

**Convex Optimization  
of a  
First-Order Sigma Delta Modulator**

**Jon Dattorro**

**A project submitted to  
Prof. Stephen Boyd  
T.A. Cesar Crusius**

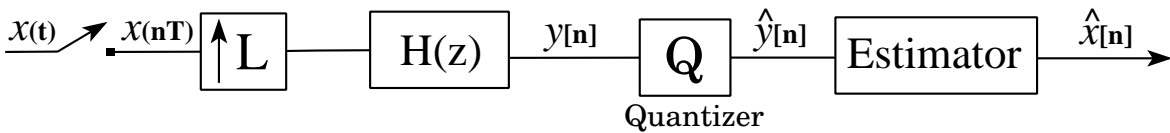
**in partial fulfillment of the requirements for**

**EE364  
Stanford  
Winter 1999**

## 1. Introduction

We define our optimization problem in Section 2, but first we give some background.

This project grew out of an idea that Prof. Boyd presented in EE263 for enhanced encoding of ordinary successive-approximation-type A/D converters. He noticed that by upsampling and placing some known filtering operation ahead of the quantizer  $Q$ , as illustrated in Figure 1, improved estimates of the discrete-time input signal could be obtained by least-squares methods operating upon the observed quantized output. While the reduction in noise power of the estimation is roughly proportional to the upsampling factor  $L$ , as predicted by classical high-resolution distortion theory, the surprising observation is that there exists a wide latitude of filters  $H(z)$  for which this noise reduction can be obtained.<sup>1</sup> The set of *usable filters* includes the first-order lowpass, second-order bandpass, "filters" having white uniformly distributed random noise constituting their impulse response, and the classical linear-phase FIR brick-wall designs. This list is not exhaustive, but is neither the topic of this investigation.



**Figure 1.** Boyd's A/D converter.

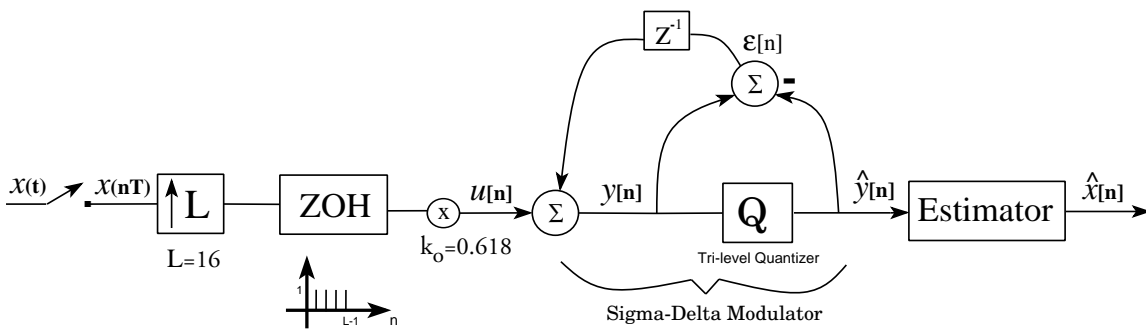
The foremost problem with the design in Figure 1 is that the quantizer works poorly on low level input signals when the quantizer is low rate; i.e., one or two bits. The reason is simply that the thresholds of the quantizer are never crossed by consistently low-level input signals. More traditional DSP methods of improving the dynamic range of the low-rate quantizer  $Q$  have focused mainly upon the quantizer input through the use of dithering, memoryless nonlinearity, or range scaling of the input signal (floating point). We choose a more contemporary method to improve quantizer performance, called sigma-delta modulation. In the following sections, we construct estimators by applying methods of convex optimization to the observed output of a first-order sigma-delta modulator.

---

<sup>1</sup>It is important to recognize that the noise reduction of the estimate is *not* brought about by the filtering effect upon quantization noise, because the filter  $H(z)$  is ahead of the quantizer.

## 2. The Problem Statement

Figure 2 presents the sigma-delta type A/D converter circuit that is the subject of this investigation. The quantizer of Figure 1 has been replaced with a sigma-delta modulator. The estimator observes only the quantized output  $\hat{y}[n]$  from the sigma-delta modulator, and produces an estimate  $\hat{x}[n]$  of the discrete-time input signal  $x(nT)$ . The problem that we wish to investigate is whether there exists an estimator that can be expressed in the form of a convex optimization problem. Given that there exist several such estimators, then we wish to know if an interior-point type estimator produces a closer estimate to the original discrete-time input signal which itself is presumably represented deep inside the feasible set  $\{\hat{x}[n] \mid f: x(nT) \rightarrow \hat{y}[n]; n=0 \dots N-1\}$ . Of course the job of the estimator will be easier if the preceding circuitry succeeds in diminishing the size of the feasible set. We reiterate that many simple filters diminish the feasible set to the same degree as do filters of much higher order and complexity. For the remaining sections, we accept this as fact based upon empirical observation of many feasible sets in 3-space, and we substitute the zero-order hold (ZOH) as a place-holder for the filter  $H(z)$  throughout.<sup>2</sup>

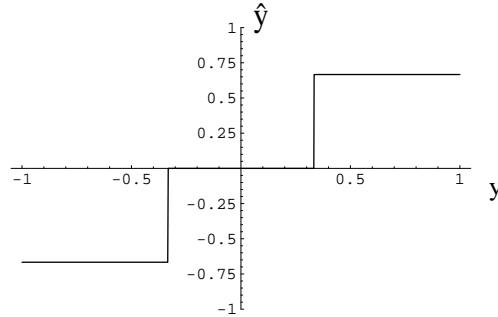


**Figure 2.** One-stage sigma-delta style A/D converter.

<sup>2</sup> We made Matlab movies of rotating feasible sets in 3-space corresponding to many filter types and complexities in order to make this observation. We accept the observation "as fact" because the *choice* of filter is not the main topic of our investigation. The ZOH is *not* on our list of "usable filters" because its corresponding feasible set is huge.

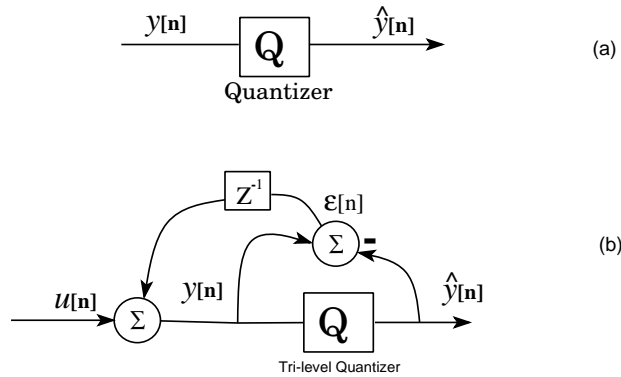
### 3. From An Open Loop Quantizer to a Sigma-Delta Modulator

The quantizer we use has the characteristic shown in Figure 3. We choose a tri-level quantizer rather than the predominant bi-level quantizer because the tri-level Q reduces the feasible set  $\{\hat{x}[n]\}$  to an intersection of slabs in hyperspace. This action is assumed superior to the long wedges of feasible set that would otherwise be produced using a bi-level (one-bit) quantizer.



**Figure 3.** Tri-level quantizer. Quantizer input signal assumed normalized to  $\pm 1$ .

In Figure 4 we focus our attention on the quantizer from Boyd's original problem shown back in Figure 1. The quantizer Q in Figure 4(a) and (b) implements the characteristic shown in Figure 3. Figure 4(a) is taken from Figure 1, while Figure 4(b) is from Figure 2. Figure 4(b) shows the quantization error feedback scheme whose purpose is to dither the input of the quantizer in a deterministic manner. [Gray] The purpose of the dither is to improve the low-level signal performance of the vanilla tri-level 1.58-bit quantizer Q.



**Figure 4.** (a) The 1.58 bit quantizer. (b) The Sigma-Delta Modulator using that quantizer.

#### 4.1. The Convex Optimization Problem

Referring to Figure 4(b), that is taken from Figure 2, we analyze the output produced by this nonlinear circuit starting from initial rest conditions.

$$y[0] = u[0]$$

The first quantized output is the same as having directly quantized the first sample in the  $u$ -input sequence. From our quantizer characteristic in Figure 3, then we know it must be true that

$$u[0] - \frac{1}{3} \leq \hat{y}[0] < u[0] + \frac{1}{3}$$

On the next sample cycle we have

$$\begin{aligned} y[1] &= u[1] + \varepsilon[0] = u[1] + (y[0] - \hat{y}[0]) = u[0] + u[1] - \hat{y}[0] \\ \Rightarrow u[0] + u[1] - \hat{y}[0] - \frac{1}{3} &\leq \hat{y}[1] < u[0] + u[1] - \hat{y}[0] + \frac{1}{3} \end{aligned}$$

Proceeding in a similar manner, we may analytically derive all subsequent quantized outputs,  $\hat{y}[n]$ , and come up with the general equation

$$\sum_{i=0}^n u[i] - \sum_{j=0}^{n-1} \hat{y}[j] - \frac{1}{3} \leq \hat{y}[n] < \sum_{i=0}^n u[i] - \sum_{j=0}^{n-1} \hat{y}[j] + \frac{1}{3}$$

from which we can write the more compact expression

$$\left| \sum_{i=0}^n (\hat{y}[i] - u[i]) \right| < \frac{1}{3} \quad (1)$$

Given only the observations  $\hat{y}[n]$  then, we can easily formulate Eq.(1) as an optimization problem by rewriting it in matrix form. Suppose that the vector  $u \in \mathbf{R}^N$ , and  $\hat{y} \in \mathbf{R}^N$ .

Then we can write

$$|R\hat{y} - Ru| < \frac{1}{3} \mathbf{1}$$

where  $R \in \mathbb{R}^{N \times N}$  is lower triangular;

$$R = \begin{pmatrix} 1 & & & \mathbf{0} \\ 1 & 1 & & \\ \vdots & \vdots & \ddots & \\ 1 & 1 & \dots & 1 \end{pmatrix}$$

Considering the vector  $u$  and the scalar  $t$  as the variables, then the optimization problem can be written

$$\begin{aligned} \min \quad & t \\ \text{s.t.} \quad & R\hat{y} - Ru \leq t\mathbf{1} \\ & -R\hat{y} + Ru \leq t\mathbf{1} \\ & t < \frac{1}{3} \end{aligned}$$

This is clearly a convex optimization problem as the objective function is affine and the constraints describe the feasible set as an intersection of half-spaces. This problem describes the circuit in Figure 4(b) (taken from Figure 2), and finds the vector  $\hat{u}$  that best describes the input signal  $u$  in a minimax sense over the feasible set. The problem expressed in this manner, however, always finds a solution that resides at a vertex of the feasible set which itself can be described as a hyper-dimensional polyhedron. Later, we will consider finding instead an interior point of this feasible set.

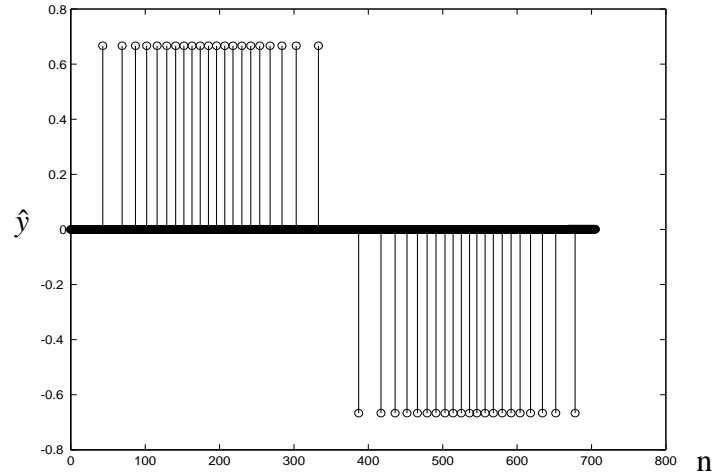
#### 4.2. Problem Formulation Incorporating the Upsampler

It is simple to extend this analysis to the complete circuit in Figure 2. Suppose now that  $x \in \mathbb{R}^N$ ,  $u \in \mathbb{R}^{NL}$ ,  $\hat{y} \in \mathbb{R}^{NL}$ , and  $R \in \mathbb{R}^{NL \times NL}$  remains lower triangular as before. The combination effect of the upsampler (L block) followed by a ZOH, of  $L$  samples in duration, is to introduce another matrix  $F \in \mathbb{R}^{NL \times N}$  into the analysis.  $F$  relates  $x$  to  $u$  through the equation

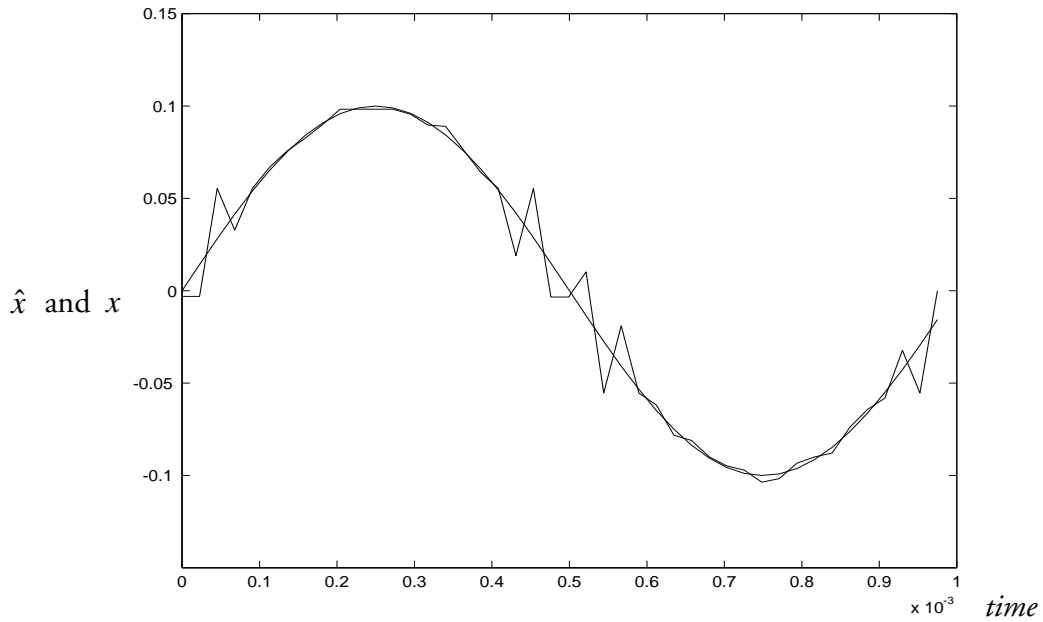
$$u = k_o F x$$

where  $k_o$  is the circuit's scalar gain constant.<sup>3</sup> The columns of this filter matrix  $F$  hold





**Figure 5.** Quantizer output in response to a 1000 Hz sine wave sampled at 44.1 kHz.



**Figure 6.** The jagged line connects the discrete solution  $\hat{x}$  found. Superimposed is the actual sinusoid  $x(nT)$  that we are trying to estimate.

The parameters for this problem are  $N=44$ ,  $L=16$ , sinusoid frequency=1000Hz, amplitude=0.1, phase=0, and  $F_s = 44.1$  kHz. The measured  $S/N$  is 15.864 dB. The traditional high-resolution distortion formula [Opp&Sch,ch.4] predicts

$$\frac{\mathbf{S}}{\mathbf{N}} = 10 \log_{10} \left( \frac{\text{amplitude}^2/2}{2^{-2B}/(12L)} \right)$$

at only about 3.344 dB where  $B = \log_2(3) - 1$ . The factor  $L$ , which improves  $\mathbf{S}/\mathbf{N}$ , comes from standard techniques of oversampling. [ibid.] This formula tends to be optimistic for low-resolution quantization such as we have here. Nevertheless, we should always try to beat this traditional rule of thumb as we have.

### 4.3. An Interior-Point Solution using a Newton Method

We next try an interior-point method of solution. For this part of the investigation we use a Newton-method algorithm, (fminu()) found in Matlab's Optimization Toolbox, that is based upon the BFGS technique.<sup>5</sup> All the data and parameters are the same as before. We use the Newton algorithm to perform an unconstrained optimization of the log-barrier function derived from the constraints of the LP, Eq.(2):

$$\begin{aligned} \phi(x, t) = & - \sum_{i=1}^{NL} \ln(- (R\hat{y} - k_o R F x - t\mathbf{1})_i) - \sum_{i=1}^{NL} \ln(- (-R\hat{y} + k_o R F x - t\mathbf{1})_i) \\ & - \ln\left(-\left(t - \frac{1}{3}\right)\right) \end{aligned}$$

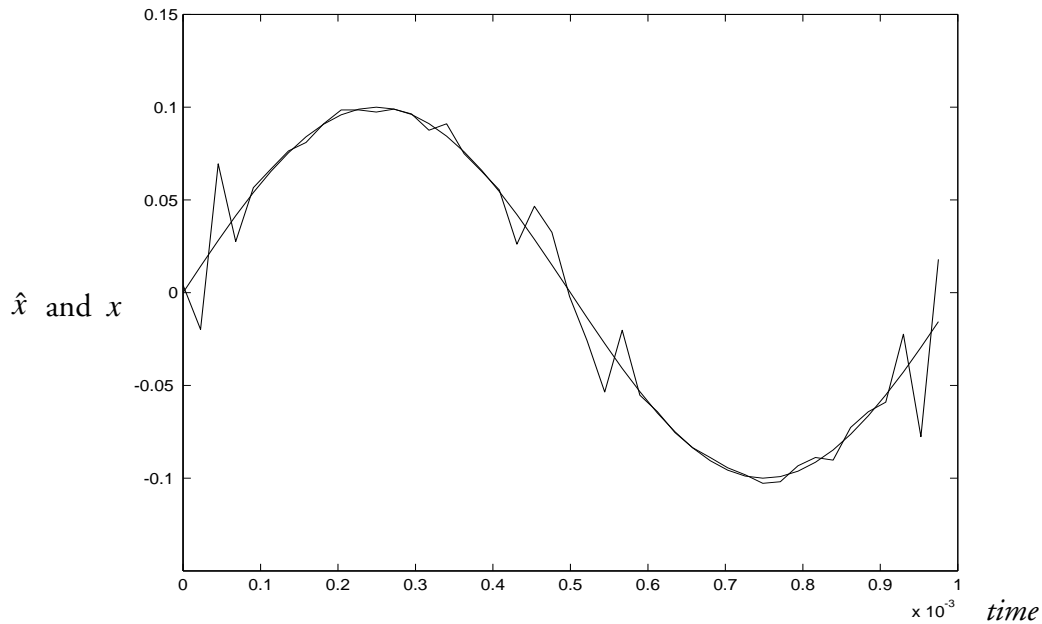
In order to use Matlab's Newton algorithm successfully, an expression for the gradient of the log-barrier function is required:

$$\begin{aligned} \nabla\phi(x)_j &= - \sum_{i=1}^{NL} \frac{k_o (RF)_{i,j}}{- (R\hat{y} - k_o R F x - t\mathbf{1})_i} - \sum_{i=1}^{NL} \frac{-k_o (RF)_{i,j}}{- (-R\hat{y} + k_o R F x - t\mathbf{1})_i} \quad ; j = 1 \dots N \\ \nabla\phi(x)_{N+1} &= - \sum_{i=1}^{NL} \frac{1}{- (R\hat{y} - k_o R F x - t\mathbf{1})_i} - \sum_{i=1}^{NL} \frac{1}{- (-R\hat{y} + k_o R F x - t\mathbf{1})_i} - \frac{1}{t - \frac{1}{3}} \end{aligned}$$

---

<sup>5</sup>The source code can be found in the Appendices.

We leave all the equations in this section without simplification so that they may be visually compared with Eq.(2).



**Figure 7.** The jagged line connects the discrete solution  $\hat{x}$  found via the Newton method. Superimposed is the actual sinusoid  $x(nT)$  that we are trying to estimate.

We choose the starting point for the Newton algorithm using the old results  $\hat{x}$  of the LP solution in the previous section. The Newton algorithm converged in 368 iterations. At convergence,  $t$  became minimized to the value 0.333223. Figure 7 compares the solution  $\hat{x}$  found by the Newton method, to the desired result  $x$ . The measured **S/N** at convergence is 13.758 dB; 2 dB worse than the previous LP solution.

## 5. Conclusions

One noteworthy observation made during the course of this investigation is that it seems unnecessary to agonize over the choice of filter  $H(z)$  in Figure 1, since the feasible set appears to be insensitive over a wide range of filter types and complexity. One must basically insure that the chosen filter's impulse response is dynamic, and excurses most of the filter's output signal range in  $L$  samples. We therefore believe that by replacing the ZOH with one of our "usable filters", mentioned in the Introduction, the corresponding feasible set would be reduced by an order of magnitude thus greatly improving our results.

We wish that we could say that the interior-point estimation is an improvement over LP, but instead it is about 2 dB worse for the particular data set and filter used in this experiment. The LP solution is itself better than the classical high-resolution prediction of noise power by about 12 dB. Further research is indicated because for this particular problem it is known *a priori* that the input signal vector, in fact, lies somewhere deep within the interior of the feasible set. New approaches might employ different barrier functions; e.g., reformulating our convex problem as an LMI employing a log-det barrier instead. We have tried substituting the inverse-barrier for the log-barrier and achieved about the same results.

## References

[Gray] Robert M. Gray, *Source Coding Theory*, Kluwer academic Publishers, 1990

[Opp&Sch] A.V. Oppenheim, R.W. Schaffer, John R. Buck, *Discrete-Time Signal Processing*, second edition, Prentice-Hall, 1999

## Appendix I. Gallery

Figure 8(a)...(.) is a gallery of visualization tools that we developed during the course of this project to investigate the characteristics of the A/D schemes in Figure 1 and Figure 2. Figure 8(a) (on the cover) shows the feasible set of  $Ax \leq b$  for two-dimensional data  $x$  and some random  $A$  and  $b$ .

Figure 8(b) and Figure 8(c) are taken from work on Boyd's original problem in EE263 (Figure 1). The most relevant observation here is that the least squares estimate of the two-dimensional input signal is consistently in error and outside the feasible set. We used SVD in the calculation of the uncertainty ellipsoid for the least squares estimate and found it to be equally un-illuminating. These results strongly motivate the use of other methods for estimation such as linear programming, that in fact yield better results.

Figure 8(d) shows the "error signal"  $y[n]$  from the sigma-delta modulator in Figure 2. This signal can be observed in real-time (somewhat) by running the Simulink file provided on the floppy.