trans. Mobile Comput.*, vol. 13, no. 8, Aug. 2014, pp. 1720–32.

[7] C.-K. Wen *et al.*, “Channel Estimation for Massive MIMO Using Gaussian-Mixture Bayesian Learning,” *IEEE Trans. Wireless Commun.*, vol. 14, no. 3, Mar. 2015, pp. 1356–68.
[8] D. F. Specht, "A general regression neural network," in *IEEE Transactions on Neural Networks*, vol. 2, no. 6, pp. 568-576, Nov. 1991.
[9] Balázs Csanád Csáji (2001) Approximation with Artificial Neural Networks; Faculty of Sciences; Eötvös Loránd University, Hungary.
[10] B. Sklar, Digital Communications: Fundamentals and Applications, Englewood Cliffs, NJ: Prentice Hall, 1988.
[11] MacKay, David J. C. "Bayesian interpolation." *Neural computation.* Vol. 4, No. 3, 1992, pp. 415–447.
[12] Foresee, F. Dan, and Martin T. Hagan. "Gauss-Newton approximation to Bayesian learning." *Proceedings of the International Joint Conference on Neural Networks*, June, 1997.