GPU Implementation of Data-Aided Equalizers

Jeffrey T. Ravert

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Michael D. Rice, Chair
Brian X. Jeffs
Brian X. Mazzeo

Department of Electrical and Computer Engineering

Brigham Young University

April 2017

ABSTRACT

GPU Implementation of Data-Aided Equalizers

Jeffrey T. Ravert

Department of Electrical and Computer Engineering

Master of Science

Multipath is one of the dominant causes for link loss in aeronautical telemetry. Equalizers have been studied to combat multipath interference in aeronautical telemetry. Blind Constant Modulus Algorithm (CMA) equalizers are currently being used on SOQPSK-TG. The Preamble Assisted Equalization (PAQ) has been funded by the Air Force to study data-aided equalizers on SOQPSK-TG. PAQ compares side by side no equalization, data-aided zero forcing equalization, data-aided MMSE equalization, data-aided initialized CMA equalization, data-aided frequency domain equalization, and blind CMA equalization. An real time experimental test setup has been assembled including an RF receiver for data acquisition, FPGA for hardware interfacing and buffering, GPUs for signal processing, spectrum analyzer for viewing multipath events, and an 8 channel bit error rate tester to compare equalization performance. Lab tests were done with channel and noise emulators. Flight tests were conducted in March 2016 and June 2016 at Edwards Air Force Base to test the equalizers on live signals. The test setup achieved a 10Mbps throughput with a 6 second delay. Counter intuitive to the simulation results, the flight tests at Edwards AFB in March and June showed blind equalization is superior to data-aided equalization. Lab tests revealed some types of multipath caused timing loops in the RF receiver to produce garbage samples. Data-aided equalizers based on data-aided channel estimation leads to high bit error rates. A new experimental setup is been proposed, replacing the RF receiver with a RF data acquisition card. The data acquisition card will always provide good samples because the card has no timing loops, regardless of severe multipath.

Keywords: MISSING

ACKNOWLEDGMENTS

Students may use the acknowledgments page to express appreciation for the committee members, friends, or family who provided assistance in research, writing, or technical aspects of the dissertation, thesis, or selected project. Acknowledgments should be simple and in good taste.

Table of Contents

List of Tables

List of Figures

Chapter 1

Equalizer Equations

1.1 Overview

There are 3 different kinds of equalizers I run 1. the solving ones!!! They are equations like Ax=b where I have A and b but I need x 2. the initialized then iterative ones. CMA is initialized with MMSE then runs as many times a possible 3. the multiply ones! the FDEs are a simple multiply in the frequency domain

1.2 Equations

1.2.1 The Solving Equalizers

The Zero-Forcing Equalizer

The ZF equalizer was studied in the PAQ Phase 1 Final Report in equation 324

$$\mathbf{c}_{\text{ZF}} = (\mathbf{H}^\dagger \mathbf{H})^{-1} \mathbf{H}^\dagger \mathbf{u}_{n_0} \tag{1.1}$$

where $\mathbf{c}_{\text{ZF}}$ is a $L_{eq} \times 1$ vector of equalizer coefficients computed to invert the channel estimate $\mathbf{h}$. The channel estimate is used to build the $L_{eq} + N_1 + N_2 \times L_{eq}$ convolution matrix

$$\mathbf{H} = \begin{bmatrix} h(-N_1) & & & \\ h(-N_1+1) & h(-N_1) & & \\ \vdots & \vdots & \ddots & \\ h(N_2) & h(N_2-1) & & h(-N_1) \\ & h(N_2) & & h(-N_1+1) \\ & & & \vdots \\ & & & h(N_2) \end{bmatrix}. \tag{1.2}$$

and $\mathbf{u}_{n_0}$ is the desired channel impulse response centered on $n0 = N_1 + L_1 + 1$

$$\mathbf{u}_{n_0} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \begin{matrix} \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} n_0 - 1 \text{ zeros} \\ \\ \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} N_1 + N_2 + L_1 + L_2 - n_0 + 1 \text{ zeros} \end{matrix} . \tag{1.3}$$

The computation of the coefficients in Equation (1.1) can be simplified in a couple of ways: First the matrix multiplication of $\mathbf{H}^\dagger$ and $\mathbf{H}$ is the autocorrelation matrix of the channel

$$\mathbf{R}_h = \mathbf{H}^\dagger \mathbf{H} = \begin{bmatrix} r_h(0) & r_h^*(1) & \cdots & r_h^*(L_{eq}-1) \\ r_h(1) & r_h(0) & \cdots & r_h^*(L_{eq}-2) \\ \vdots & \vdots & \ddots & \\ r_h(L_{eq}-1) & r_h(L_{eq}-2) & \cdots & r_h(0) \end{bmatrix} \tag{1.4}$$

where

$$r_h(k) = \sum_{n=-N_1}^{N_2} h(n) h^*(n-k). \tag{1.5}$$

Second the matrix vector multiplication of $\mathbf{H}^\dagger$ and $\mathbf{u}_{n_0}$ is simply the n_0th row of $\mathbf{H}^\dagger$ or the conjugated n_0th column of $\mathbf{H}$. A new vector $\mathbf{h}_{n_0}$ is defined by

$$\mathbf{h}_{n_0} = \mathbf{H}^\dagger \mathbf{u}_{n_0} = \begin{bmatrix} h(L_1) \\ \vdots \\ h(0) \\ \vdots \\ h(-L_2) \end{bmatrix} . \tag{1.6}$$

Replacing the matrix multiplication $\mathbf{H}^\dagger\mathbf{H}$ and $\mathbf{H}^\dagger\mathbf{u}_{n_0}$ simplifies Equation (1.1) to

$$\mathbf{c}_{\text{ZF}} = \mathbf{R}_h^{-1}\mathbf{h}_{n_0}. \tag{1.7}$$

Computing the inverse of $\mathbf{R}_h$ is computationally heavy because an inverse is an N^3 operation. To avoid an inverse, $\mathbf{R}_h$ is moved to the left side and $\mathbf{c}_{\text{ZF}}$ is found by solving a system of linear equations. Note that $r_h(k)$ only has support on $-L_{ch} \leq k \leq L_{ch}$ making $\mathbf{R}_h$ sparse or $\%63$ zeros. The sparseness of $\mathbf{R}_h$ can be leveraged to reduce computation drastically. The Zero-Forcing Equalizer coefficients are computed by solving

$$\mathbf{R}_h\mathbf{c}_{\text{ZF}} = \mathbf{h}_{n_0}. \tag{1.8}$$

MMSE Equalizer

The MMSE equalizer was studied in the PAQ Phase 1 Final Report in equation 330.

$$\mathbf{c}_{MMSE} = \big[\mathbf{G}\mathbf{G}^\dagger + \frac{\sigma_w^2}{\sigma_s^2}\mathbf{I}_{L_1+L_2+1}\big]\mathbf{g}^\dagger \tag{1.9}$$

where

$$\mathbf{G} = \begin{bmatrix} h(N_2) & \cdots & h(-N_1) & & & & \\ & h(N_2) & \cdots & h(-N_1) & & & \\ & & \ddots & & \ddots & & \\ & & & h(N_2) & \cdots & h(-N_1) \end{bmatrix} \tag{1.10}$$

and

$$\mathbf{g} = \Big[h(L_1)\cdots h(-L_2)\Big]. \tag{1.11}$$

The matrix multiplication $\mathbf{G}\mathbf{G}^\dagger$ is the same autocorrelation matrix $\mathbf{R}_h$ as Equation (1.4). The vector $\mathbf{g}^\dagger$ is also the same vector as $\mathbf{h}_{n_0}$. The signal-to-noise ratio estimate $\frac{1}{2\hat{\sigma}_w^2}$ is substituted in for the fraction $\frac{\sigma_w^2}{\sigma_s^2}$ using Equation 333 Rice's report. Equation (1.9) can be reformulated to

$$\big[\mathbf{R}_h + \frac{1}{2\hat{\sigma}_w^2}\mathbf{I}_{L_1+L_2+1}\big]\mathbf{c}_{\text{MMSE}} = \mathbf{h}_{n_0}. \tag{1.12}$$

The MMSE Equalizer coefficients are solved for in a similar fashion to Zero-Forcing is in Equation (1.8). The only difference between Equation (1.12) and (1.8) is the noise variance is added down the diagonal of $\mathbf{R}_h$. The matrix $\mathbf{R}_{hw}$ is defined to make the computation of the MMSE Equalizer coefficients the same as the Zero-Forcing Equalizer coefficients by adding the noise variance to the diagonal of $\mathbf{R}_h$

$$\mathbf{R}_{hw} = \mathbf{H}^{\dagger}\mathbf{H} = \begin{bmatrix} r_h(0) + \frac{1}{2\hat{\sigma}_w^2} & r_h^*(1) & \cdots & r_h^*(L_{eq}-1) \\ r_h(1) & r_h(0) + \frac{1}{2\hat{\sigma}_w^2} & \cdots & r_h^*(L_{eq}-2) \\ \vdots & \vdots & \ddots & \\ r_h(L_{eq}-1) & r_h(L_{eq}-2) & \cdots & r_h(0) + \frac{1}{2\hat{\sigma}_w^2} \end{bmatrix}. \tag{1.13}$$

The MMSE Equalizer coefficients are computed by solving

$$\mathbf{R}_{hw}\mathbf{c}_{\text{MMSE}} = \mathbf{h}_{n_0}. \tag{1.14}$$

1.2.2 The Iterative Equalizer

The Constant Modulus Algorithm

CMA is a steepest decent algorithm.

$$\mathbf{c}_{b+1} = \mathbf{c}_b - \mu \nabla \mathbf{J} \tag{1.15}$$

The vector $\mathbf{J}$ is the cost function and $\nabla \mathbf{J}$ is the cost function gradient defined in the PAQ report 352 by

$$\nabla \mathbf{J} = \frac{2}{L_{pkt}} \sum_{n=0}^{L_{pkt}-1} \Big[y(n) y^*(n) - R_2 \Big] y(n) \mathbf{r}^*(n). \tag{1.16}$$

where the constant $2N_b$ from the report is the number of samples in a packet L_{pkt}. The term $y(n)$ is a sample of the equalized signal using the equalizer $\mathbf{c}_b$. The vector $\mathbf{r}(n)$ is the received samples that have been derotated to compensate for frequency offset but unequalized defined in the PAQ report by

$$\mathbf{r}(n) = \begin{bmatrix} r(n+L_1) \\ r(n+L_1-1) \\ \vdots \\ r(n+1) \\ r(n) \\ r(n-1) \\ \vdots \\ r(n-L_2) \end{bmatrix}. \tag{1.17}$$

Note the PAQ report defines the vector $\mathbf{r}(n)$ in reverse order.

The cost function gradient in Equation (1.16) can be reformulated to look like a convolution. To use convolution to compute $\nabla \mathbf{J}$, the output of the convolution and an input is time reversed. To show this, the equation is first reformulated into a simpler summation then an input and the output is time reversed.

To reform Equation (1.16) into a simpler summation, the scalar $z(n)$ is substituted in by

$$z(n) = \Big[y(n) y^*(n) - R_2 \Big] y(n) \tag{1.18}$$

making the computation of $\nabla \mathbf{J}$ to be

$$\nabla \mathbf{J} \approx \frac{2}{L_{pkt}} \sum_{n=0}^{L_{pkt}-1} z(n)\mathbf{r}^*(n). \tag{1.19}$$

Equation (1.19) is simply a summation of scaled vectors. Expanding and writing out the summation for $\nabla \mathbf{J}$ gives

$$\begin{bmatrix} \nabla J(0) \\ \nabla J(1) \\ \vdots \\ \nabla J(k-1) \\ \nabla J(k) \\ \nabla J(k+1) \\ \vdots \\ \nabla J(L_{pkt}-1) \end{bmatrix} = z(0) \begin{bmatrix} r^*(L_1) \\ r^*(L_1-1) \\ \vdots \\ r^*(L_1-(k-1)) \\ r^*(L_1-k) \\ r^*(L_1-(k+1)) \\ \vdots \\ r^*(-L_2) \end{bmatrix} + z(1) \begin{bmatrix} r^*(1+L_1) \\ r^*(1+L_1-1) \\ \vdots \\ r^*(1+L_1-(k-1)) \\ r^*(1+L_1-k) \\ r^*(1+L_1-(k+1)) \\ \vdots \\ r^*(1-L_2) \end{bmatrix} + \cdots. \tag{1.20}$$

The expansion of (1.19) shows a new summation can be formed to compute a single index of $\nabla \mathbf{J}$ to be.

$$\nabla J(k) = \frac{2}{L_{pkt}} \sum_{n=0}^{L_{pkt}-1} z(n) r^*(n + L_1 - k). \tag{1.21}$$

Now that the computation of an element of $\nabla \mathbf{J}$ is a summation, Equation (1.21) can be massaged to look something like a convolution of the vectors $\mathbf{z}$ and $\mathbf{r}^*$

$$\nabla J(k) = \sum_{n=0}^{N-1} z(n) r^*(k - n + C) \tag{1.22}$$

where C is a constant.

To make the computation of $\nabla J(k)$ in Equation (1.21) to look like the convolution in Equation (1.22), the signs on n and k in the index for r^* must be changed. To change the signs on k and n, the vectors $\mathbf{r}^*$ and $\nabla \mathbf{J}$ are time reversed. To time reverse the L_{eq} long output vector $\nabla \mathbf{J}$, the index k is substituted with $L_{eq} - k$ in $\mathbf{r}^*$ and $\nabla J(k)$. To time reverse the L_{pkt} long input vector $\mathbf{r}^*$, the index n is substituted with $L_{pkt} - n$ in only $\mathbf{r}^*$.

Time reversing $\mathbf{r}^*$ and $\nabla \mathbf{J}$ in Equation (1.21) results in

$$\begin{aligned}\nabla J(L_{eq}-k) &= \frac{2}{L_{pkt}} \sum_{n=0}^{L_{pkt}-1} z(n) r^*((L_{pkt}-n)+L_1-(L_{eq}-k)) \\ &= \frac{2}{L_{pkt}} \sum_{n=0}^{L_{pkt}-1} z(n) r^*(k-n+C) \end{aligned} \tag{1.23}$$

where $C = L_1 + L_{pkt} - L_{eq}$. Equation (1.23) now looks just like (1.22) and it shows the time reversed $\nabla \mathbf{J}$ can be computed using a scaled convolution of the vectors $\mathbf{z}$ and time reversed $\mathbf{r}^*$.