Analog to Information

Convex Optimization for Minimized Sampling

CALTECH

CONTENTS

[TO UPDATE THE TABLE OF CONTENTS, RIGHT CLICK ON THE TABLE AND CLICK "UPDATE FIELD -> UPDATE ENTIRE TABLE"]

1 Outline of Research Accomplishments

One of the central tenets of signal processing and data acquisition is the Shannon/Nyquist sampling theory: the number of samples needed to capture a signal is dictated by its bandwidth. Very recently, we have developed an alternative sampling or sensing theory which goes against this conventional wisdom. This theory now known as "Compressed Sensing" or "Compressive Sampling"—or CS for short—allows the faithful recovery of signals and images from what appear to be highly incomplete sets of data, i.e. from far fewer data bits than traditional methods use.

While CS is a very recent discovery, there is little doubt that it has the potential for considerable impact in many fields of science and technology. For instance, CS may come to underlie procedures for sensing and compressing data simultaneously and much faster (or in other words, one could translate analog data into already compressed digital form). In practice, this means that one could obtain super-resolved signals from just a few sensors. It goes without saying that this situation raises tantalizing opportunities. In medical imaging, radiologists recognize that CS will enable new technologies such as high-speed magnetic resonance angiograms; in fuel cell research, CS will empower state of the art imaging technologies. In the area of analog-to-digital converter (ADC) technology, CS may come to enable revolutionary advances in data conversion, which is the subject of this program.

During the whole length of the program, Caltech and Northrop Grumman have worked in close collaboration, and led a *concerted* and *focused* effort which demonstrates that Compressive Sampling can indeed help addressing enormous challenges in the acquisition and processing of ultra wideband radio frequency signals. By way of background, recall that the classical or standard Shannon theory asserts that to sample a signal with a given frequency bandwidth, one needs to digitize the signal at a sampling rate which is at least twice the bandwidth. This is extremely problematic since for ultra wideband signals, high speed ADC technology indicate that current capabilities fall well short of needs, and that hardware implementations of high precision Shannon-based conversion seem out of sight for decades to come. In short, conventional thinking seems to have hit a brick wall. In application areas, this has some significant implications. For instance, Shannon based analog-to-digital conversion limits the ability of systems to cope with evolving threats.

Against this background, the focus of our concerted effort has been the design of two novel ADC architectures which exploit the assumed structure of the signal we wish to acquire in order to dramatically reduce the sampling rate and enable new approaches to data conversion. Hence, our work directly addresses the major challenges set forth in the Broad Agency Announcement. Namely, our work "enables practical data conversion approaches which more effectively apply system resources to find the useful information content embedded in a complex RF environment and directly measure it in a more concentrated form than is possible in currently." To cut a long story short, we designed and constructed hardware prototypes directly inspired by the theory of Compressive Sampling and demonstrated that in the laboratory, one could achieve the gains which were theoretically

anticipated. On top of this achievement, our experiments show that one can reliably acquire signals which are in principle far outside of the range of current data converters. That is, we believe that if our prototypes were to scale up as expected, then one would hold a new generation of ADCs with sensing capabilities way beyond the brick wall discussed earlier. To summarize, our effort is a first and extremely encouraging step in transforming an enormously promising mathematical theory into effective new hardware.

1.1 Technical contributions

The remainder of this report will provide a full account of our accomplishments, of our technical results and of the various metrics which we have used to assess performance. Before getting into a detailed exposition, however, it might be best to begin by giving a general overview. During this last year, our research has essentially focused on three fronts: 1) the design and fabrication of a differential non-uniform architecture; 2) the design of a random pre-integration architecture; and 3) the study of how these architectures perform in real world applications, especially in the context of GSM-based communications. We briefly outline each of these.

- *A differential non-uniform sampling architecture.* First, we have designed a non-uniform analog-to-digital converter, simulated its expected performance, and constructed a prototype able to produce real data. In a nutshell, our non-uniform sampler (NUS) digitizes the signal at randomly sampled time points. In effect, these random or pseudo-random time points are obtained by jittering nominal (low-rate) sample points located on a regular lattice. The rationale behind this architecture is that the theory of Compressive Sampling asserts that this data acquisition strategy allows the reconstruction of wideband frequency signals with no information loss provided that the spectrum of the signal under study is sufficiently sparse. In the situation where the spectrum is only approximately sparse, one can reconstruct the full signal with very low distortion.

  There are of course tremendous benefits associated with a reduced sampling rate as this provides added circuit settling time. In addition, the longer time for S/H settling has the effect of reducing the noise level. Of concern, however, is whether all the circuit's nonidealities, such as thermal noise, sample jitter due to clock timing errors, pattern noise due to the use of a non-uniform clock, quantization feedback inter-symbol, and amplifier nonlinearities cause serious problems for the reconstruction. To investigate all these effects, we have run a series of extensive simulation studies showing that all these nonidealities behave like additive Gaussian noise. Because we also developed noise aware reconstruction algorithms, our finding shows that with realistic error models, the quality of the reconstruction, e.g. the effective number of bits, smoothly degrades as the various effects get amplified. That is, the effect of all these errors is the same as adding noise with the same power. In truth, we did not expect such a remarkable performance prior to this study.

  Further, we also developed a prototype sampler able to digitize real analog signals and tested our methods and algorithms on real data. Of major concern is the fine

tuning of algorithms and extensions of our theories and methods to include continuous time signals (all of the existing CS theory is exclusively about finite discrete time signals). Especially, methods which extend as much as possible the dynamic range of signals we wish to detect and recover are of special interest. The upshot here is that early experiments with the prototype demonstrate performance for real world applications. For instance, our NUS architecture combined with our reconstruction algorithms reproduce very high frequency signals even though these signals are sampled at a rate which is way below the Nyquist rate.

In conclusion, our experiments suggest that the potential impact of CS on ADC technology is real. For instance, one can assess these implications in the context of the so-called Walden's curve. Roughly speaking, Walden observed that the performance limits of current analog-to-digital converters have three regimes determined by whether the sampling rate is low, intermediate or high. In each region, there is a curve bounding the accuracy of an ADC (the number of effective bits) as a function of the sampling rate; the higher the rate, the fewer the number of effective bits. Note that there is a brick wall at about 1GHz which says that it almost impossible to design an accurate ADC operating near that speed. The feasible region is the set of all ADCs obeying Walden's observation, those with a prescribed maximum number of effective bits for a given sampling rate. In this context and if one is interested in the acquisition of signals with a reasonably sparse frequency spectrum, then one can use CS to significantly extend the achievable region beyond the Walden curve. First, for moderate and intermediate sampling rates, one can obtain effective numbers of bits which were perhaps thought as unattainable. And, second one can extend the achievable region past the ambiguity curve (past the brick wall) and digitize ultra wideband signals with reasonable accuracy.

- *A random pre-integration architecture.* We studied another CS architecture for digitizing analog signals which we dubbed RPI for random pre-integration. Whereas the NUS architecture is in some sense optimal for signals with a sparse frequency spectrum, the RPI architecture is adapted to a wide variety of sparsity domains. For instance, the RPI architecture is extremely well suited to digitize very high frequency pulses with all kinds of shapes. More broadly, the RPI architecture is nearly ideal for signals having a sparse signature in the time-frequency plane.

  Whereas it may not be possible to digitize the analog signal at a very high rate rate, it may be quite possible to change its polarity at a high rate. The idea of the RPI architecture is then to multiply the signal by a pseudo-random sequence of plus and minus ones, integrate the product over time windows, and digitize the integral at the end of each time interval. This is a parallel architecture and one has several of these random multiplier–integrator pairs running in parallel using distinct or event nearly independent pseudo-random sign sequences. In effect, the RPI architecture correlates the signal with a bank of sequences of +/-1, a sensing mechanism known in the theory of CS to enjoy remarkable properties in terms of the low number of measurements needed for a given "information rate."

Just as for the NUS architecture, a main concern has been whether our proposed architecture and reconstruction algorithms are robust vis a vis hardware imperfections. We then developed an error model and performed extensive simulation studies to assess the impact of the three main sources of uncertainty; namely, thermal noise, timing jitters (random and patterned) which has the effect of only approximately knowing the location of the sign switches, and cross-talks between channels. The conclusion of these studies is that all three sources of imperfection behave like thermal noise. That is to say, the recovery error closely tracks the measurement errors. In other words, one witnesses the same kind of robust performance as in the case of the NUS architecture. Again, this is extremely encouraging.

We then demonstrated the potential of this compressive sampling implementation concept via tests and simulations. Of special interest here is the acquisition of sparse pulse trains which possibly overlap. Our simulations include a realistic integrator model together with a realistic error model (SPICE). Here, much work was needed on the methodological side to obtain the highest possible performance. In a nutshell, developments included the development of a general bank of waveforms in which the assumed sparsity is expressed, the development of improved and innovative processing algorithms, and the development of alternatives to l_1-minimization for enhanced accuracy. All of our latest results indicate a very significant potential impact, should this architecture ever be implemented.

Despite the enormous promise of this architecture, we did not develop an RPI test bed because of budget constraints.

- *GSM detection & decoding from subNyquist sampling.* To test the effectiveness of the NUS architecture in a real world application and in order to develop metrics comparing CS and conventional sampling, we considered a simulated communications scenario involving a GSM cellular system. The setup is roughly as follows: a 20MHz frequency band is divided into 100 channels of bandwidth 200kHz each. During a single time slot, some of these 100 channels may contain encoded message packets. The binary messages are encoded Gaussian minimum-shift keying (GMSK) and then modulated up to the appropriate carrier frequency in the GSM band. If in a single time slot, only a few channels contain encoded information, then the signal of interest has a sparse spectrum and falls well in the range of the capabilities of the NUS architecture.

  We conducted experiments both with simulated data and with NUS test bed data. In the experiments with simulated data, we constructed GSM spectra with varying numbers of active channels. Taking nonuniform samples of the composite signal, we were able both to estimate the unknown carrier frequencies positions of the active channels and furthermore to invert the measurement process on these channels alone. Ultimately we were able to recover the GSM pulses on the active channels using a total nonuniform sampling rate proportional only to the 200kHz bandwidth of each

active channel times the number of active channels. We also found these experiments to be robust to noise, with a predictable degradation as the noise level increases.

In the experiments with NUS test bed data, we constructed spectra containing small numbers of active GSM channels (at known carrier frequencies). We again were able to recover these signals with low bit error rates using a total nonuniform sampling rate proportional only to the 200kHz bandwidth of each active channel times the number of active channels, and again we found these experiments to be robust to noise. We also conducted additional experiments where the spectrum contains strong interferers (TV signals) at known frequencies outside the GSM bandwidth. For these experiments we were able to again recover the GSM pulses (by inverting the measurement operator over both the GSM channels plus the TV bandwidth); in this case the requisite sampling rate was proportional to the unknown GSM bandwidth plus the TV spectrum bandwidth.

In summary, we found in each of these experiments that the necessary nonuniform sampling rate derived from the information level (the bandwidth) of the unknown portions of the spectrum, rather than the total bandwidth of the spectrum, and that the recovery process was robust to noise. Each of these conforms to the general performance predicted in CS theory.

Clearly, our research addresses many of the goals outlined in the BAA. First, we are enabling practical data conversions approaches which are far more effective than traditional converters. Second, we have demonstrated in the laboratory the power of an explicit sensing mechanism, namely, the NUS architecture. Based on this experience on the one hand and on the numerical studies of the RPI architecture on the other, we are confident that a hardware implementation of the RPI architecture will also eventually meet our expectations. We are of course fully aware of both the technical challenges and the considerable amount of work which lie ahead. And third, we have deployed our prototype and algorithms to deal with important real world situations with significant success.

1.2 Looking ahead

Our team has accumulated a significant amount of evidence pointing in a clear direction: future CS-based systems are very likely to work. Looking ahead, this provides a unique opportunity to change current approaches to data conversion. Energized by this success, Caltech and Northrop Grumman already have a plan to move forward. Just as our past work has had a lot of focus, our plans for the future are concentrated around very specific and concrete goals:

- The design and hardware implementation of high-speed non-uniform sampling architectures operating in the GHz range; that is, able to acquire signals with frequency content up to 10 or 20 GHz.

- The hardware implementation of one (or several) random pre-integration architecture; the target is the acquisition of pulse-like signals at extremely high frequencies, again with content up to 10 or 20 GHz.

Clearly, our aim is to make real systems which could change the way one views RF signal acquisition. Making this possible, however, would require assembling a team with a broader set of skills and expertise. For instance, one would have to address the problem of implementing hardware architectures capable of processing massive amounts of data and producing signals of full length in reasonable time. Our team calls COMS the process of taking as input undersampled data and returning a full length signal, a shorthand for convex optimization with minimal sampling. One can thus complement our list with a third additional goal:

- The design and efficient hardware implementation of data processing architectures for COMS.

In light of this last goal, it is clear that in addition to engineers with a specialty in ADCs, and applied mathematicians with a specialty in signal processing, large scale computations, and Compressive Sampling, we would also need expertise in large scale convex optimization and in programmable analog or digital signal processing. Now our team has already taken steps in this direction.

First, Caltech has started a collaboration with the group led by Professor Stephen Boyd from Stanford University. Stephen Boyd is a world expert in optimization and together with Michael Grant, a member of his research team, we have started to exchange promising ideas for speeding up COMS. Second, Caltech has engaged Paul Hasler, a Professor at the Georgia Institute of Technology. Paul Hasler is working on relieving the enormous pressure put on DSP units. His work shows how the range of analog signal processing functions available results in many potential opportunities to incorporate these analog signal processing systems with digital signal processing systems for improved overall system performance. Professor Hasler believes that his "analog thinking" could contribute significantly to COMS.

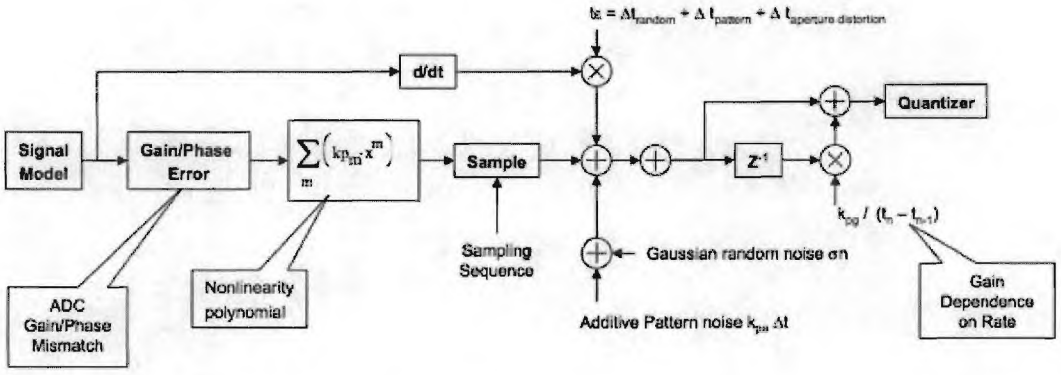
Both Professor Boyd and Professor Hasler are extremely interested in our project and have indicated that they would be eager to join our team, should we receive additional funding and move forward.

2 Non-Uniform Sampling

2.1 Numerical Modeling and Sensitivity Analysis

The non-uniform sampling architecture is shown in Figure 1. We will examine the stability of COMS in the face of four hardware non-idealities: additive thermal noise, sample location jitter, quantization, and nonlinear gain. The goal is to show that despite their

somewhat specialized nature, none cause pathological errors in the COMS reconstruction. We isolate and judge each of these in turn.



$\Delta t_{random} = 10^{-14}$ s rms Gaussian random jitter

$\Delta t_{aperture\ distortion} = k_{ad} \times$ signal

$\Delta t_{pattern} = k_{pj1} / (t_n - t_{n-1})$

σn: Additive random noise (2Qrms)

k_{pa}: Additive pattern noise due to non-uniform clock (kpa=3x10^{4})

K_{pg}: Gain rate dependance coefficient (10^{-13})

kp_m: nonlinearity polynomial coefficient (kp_1=1, kp_2=6x10^{-5}, kp_3=4x10^{-4})

K_{ad}: aperature distortion coefficient (10^{-14})

K_{pj1}: pattern jitter coefficient for dependancy on time since last change (10^{-22})

Figure 1: Non-uniform sampling architecture model for sensitivity analysis

2.1.1 Numerical Model

We simulate the system for an input signal which is bandlimited and periodic with fundamental interval [0,T]. We can expand the input $x(t)$ using the Fourier Series

$$x(t) = \sum_{\omega=0,\ldots,N-1} \hat{c}(\omega) e^{j2\pi\omega t/TN}$$

where the maximum possible frequency is $N/2T$. The M sample locations are given by $0 \le t_1 < t_2 < \cdots < t_M \le T$. We can interpolate to arbitrary sample locations by using $x(t_m) = \sum_{\omega} \hat{c}(\omega) e^{j2\pi\omega t_k/TN}$.

This makes the recovery problem finite dimensional: we want to reconstruct each of the N Fourier coefficients (most of which will be zero if the signal is spectrally sparse) from the M sample measurements. We will denote the mapping from a set of Fourier coefficients to the samples on $\Gamma := \{t_1,\ldots,t_m\}$ with the MxN "measurement matrix" A.

We recover the signal in two stages. Given the M vector of measurements y, the first stage consists of solving the linear program

$$\min_{\hat{d}} \sum_{\omega} |\hat{d}(\omega)| \quad \text{subject to} \quad \left\| A^T(A\hat{d} - y) \right\|_{\infty} \le \varepsilon$$

for an appropriate value of ε (usually close to 2σ, where σ is the standard deviation of the noise in the measurements). Setting Ω to be the support of the solution to the above (the

frequencies with non-zero components), the second stage consists of a least-squares projection onto this discovered support:

$$\tilde{d}_\Omega = (A_\Omega^T A_\Omega)^{-1} A_\Omega^T y, \qquad \tilde{d}(\omega) = 0, \text{ for } \omega \notin \Omega.$$

Here, the $M \times |\Omega|$ matrix A_Ω is formed by taking the columns from A whose indices are in Ω. In situations where all of the frequency components are well above the noise floor, the first stage will almost always recover the correct support. This was the case in all the experiments here, as we are primarily interested in high SNR applications.

2.1.2 Thermal Noise

Thermal noise is modeled by adding additive white Gaussian noise to the measured samples. If the support is discovered correctly in the first state, the expected mean-square error in the recovery is

$$\mathrm{E}\left\|\tilde{d} - \hat{c}\right\|_2^2 = \mathrm{Trace}\left((A_\Omega^T A_\Omega)^{-1}\right) \cdot \sigma^2$$

where σ^2 is the noise variance. In previous publications, the authors have developed uncertainty principles which tell us in effect that

$$\mathrm{Trace}\left((A_\Omega^T A_\Omega)^{-1}\right) \approx \frac{N}{M} \cdot S$$

where S is the number of non-zero frequency components: $S = |\Omega|$.

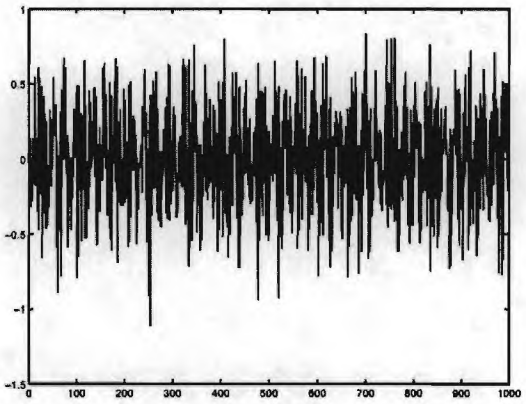
The extent to which the experiments match this theory is striking. Consider a particular case where N=1000, M=110, and S = 20. We will measure the signal-to-noise ratio in terms of effective number of bits:

$$\mathrm{enob} = \frac{1}{2}\log_2\left(\frac{\mathrm{E}\|\mathrm{signal}\|_2^2}{\mathrm{E}\|\mathrm{error}\|_2^2}\right) + 1.$$

For the signal shown in Figure 2, we choose the noise variance so that the enob for the Nyquist samples is 6 bits. In this case we expect the enob for the recovery to be

$$6 + \frac{1}{2}\log_2\left(\frac{M}{S}\right) \approx 7.2 \text{ bits.}$$

In our experiments, we observe a recovery error of about 7.1 bits (see Figure 3). This result is typical for the thermal noise results and the other sources of error; compressive sampling seems to come within 0.1 to 0.2 bits of the ideal enob.



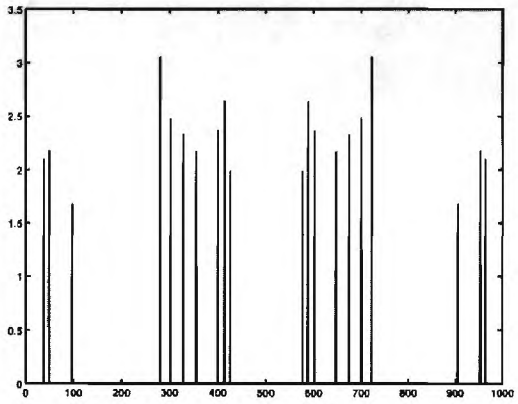


Figure 2: Time and Frequency Domain Plots of the Multi Tone Signal Used for Simulating Effects of H/W Model Error Sources

Figure 3 shows the recovery error versus Nyquist enob for different values of M and σ^2. Notice that the recovery enob tracks the input enob on a straight line of slope 1. This is the exact same behavior we would observe if the recovery process simply consisted of inverting an orthogonal transform, even though the COMS recovery process is exceptionally nonlinear.

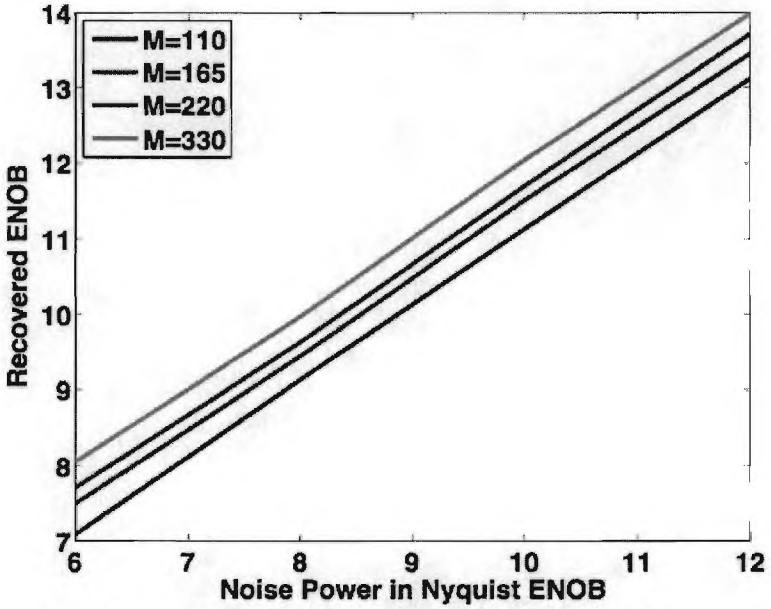


Figure 3: Predicted Effect of Thermal Noise on Recovered ENoB

2.1.3 Sample Location Jitter

Unlike thermal noise, the perturbations caused by timing jitter are signal dependent. Nonetheless, our experiments show that to COMS, it is essentially AWGN of the same variance.

To model random timing jitter, we set
$y_k = f(t_k + \delta_k), \quad k = 1,\ldots,M$
where the δ_k are independent Gaussian random variables with variance σ^2. We can derive the expected noise power introduced by timing jitter. The noise power will of course depend on the frequency content of the signal being sampled. The Fourier transform of the error $f(t_k + \delta_k) - f(t)$ is then $X(\omega) = F(\omega)(e^{j\omega\delta_k} - 1)$. The variance for a single frequency component is then

$$\mathrm{E}[X(\omega)X^*(\omega)] = |F(\omega)|^2 \,\mathrm{E}[2 - e^{j\omega\delta_k} - e^{-j\omega\delta_k}] = 2|F(\omega)|^2 \left(1 - e^{-\omega^2\sigma^2/2}\right).$$

When $\omega\sigma$ is small, $1 - e^{-\omega^2\sigma^2/2} \approx \omega^2\sigma^2/2$, and so

$\mathrm{E}[|X(\omega)|^2] \approx |F(\omega)|^2 \omega^2 \sigma^2$.

Thus if the input is a single complex sinusoid the expected measurement SNR will be

$$\mathrm{SNR} = \frac{1}{\omega^2 \sigma^2}$$

In general, the mean-square jitter error will be the sum of the $\mathrm{E}|X(\omega)|^2$ above over all the frequency components in the signal.

The question is now whether the jitter error looks any different to the COMS reconstruction than additive white Gaussian noise with the same SNR. The answer, as shown in Figure 4 below, is that it does not. Just as in the thermal noise experiments, the enob in the recovery from jittered samples tracks the measurement enob on a straight line, so we can conclude that the signal dependent nature of the jitter error does not cause any unforeseen problems.

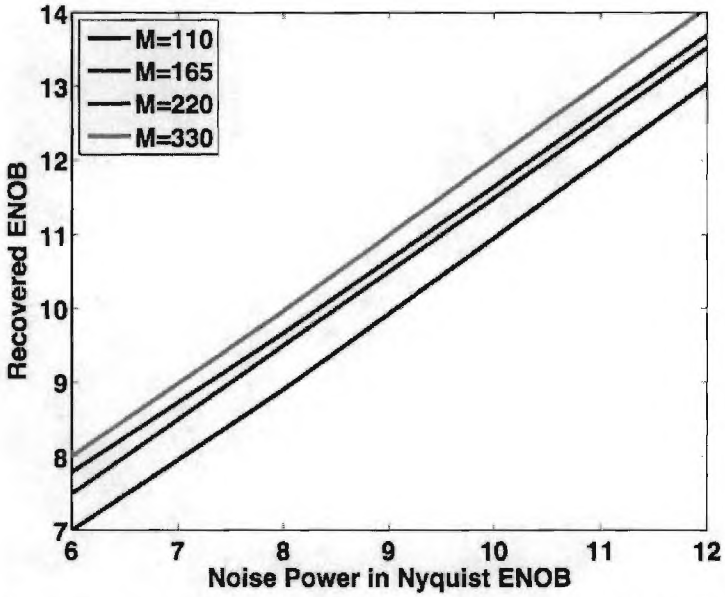


Figure 4: Predicted Effect of Jitter on Recovered ENoB

There are two other sources of timing jitter: patterned jitter which depends on the differences between the sampling locations, and aperture distortion, which is caused by interference from the signal itself. In practice, we expect the errors caused by these two sources to be quite small. Regardless, in simulating both types of these types of jitter we observed that they behaved (as far as reconstruction error is concerned) just like the random jitter of comparable power described above.

2.1.4 Quantization

As expected, the errors produced by quantization also did not cause any problems. An quantizer with step size q essentially behaved as a additive white Gaussian noise with per-sample variance of $q^2/12$.

We verified that this statement stays true when the quantization error is patterned. We modeled this feedback error by having the simulated quantizer obey

$$x^q(t_k) = \text{Quantize}\left(x(t_k) + \frac{C}{t_k - t_{k-1}} \cdot x^q(t_{k-1}) \right),$$

where C is a constant which controls the magnitude of the feedback, and $x^q(t_k)$ is the quantized value of the signal at time t_k. This modification to the model did not change things; the recovery error still behaved as it does in the face of AWGN with variance $q^2/12$.

2.1.5 Nonlinearity Polynomial

Unlike our other hardware nonidealities, the error caused by a polynomial gain does not look random at all. The reason for this is not hard to see, especially when the input signal is a trigonometric polynomial. If $x(t)$ is the superposition of a few sinusoidal components, then the output of the polynomial gain element

$$x_p(t) = x(t) + a_2 x^2(t) + a_3 x^3(t)$$

will also be the superposition of a small number of sinusoidal components. Thus the residual is spectrally sparse, and COMS cannot distinguish it from the true signal. Worse, the error is concentrated on the support of the original signal; an example is shown in Figure 5.

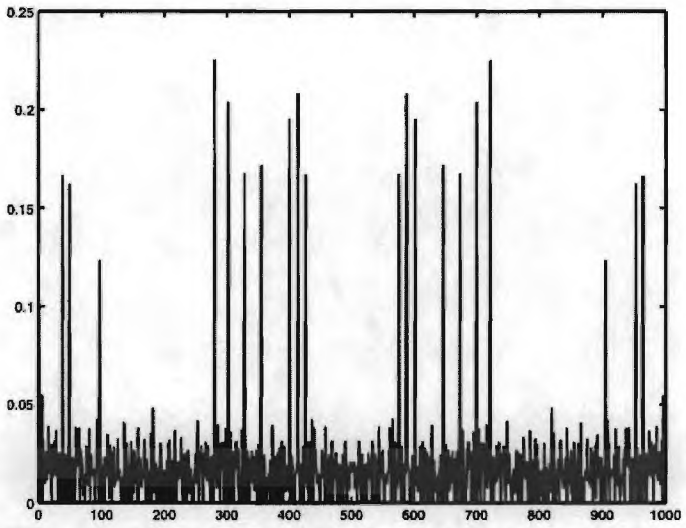


Figure 5: Residual error caused by a polynomial gain is spectrally sparse

We adjusted the gain coefficients a_2 and $a_3 = 6.67a_2$ so that the enob of the Nyquist samples of the distorted signal was 6, and recovered using COMS for different numbers of samples.

As we can see from the plot in Figure 6, adding samples does not help the recovery after a point, as the error itself has the same structure as the signal.

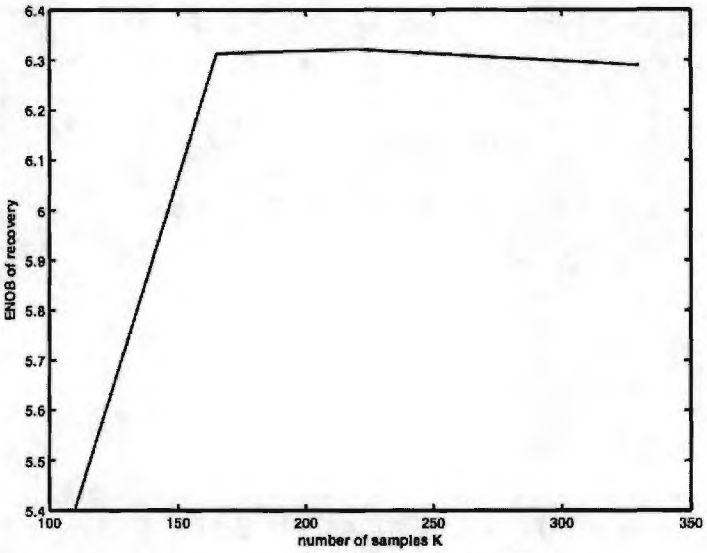


Figure 6: Oversampling only helps up to a point with polynomial distortion

Fortunately, we expect the size of this error to be very small in an actual hardware implementation.

2.1.6 Windowing for "off the grid" signals

[THIS WAS A FRAGMENT IN JUSTIN'S DOCUMENT... DO WE WANT TO SAY SOMETHING HERE?]

2.2 Undersampled A/D and the Walden Curve

From the simulation results in the previous section, we can begin to see the potential impact of compressive sampling on ADC technology. In this section, we discuss this potential impact in the context of the so-called "Walden Curve".

In [Walden, 1999], Walden surveyed a collection of analog-to-digital converters, and concluded that their performance was limited by three main factors:

- **Thermal noise.** The faster the ADC runs, the more the input signal is subject to various types of noise. The net effect of these various perturbations can be modeled as additive white Gaussian noise whose variance is proportional to the sampling frequency. Under standard operating conditions, thermal noise limits the effective number of bits by

  $$b_{therm} = 31.56 - \frac{1}{2}\log_2 R_{eff} - \frac{1}{2}\log_2 F_{samp},$$

  where R_{eff} is an "effective resistance" for the noise sources in the integrated circuit,

and F_{samp} is the sampling frequency at which the ADC is operating. Note that doubling the sample rate costs us 0.5 effective bits.

- **Aperture jitter**. As discussed in the previous section, the locations at which the ADC takes samples are also subject to some uncertainty. If σ_{jit}^2 is the variance of the sample location uncertainty, aperture jitter limits the effective number of bits by

$$b_{jit} = -3.44 + \log_2\left(\frac{1}{\sigma_{jit}}\right) - \log_2 F_{\max}$$

where $F_{\max}$ is the highest frequency (expressed in Hertz) in the input signal. The jitter impairment is signal dependent; doubling the maximum frequency in the input will cost us 1 effective bit.

- **Comparator Ambiguity**. Ultimately, the maximum speed at which the ADC can run is determined by the process used to fabricate the integrated circuit. The limit on the effective number of bits brought on by the comparator ambiguity is given by

$$b_{amb} = 0.453 \cdot \frac{f_T}{F_{samp}} - 1.1,$$

where f_T is a parameter that characterizes how quickly the transistors in the circuit can react to small change in voltage. Note that b_{amb} differs dramatically in form from b_{therm} and b_{jit}; the ambiguity error grows exponentially with the sampling frequency.

To make things concrete, we will use $R_{eff} = 50$ ohms, $\sigma_{jit} = 20$ femtoseconds, and $f_T = 250$ GHz. The corresponding Walden curve (assuming that we are sampling at the Nyquist rate $F_{samp} = 2F_{\max}$) is shown in Figure 7. The region bounded above by these three functions is the set of effective-number-of-bits (enob) / maximum-frequency-content performance points that we can achieve using standard Nyquist sampling. As an example (correspond to the yellow dot in Figure 7), we could easily expect around 10 effective bits from a state-of-the-art ADC for a signal bandlimited to 5 GHz and sampled at 1 GHz.

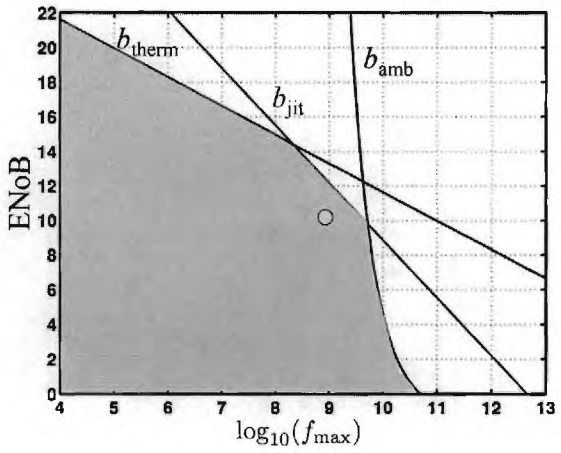


Figure 7: Walden Curve

If the signal is spectrally sparse, the number of effective bits can be boosted using some simple signal processing. Thermal and jitter noise can be mitigated by "pulling out" (selectively filtering) only the frequencies in which we are interested. If F_{cont} is the total frequency content of the signal, this post-processing would yield a gain of

$$\text{Sparsity gain} = \frac{1}{2}\log_2 \frac{F_{\max}}{F_{cont}} \quad \text{bits.}$$

The revised Walden Curve for $F_{\max}/F_{cont} = 100$ (the signal only occupies 1% of the spectrum between DC and $F_{\max}$) is shown in Figure 8; the thermal and jitter noise lines have moved up by 3.32 bits.

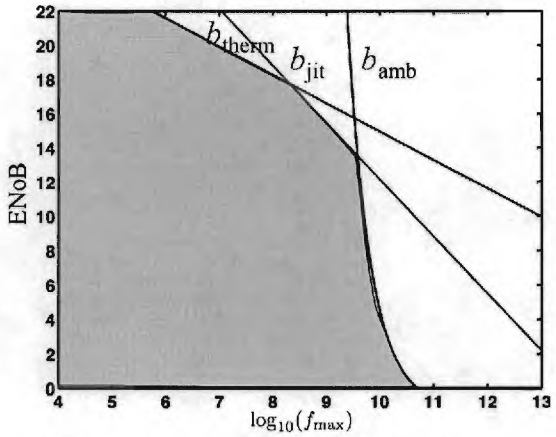


Figure 8: Revised Walden curve with shifted thermal and jitter noise lines

Sparsity also allows us to extend the feasible region past the ambiguity curve by using compressive sampling. We have seen that in practice, a signal with S Fourier modes can be reconstructed from only $M \geq 5S$ of its N Nyquist samples (on an interval [0,T]) with an effective number of bits very close to

$$b_{CS} \approx b_{nyq} + \frac{1}{2}\log_2 \frac{M}{S},$$

where b_{nyq} is the effective number of bits for the limiting noise factor --- $b_{nyq} = b_{jit}$ for high-frequency input signals, and $b_{nyq} = b_{therm}$ when the ADC is running more slowly. The currently unachievable region is dominated by jitter noise; at these input frequencies, our enob will be approximately

$$b_{CS} \approx -3.44 + \log_2\left(\frac{1}{\sigma_{jit}}\right) - \log_2 F_{\max} + \frac{1}{2}\log_2 \frac{F_{samp}}{F_{cont}},$$

where we have translated "samples per period" into "sampling rate". (For COMS to be effective, the samples need to be uniformly spaced. We can interpret F_{samp} as the average sampling rate. We are going to make the assumption that the curve bounding the feasible region is the same for an ADC that takes non-uniformly spaced samples is the same as in the

equispaced case with comparable rate.) The new feasible region is shown in

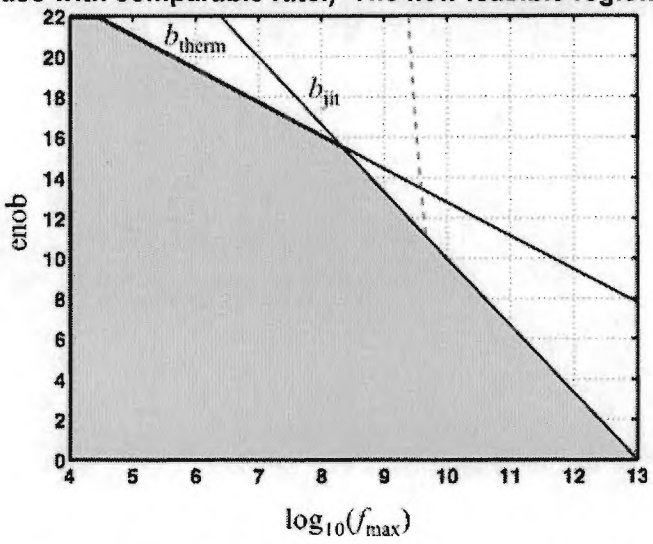


Figure 9; for the parameter values used there and for $F_{\max}$ large, the above will hold for $5F_{cont} \le F_{samp} \le 10$ GHz.

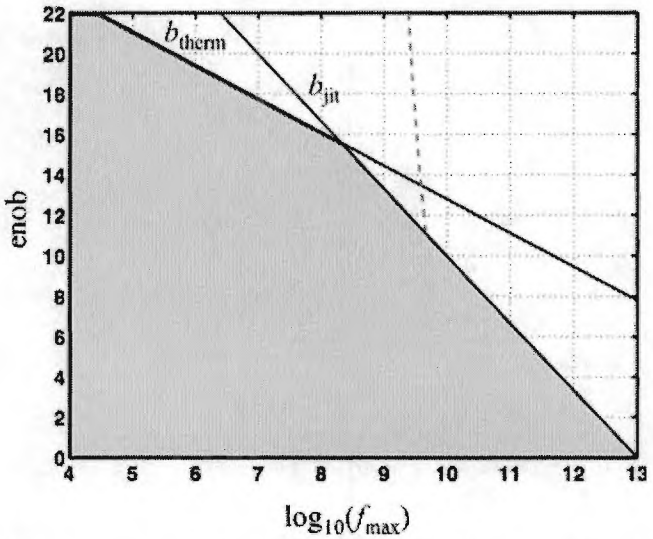


Figure 9: Revised Walden curve for nonuniform sampling followed by COMS reconstruction (with $F_{samp} >= 5F_{cont}$). The feasibility region extends past the comparator ambiguity curve, making signal acquisition possible for signals with very high frequency content.

For a graphical interpretation, we see from Figure 7 that there exist ADCs that yield 10 effective bits at a sampling rate of 1 GHz. Of course, we can use this ADC at lower sampling rates, and the effective number of bits should increase along a line that has slope 1 (the same slope at the line for b_{therm}). The achievable enob for different operating rates is shown by a yellow region in Figure 10 (left panel). Using COMS, the feasible region continues to the right along a line of slope 1 until the reconstruction error becomes jitter limited, after which the feasible region follows a line of slope 2 downwards. The extended yellow region in Figure 10 (right panel) illustrates this. Since the ADC is operating at a

constant rate (1 GHz) for every point in this region, comparator ambiguity does not limit the performance.

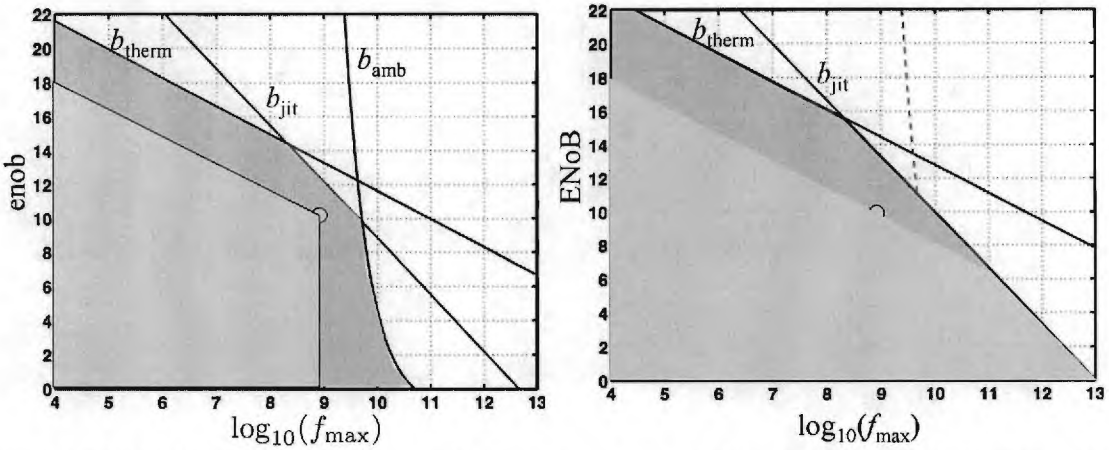


Figure 10: Left: By changing the sampling speed on a traditional ADC (in proportion to the maximum frequency in the input) we can achieve any point in the yellow region. Right: Using COMS, we can achieve any point in the extended yellow region without changing the sampling rate (as long as $F_{samp} >= 5F_{cont}$).

2.3 GSM Case Study

2.3.1 Simulation

To test the effectiveness of recovering sparse signals from nonuniform samples, we consider a simulated communications scenario involving a GSM cellular system.

For our simulation, we consider a 20MHz frequency band (from 2MHz to 22MHz) divided into 100 channels of bandwidth 200kHz each. During a single time slot (approximately 500 microseconds), some or all of these 100 channels may contain encoded message packets. Each message packet has a data rate of 270.8333 kbps. The binary messages are encoded Gaussian minimum-shift keying (GMSK) with bandwidth parameter 0.3 and then modulated up to the appropriate carrier frequency in the GSM band.

Figure 11 shows the basic setup for a prototypical GSM receiver: in this conventional setting, the GSM signal would be sampled at least 2x higher than the 20MHz bandwidth of the GSM band. These high-rate uniform samples would then be used, for example, to determine which of the 100 channels were active during the transmission (which should be apparent just by taking an FFT of the samples), and then the samples could be demodulated down to from each desired carrier frequency, lowpass filtered, and GMSK decoded.

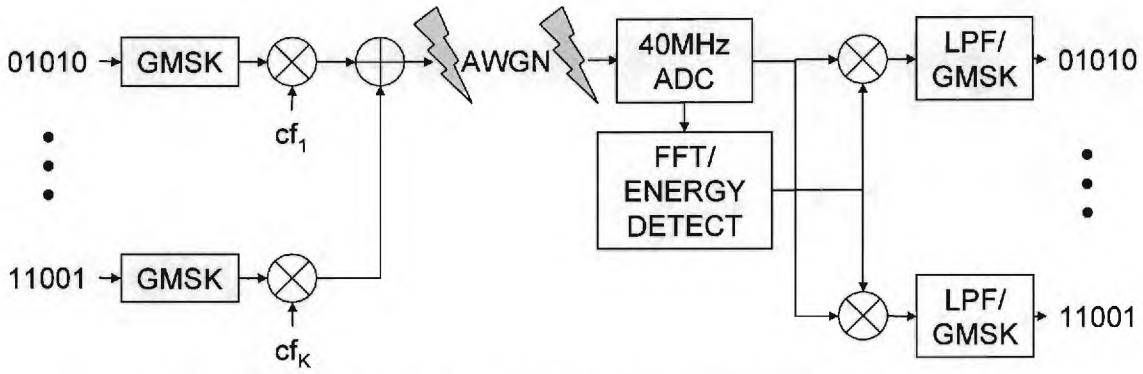


Figure 11: GSM Detection & Decoding from Full-rate Nyquist Sampling

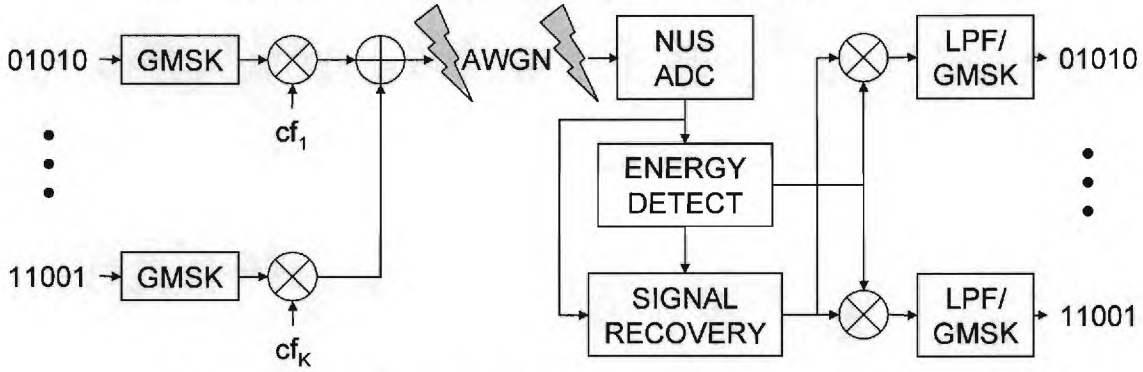


Figure 12: GSM Detection & Decoding from N-U Low-rate Sampling

We will examine an alternative receiver, as shown in Figure 12. In this setup we take only a small number of nonuniform samples of the GSM signal. From these samples we must perform the same tasks: determine the locations of the active channels, and decode the packets corresponding to those specific channels. Our methodology is as follows:

Active Channel Detection: Supposing we have M samples from a length-N discrete signal, we estimate the signal's Fourier transform by creating a length-N signal that contains all zeros except for the M sample values at the proper times. We take the FFT of this length-N signal, and then create 100 test statistics by summing the squared magnitudes of the FFT coefficients in each of the 100 disjoint GSM channels. We let the largest of these statistics determine our prediction for the α active channel positions.

Signal Recovery on Known Channels: Once we have determined the proper channels on which the nonzero Fourier coefficients must be estimated from the nonuniform samples, we set up an overdetermined system of equations $y = A*f$, where f is a complex vector of length K (where $K < M$), y are the M nonuniform samples, and A is an MxK subsampled IFFT matrix whose columns correspond to frequencies only on the active channels and whose rows correspond only to the nonuniform sample times. The vector f represents the unknown Fourier coefficients on the channels to be estimated. (The remaining Fourier coefficients of the signal are implicitly assumed to be zero.) Because the above system is overdetermined, we consider minimizing the l_2

residual $\|y - A*f\|_2$, which can be solved efficiently using conjugate gradients. (The computation merely requires a moderate number of length-N FFTs.)

These are the basic algorithmic components of the following experiments.

EXPERIMENT 1:

As a first experiment, we consider real-world GSM packet signals collected from an antenna at a carrier frequency of approximately 1.9496 GHz. After modulation to baseband, the signal was sampled at a rate of 640kHz and broken into packets of time slot duration 531 microseconds. Each complex baseband packet has much of its energy concentrated in the range [-150kHz,+150kHz]. A total of 623 packets were collected from the antenna.

A single trial of the experiment considers a single time slot and proceeds as follows:

1. Create test signal.
   We select some number α (between 1 and 11) of channels to be active. We randomly select α out of the possible 623 GSM packets from real-world data. These packets are lowpass filtered to the range [-200kHz,+200kHz] and then modulated up to α random carrier frequencies out of the 100 possibilities, with the assurance that no two active channels are fewer than 5 channels apart. The packets are maintained with equal power, a setup possibly representative of the transmission from the basestation. The full, high-rate discrete signal has an effective sampling rate of 46.08MHz, for a total of N = 24480 high-rate samples over the simulation time slot of 531 microseconds.
2. Add noise.
   We select a target SNR (between 0dB and 200dB), based on which we add Gaussian noise across the entire GSM band, so that the SNR *in the active channels* meets this target SNR. For computation of SNR in active channels we use 400kHz of bandwidth in each active channel centered at the carrier frequency.
3. Subsample.
   We select some number M (between 1000 and 5000) of random nonuniform samples from the noisy length-N test signal. The sample times are drawn at random from the N=24480 discrete possibilities of the full-sample signal.
4. Estimate active channels.
   Based on the nonuniform samples, we use the FFT to estimate which channels (out of 100 possibilities) were active. To do this, we create a length-N signal out of the M nonuniform samples by zeropadding at the non-sampled locations. We take the FFT of this signal, and then create 100 test statistics by summing the squared magnitudes of the FFT coefficients in each of the 100 disjoint GSM channels. We let the α largest of these statistics determine our prediction for the α active channel positions.
5. Recover packets.
   At this stage, we use the M available nonuniform samples to invert the measurement process on the estimated active channels. Allowing 400kHz of bandwidth for each packet to be recovered, we have a total of α*400kHz of bandwidth to estimate. Over the time slot of 531 microseconds, this corresponds to approximately α*424 real-

valued unknown parameters, which we must estimate from M real-valued samples. With no additional information on the structure of each GSM packet, this is possible only when M is (moderately) larger than α*424. To recover the coefficients we set up the overdetermined system of equations y = A*f, where f is a complex FFT coefficient vector of length α*212, y are the M nonuniform samples, and A is a subsampled IFFT matrix whose columns correspond to frequencies only on the active channels and whose rows correspond only to the nonuniform sample times. Because the system is overdetermined we minimize the l_2 residual $\|y - A*f\|_2$, which can be solved efficiently using conjugate gradients. (The computation merely requires a moderate number of length-24480 FFTs.)

6. Evaluate performance.

   Based on the recovered FFT coefficients for the active channels, compare the recovered packets to the original packets. Because we lack ground truth on the original GSM bitstreams, our performance metric for this experiment is the decrease in SNR of the recovered packet relative to the noisy packet in step 2 (the "target SNR").

The results of this experiment are presented in Table 1 for various values of α and M. The entries of the table represent the typical decrease in SNR, as described above in step 6. We note that, with a full set of samples (no random sampling) the best possible entry in the table would be 0dB. The decreases we show in the table are fairly consistent over a wide range of target SNRs, and in most cases all α of the active channel positions were correctly estimated in step 4. Our second experiment below better reflects the impact of inaccurate channel estimation.

Table 1: GSM loss (in dB) for Real-world Data

# of non-uniform samples	Number of active channels							
	1	2	3	4	5	7	9	11
1000	-18dB	-50	N/A	N/A	N/A	N/A	N/A	N/A
2500	-10	-12	-13	-17	-25	N/A	N/A	N/A
5000	-7	-7	-8	-8	-9	-10	-13	-22

We can offer a few comments about these results. First, more samples M give better performance, as expected. Second, the results are fairly robust when M is moderately larger than α*424 – in this regime, in fact, with fixed M, the number of α active channels has very little impact on the performance. Third, the entries in the table are somewhat predictable by statistical theory: the decreases in SNR are caused by the presence in y of noise from the inactive channels. The true equation for the measurements y is given by y = A*f + B*g, where A and f are as described in step 5, g is a vector of the FFT coefficients on the inactive channels, and B is a subsampled IFFT matrix whose columns correspond to frequencies only on the inactive channels and whose rows correspond only to the nonuniform sample times. The vector g contains only Gaussian noise, and B*g will approximately represent Gaussian noise, so y can be thought of as noisy observations of A*f. As the length M of y increases,

we have more observations of A*f, each with (approximately) the same noise level. The variance of the estimation should decrease as 1/M, so a 2x increase in M should decrease squared error by 2x, corresponding to an SNR improvement of 3dB. Indeed this is what we see in the lower-left corner of the table.

EXPERIMENT 2:

As a second experiment, we construct synthetic GSM signals, which will allow us to evaluate the actual bit error rate of the reconstructed signals. Our experiment is largely the same as Experiment 1, with the following differences:

- We use a time slot of 472 microseconds instead of 531 microseconds. Over this interval, each message packet has a data rate of 270.8333 kbps, and contains 16 guard bits, leaving 111 bits of information per packet, per active channel, per time slot.
- We construct random bit sequences in Matlab and use a Matlab SimuLink module for GMSK encoding. The full, high-rate discrete signal has an effective sampling rate of 48749940Hz, for a total N=23040 high-rate samples over the simulation time slot of 472 microseconds.
- We limit the reconstruction bandwidth to 270.8333kHz instead of 400kHz. Hence there are approximately 256 real-valued unknown parameters for each packet.
- We evaluate the recovery performance by demodulating each packet from its carrier frequency down to baseband, and using a Matlab SimuLink module for GMSK decoding.

Figure 13, Figure 14, and Figure 15 show the bit error rate (BER) as a function of the nonuniform sampling rate M/N, for various numbers $\alpha = 2,4,8$ of active channels, and with each curve representing a different target in-channel SNR in the noisy generated signals before sampling. The dashed lines represent the breakdown point where $M = \alpha*256$; this is the fundamental sampling limit without any additional models on the Fourier coefficients in an active channel. We see that with only moderate oversampling relative to this fundamental limit, it is possible to recover the GSM signals with very low BER. The recovery is, of course, sensitive to the in-channel SNR before sampling.

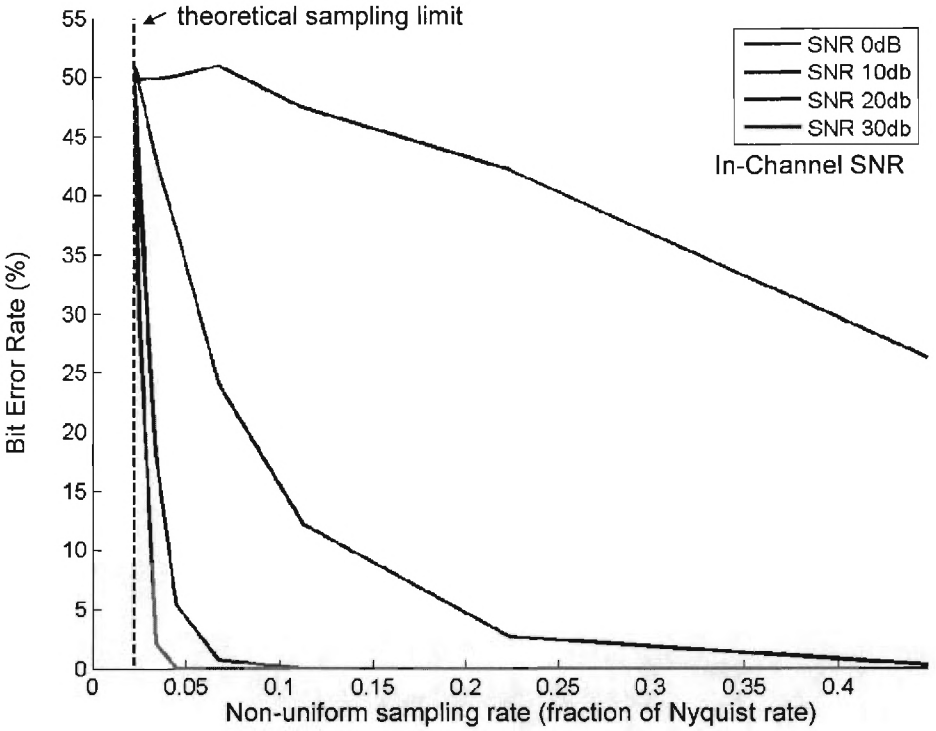


Figure 13: GSM Decoding from N-U Sampling 2 Active Channels

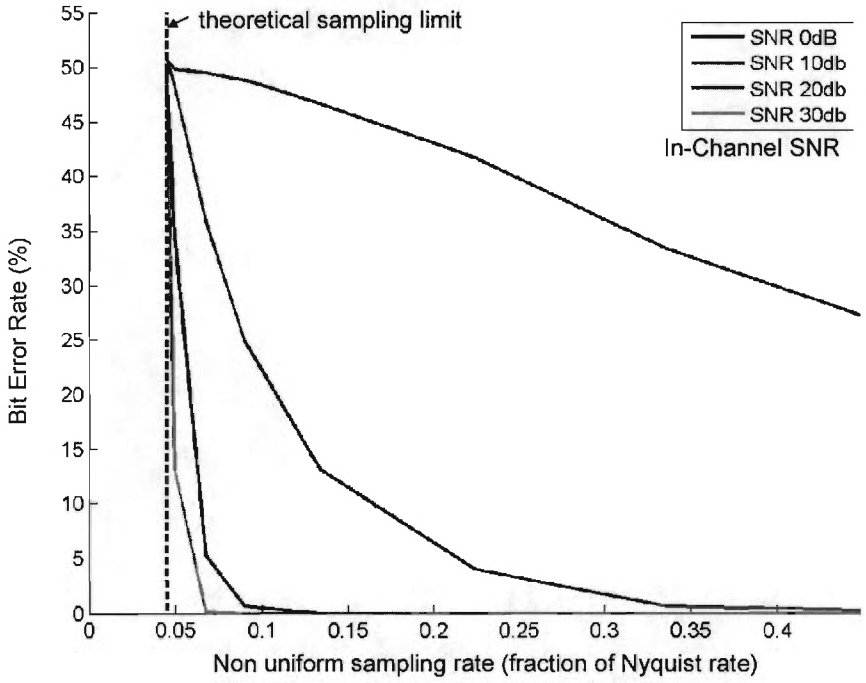


Figure 14: GSM Decoding from N-U Sampling 4 Active Channels

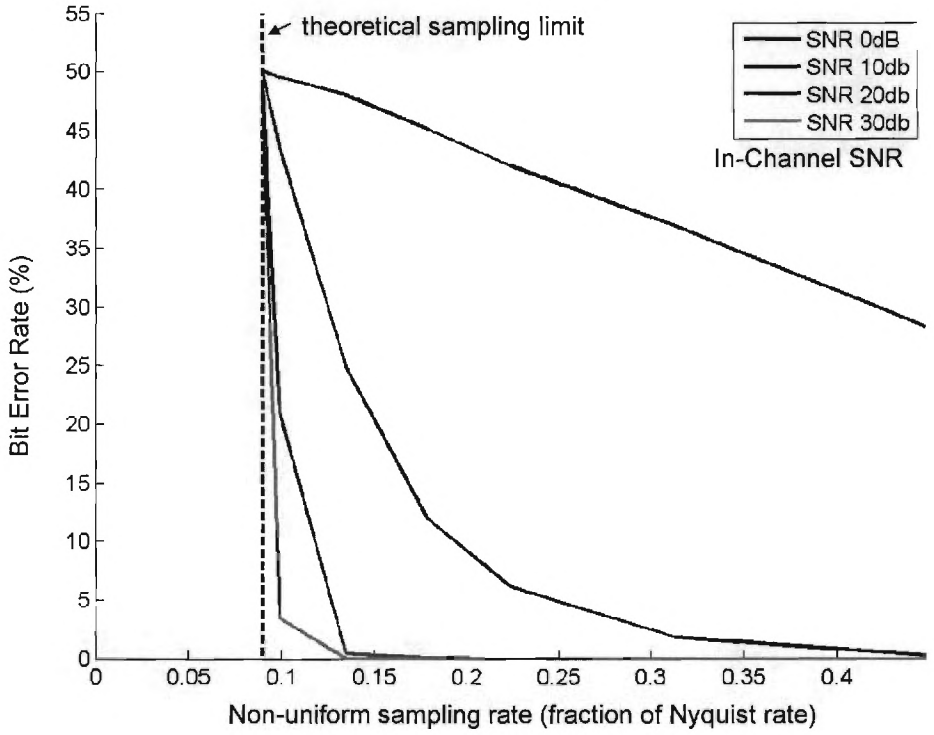


Figure 15: GSM Decoding from N-U Sampling 8 Active Channels

Figure 16 shows the effective number of bits (enob) of the recovered packets, according to the formula

$$\text{enob} = \log_2\left(\frac{2\|x\|_2}{\|\text{error}\|_2}\right)$$

where x is the set of Fourier coefficients of the active packets, and the error is measured in the recovery of these coefficients. (This figure does not involve decoding the recovered pulses at the bit level.) Examining these plots, we see that, down to a sampling regime where M is roughly 2-3x larger than the sampling limit α*256, dividing the sampling rate by 2 will reduce the enob by only ½ bit. As noticed in Experiment 1, this is true across a wide range of in-channel SNRs.

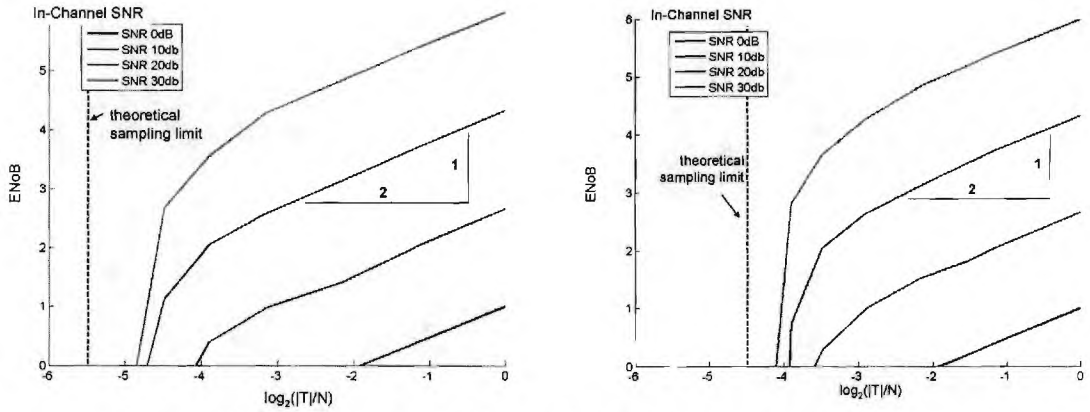


Figure 16: GSM Decoding from N-U Sampling 2 and 4 Active Channels

2.3.2 Analysis of Test Bed Data

We have also conducted simulated GSM experiments with NUS test bed data.

In these experiments, our GSM band occupied 20MHz between 102MHz and 122MHz. We again considered 100 channels within this band, each of bandwidth 200kHz.

We used Matlab and the SimuLink GMSK encoder to construct synthetic GSM signals, as described in Experiment 2 above. The full-sample rate for these synthetic discrete signals was set at 752990800Hz, and the time slot duration was 472 microseconds, giving a total of N=355840 high-rate samples per packet time slot.

These signals were then converted to analog and sampled using the NUS protocol, where the NUS sample positions were taken from a grid of resolution 752990800Hz. From this fine-scale grid, 110 out of every 1024 sample positions were kept (or 38224 out of the 355840 total high-rate samples), giving an effective overall sampling rate of approximately 81MHz.

The discrete sampled data was then imported back into Matlab for reconstruction. The experiments in this section all involve estimation of Fourier coefficients on selected bands within the 752990800Hz spectrum. This estimation from nonuniform samples proceeds exactly as described in the above GSM experiments, and again has complexity proportional to a length-355840 FFT. Subsequent decoding of GSM pulses simply involves demodulation, filtering, and GMSK decoding in Matlab SimuLink.

EXPERIMENT 1:

For the first experiment we act as an oracle that knows which channels will be active, and focus on the quality of the decoded GSM packets as the signal amplitude changes. The

signal amplitude was attenuated by various amounts before sampling, effectively changing the SNR of the sampled data.

Figure 17 shows the decoded BER for several nonuniform sampling rates, where the GSM signal consists of α=3 active channels. The highest rate (approximately 81MHz) corresponds to the entire set of test bed data; lower rates correspond to randomly chosen subsets drawn from this data. We see that, for signal attenuation levels of -40dB or better, the three GSM packets can be decoded with zero bit errors even at sampling rates approaching the fundamental limit (2*α*270kHz). For more aggressive attenuations, the BER improves as the sampling rate improves, consistent with the noise robustness observed in the above GSM experiments.

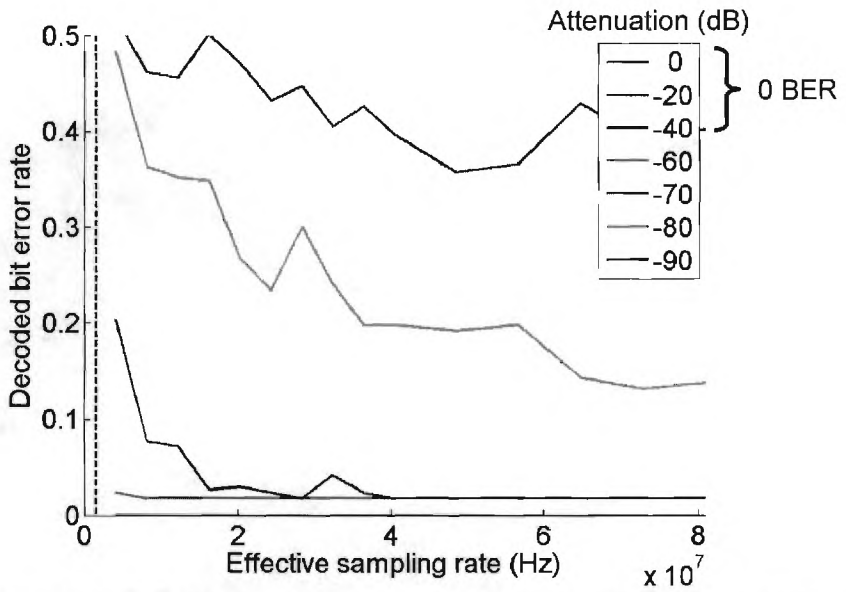


Figure 17: Bit Error Rate for Test Data for Three GSM Channels at Various Amplitudes

To confirm our analogy between signal attenuation and decreased SNR, we conduct a pure-Matlab experiment, where we begin with the original 355840 high-rate samples, quantize to 10 bits of resolution, and add white Gaussian noise intended to mimic the thermal noise of the ADC. Figure 18 shows the results of repeating our experiment on subsampled versions of this data.

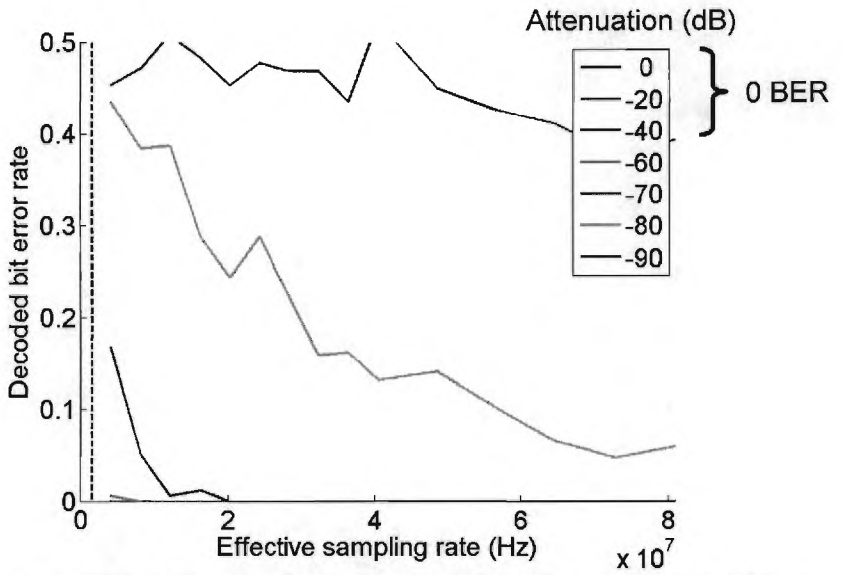


Figure 18: Ideal GSM signals with ADC SNR modeled by additive noise

EXPERIMENT 2:

For our second experiment we consider a more realistic GSM scenario where the received signal contains one or more interferers. In this simulation we model these interferers as strong TV signals (6MHz bandwidth each) with known frequencies that are outside of the GSM band. Figure 19 shows the experimental setup, where the TV signal(s) are added before sampling the signal. The first TV signal is centered at 193MHz; the second is centered at 22MHz.

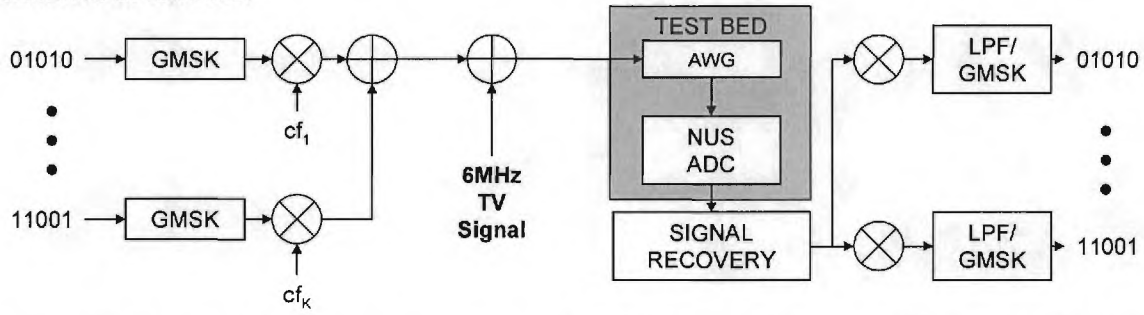


Figure 19: Signal Processing Diagram for Generation and Analysis of Test Bed GSM Signal with TV Interferers

In order to recover the GSM signals, we find it necessary to invert the measurement operator both on the known active GSM channels and on the portion of the spectrum occupied by the TV signal(s). In fact we find it useful to allow a full 10MHz of bandwidth (centered at 193MHz or 22MHz) to reconstruct each TV signal. Our decoding performance, however, is gauged solely on the BER of the GSM packets, not the fidelity of the recovered TV signal.

Figure 20 shows the decoded BER for the GSM packets ($\alpha = 3$ active channels) for various attenuations of the GSM signal before sampling; the TV signal power remains constant while the GSM signals are attenuated. The dashed lines represent the fundamental sampling limit for a bandwidth of α*270kHz + (#TV signals)*10MHz, assuming no additional models for signal structure.

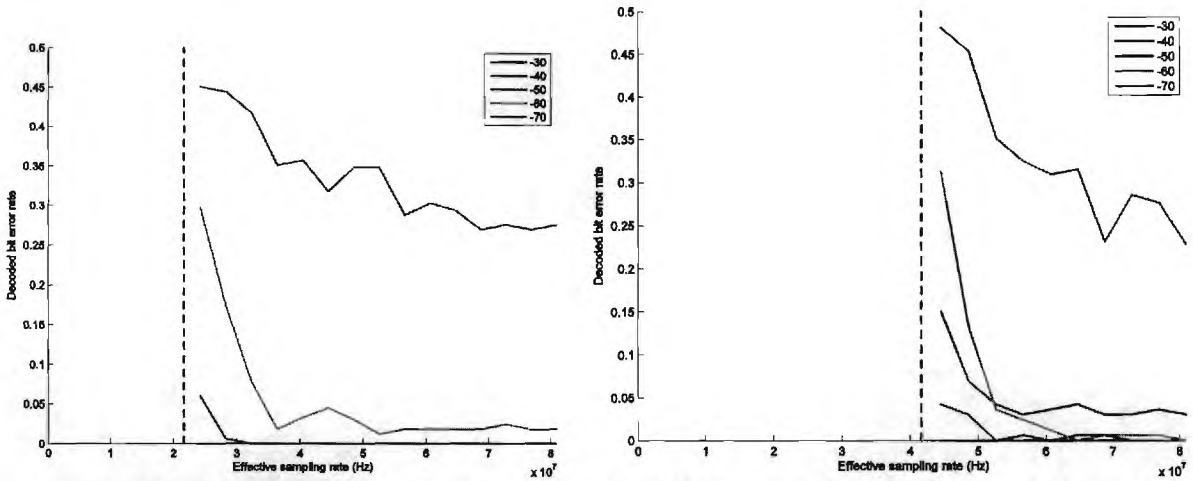


Figure 20: Bit Error Rate from Test Data for 3 GSM Signals plus One and Two TV Signals

Comparing this experiment with the plots in Figure 17, we see that the impact of the TV signals is simply an increase in sampling rate proportional to the bandwidth of the TV signals. In particular, supposing we fix a sampling rate of 2x the theoretical minimum in each plot (dashed line), then for -70dB attenuation we see BER in the range 0.2-0.34 in each plot, and for -60dB attenuation we see BER in the range 0.01-0.02 in each plot.

3 Generalized Compressive Sampling by Random Preintegration

3.1 System Characterization

Under established compressed sensing theory, one can reconstruct signals from measurements made with various random measurement ensembles, in particular the binary random ensemble. With this ensemble, each measurement consists of the inner product of the signal with a random sequence of +1's and -1's. Note that such measurements could be made by sending the signal to one channel for each measurement, randomly flipping the the polarity at the Nyquist rate, and then integrating the result to get each measurement. This is exactly what the RPI architecture does, except instead of perfect integration, a low pass filter is applied, and instead of having a single measurement for each channel, the output is subsampled. This is displayed graphically in Figure 21. The signal x is multiplied by m, a random sequence of +1's and -1's, then convolved with the single-pole filter h, then finally subsampled at times nT. The effective measurement vectors φ_n are plotted at the bottom of the figure. (Note that the plot of h is time-reversed to visually match up with φ_n . What is shown is not the impulse response, but the effective weighting of a sample.)

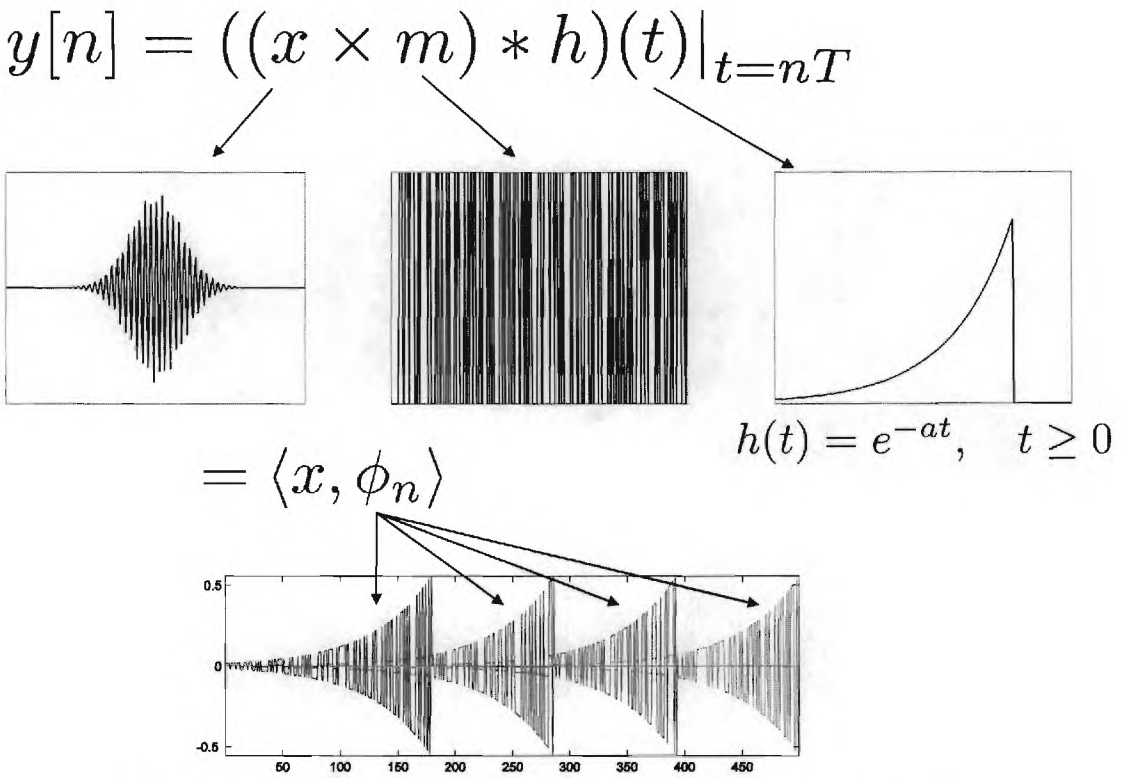


Figure 21. Graphical representation of the model of the measurement process

3.2 Numerical Modeling and Sensitivity Analysis

A single channel of our proposed Random Pre-Integration (RPI) architecture is shown in Figure 21. The input signal is modulated by a pseudo-random polarity (or "chip") sequence $P_m(t)$ and the result is passed through a non-ideal integrator and sampled. There are several non-idealities in this architecture which we will address.

First, added to the signal at the input to each channel is cross talk interference. This is essentially the interference caused by the polarity switch at the other channels. Since the chip sequence is unstructured (it appears random), we expect that the interference will essentially behave like additive white Gaussian noise. The experiments described below verify this.

Next, there is the nonlinearity polynomial, which has a pointwise polynomial response that is dominated by the linear term, and has small quadratic and cubic coefficients.

There is also timing uncertainty when the switch between polarities occurs. At first, the timing errors here seem like they are more serious than in non-uniform sampling architecture, as there will be multiple errors in each measurement. However, we will see

below that this is not too much of a concern; since there is more signal energy in each of the measurements, the overall measurement signal-to-noise ratio only increases by a factor of two. As in the non-uniform sampling case, timing errors do not cause pathological problems.

The integrator is also not perfect. However, if we have a good model for how it behaves, most of its imperfections can be accounted for in the recovery procedure by simply adjusting the measurement matrix used in the COMS algorithm. For the numerical simulations discussed below, we modeled the integrator as a linear system with a single pole at $a = 1.945 \times 10^8$, the resulting impulse response is

$$h(t) = e^{-at}, \quad t \geq 0.$$

The resulting measurement functions, shown in Figure 21, look like decaying chip sequences.

3.2.1 Numerical Model

We simulated the system with bandlimited, periodic inputs on [0,T]. The input can be expanded using the Fourier series

$$x(t) = \sum_{\omega=0,\ldots,N-1} \hat{c}(\omega) e^{j2\pi\omega t/TN},$$

where the maximum possible frequency is $N/2T$. The interval was partitioned into N subintervals by $0 = t_0 < t_1 < \cdots < t_N = 1$, over which the polarity is constant:

$$P_m(t) = p_{m,k} = \pm 1, \text{ on } [t_k, t_{k+1}].$$

The contribution at the output of the integrator over a time period $[t_1, t_2]$ where $t_k \leq t_1 \leq t_2 \leq t_{k+1}$ for some k is

$$H(t_1,t_2) = \int_{t_1}^{t_2} e^{-a(t_2-\tau)} x(\tau)\, d\tau = \sum_{\omega} \left(\frac{1}{a+j\omega} \right) \cdot \hat{c}(\omega) \cdot \left(e^{j\omega t_2} - e^{-a(t_2-t_1)} e^{j\omega t_1} \right)$$

and the output of the integrator at time t is

$$S_m(t) = \sum_{0 \leq k \leq K(t)} p_{m,k} \cdot H(t_k, t_{k+1}) + p_{m,K(t)+1} H(t_{K(t)+1}, t)$$

where $K(t)$ is the largest k such that $t_{k+1} < t$. The above equations define the rows of the measurements matrix, mapping the Fourier coefficients $\hat{c}(\omega)$ of the input to a sample value $S(t_s)$. In general, the partition given by the t_k will be different in each channel, as the sample locations are not necessarily synchronized and each polarity sequence is subject to different timing uncertainties.

3.2.2 Experimental Setup

For the numerical experiments, we used a signal consisting of S=10 cosine components with randomly chosen frequencies. The number of Nyquist samples (which is the same as the length of the chip sequence over the interval [0,T] was N=500. Five samples were taken in each channel, and the number of channels varied between 20 and 200.

The nonlinearity nonideality behaves exactly as in the non-uniform sampling case. For large values of the constants in front of the nonlinear terms, it causes an error which is concentrated on the signal support. Fortunately, we do not expect this to be a limiting factor in a real hardware system.

There are actually three types of timing jitter in our system model: random jitter, pattern jitter (caused by interference from the chip pattern), and aperture distortion (caused by interference from the input signal). We simulated each of these, and they all behaved similarly, basically behaving like random noise with a certain variance (see Figure 22). We discuss only the case of random jitter in detail below.

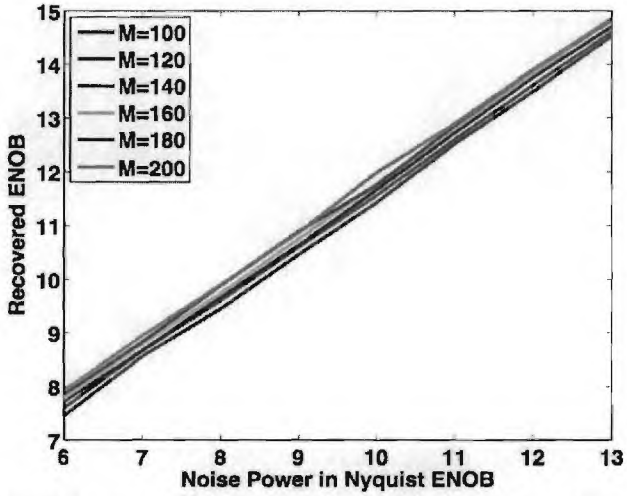


Figure 22: Thermal Noise Results. Recovery error scales linearly along with the measurement error.

3.2.3 Cross Talk

The cross talk was modeled as additive noise to the input at each channel. The magnitude of this interference depends on the locality of the measurement circuits for the two channels, here we assume that the power scales like d^{-2}, where d is the physical distance between the two channels. (This behavior does not matter much for what we are trying to establish here, i.e. that the cross talk noise does not cause pathological problems.) Specifically, the input to each of the M channels was added with a different noise realization

$$X_k(t) = \sum_{m=1}^{M} \frac{C_{XT}}{|m-k|} P_m(t - \tau_{m,k})$$

where P_m is the polarity of channel m, C_{XT} is a constant which controls the magnitude of the noise, and $\tau_{k,m}$ is a delay between channels k and m modeled here as

$$\tau_{m,k} = \frac{10^{-9}}{|m-k|},$$

but again this choice does not seem to be too critical for our end results. Since the polarity sequences are essentially random, the noise in one channel will have variance

$$\sigma^2 \approx \frac{\pi^2}{6} C_{XT}^2.$$

It is important to realize that the cross talk in each channel is essentially independent, and so the cross talk noise power in the set of all measurements will scale with the number of channels.

Figure 23 below shows that the recovery error scales linearly along with the measurement error, illustrating that the cross talk does not cause any pathological problems for the COMS recovery procedure.

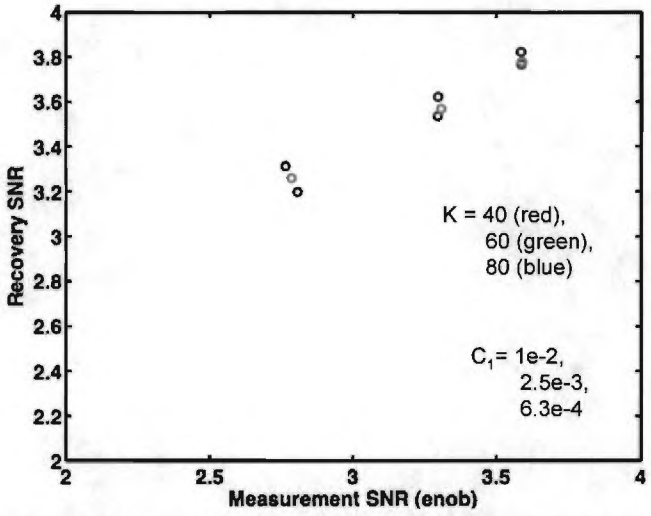


Figure 23: Cross-talk Results. Recovery error scales linearly along with the measurement error.

3.2.4 Timing Uncertainty

There will of course be some (hopefully small) uncertainty in the timing of the polarity change. We expect that this uncertainty will be on roughly the same order as the sample location jitter in the non-uniform sampling architecture. A point of concern for the RPI architecture is that each single measurement will now suffer the effects of multiple timing

errors. However, more signal power is also captured in each measurement, which we will see mitigates this effect.

Consider first what we might call an ideal RPI measurement system. A single measurement can be expressed as

$$y_k = \sum_{i=1}^{P} c_i (S(t_i) - S(t_{i-1}))$$

where $S(t_i)$ is the integral of the input signal up to time t_i, $c_i = \pm 1$ (the "chip") is the value of the polarity over the interval $[t_{i-1}, t_i]$, and P is the total number of chips used for a single measurement. If the input signal is a sinusoid, then $S(t)$ will also be a sinusoid, and we can write the error in the measurement as

$$e_k = \sum_{i=1}^{P} c_i (X(t_i) - X(t_{i+1}))$$

where $X(t_k) = S(t_k) - S(t_k + \delta_k)$ is the time jittered value of a sinusoid (as we studied in Section 2.1). The expected noise power will be

$$\mathrm{E}|e_k|^2 = \sum_{i=1}^{P} (X(t_i) - X(t_{i-1}))^2 \approx 2PA^2\omega^2\sigma^2$$

where A and ω are the amplitude and frequency of the (complex) sinusoidal input, and σ^2 is the variance of the Gaussian timing jitter. The expected measurement power will be

$$\mathrm{E}|y_k|^2 = \sum_{i=1}^{P} (S(t_i) - S(t_{i-1}))^2 \approx A^2 P$$

if the nominal chip width is far enough below the Nyquist spacing. The expected SNR for the measurements is thus just a factor of two smaller than in the non-uniformly sampled case

$$\mathrm{SNR} = \frac{1}{2\omega^2\sigma^2}.$$

As we can see in Figure 24 below, timing uncertainty also does not cause unexpected problems. The output SNR tracks the measurement SNR in a straight line.

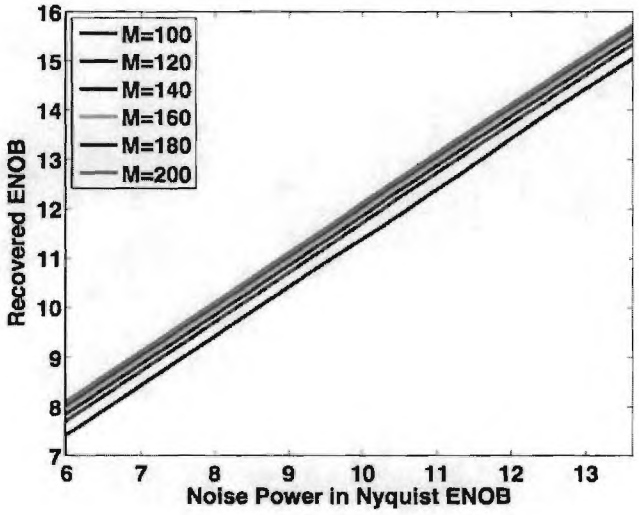


Figure 24: Timing Jitter Results. Recovery error scales linearly along with the measurement error.

3.3 Algorithms for Sparse Signal Recovery

In this section we describe recent theoretical and algorithmic work geared toward improving sparse signal recovery. We first describe the construction of a multiscale Gabor dictionary intended to provide sparse representation of smooth modulated pulses. We also describe two variations on the l_1 recovery technique that improve the recovery of sparse signals from limited measurements. The work in this section is abstract, and applies generically to compressive measurements of any modality (not necessarily taken from the RPI system). However we subsequently validate these ideas by testing on RPI data in Section 3.4.

3.3.1 Multiscale Gabor Dictionary

In order to reconstruct we need for the signal to be sparse in some dictionary Ψ. That is, we model the signal as a linear combination of relatively few vectors from some collection of vectors $\{\psi_i\}$. Ideally the vectors ψ_i resemble typical signals, and yet are generated in a systematic way so that they form a frame and can be computed quickly. Note that the question of which dictionary to use (as well as which reconstruction algorithm to use) is somewhat independent from the measurement process, so some of our experiments have been with synthetic data using ideal measurement matrices.

The one dictionary we have tried with which we have had the most success is a Gabor (or windowed Fourier) frame dictionary. This consists of Fourier vectors (sine waves) of all frequencies modulated by smooth windows to form pulses. The windows are all of the same shape, but they are of different scales and shifts. For example, the windows for the largest scale are plotted in Figure 25. Typical elements of the dictionary are plotted in Figure 26.

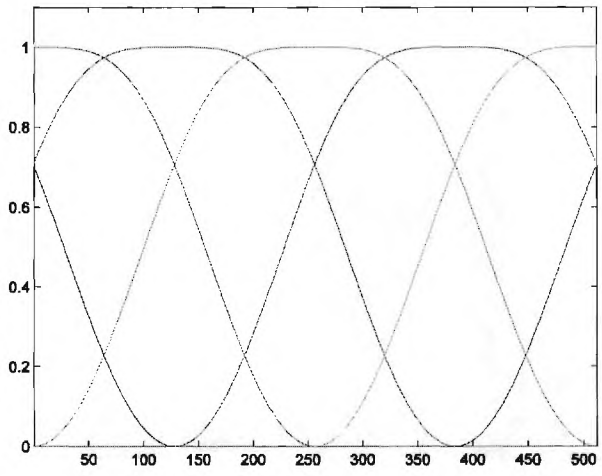


Figure 25. Largest scale iterated sine windows used for Gabor dictionary

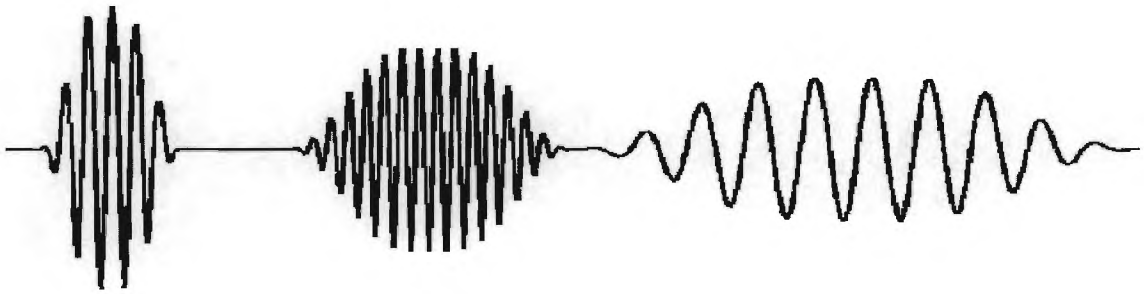


Figure 26. Example elements of Multiscale Gabor Dictionary

If the entire collection of windows form a partition of unity (meaning that the sum of their squares equals 1 everywhere), then this is a tight frame. There are two types of windows we tried using, so-called iterated sine windows, which do form a partition of unity, and Gaussian windows, which do not. It is useful but not essential that the frame be tight, and in some cases the Gaussian windowing works better.

Note that we do not do anything sophisticated to handle the boundary; for the windows which do not fit within the boundary, we simply treat the signal as equaling zero outside the boundary. However, in the RPI setup, it is reasonable to assume that since the signal is contained entirely in the interior of the domain. This is because we have a stream of data, and pulses will be relatively rare events, so if the signal is ever not in the center, we can recenter by taking a shifted block of adjacent RPI output data.

Throughout the reconstruction algorithm, we need to repeatedly apply the dictionary and its transpose. Unlike the measurement matrix Φ, the dictionary Ψ can be extremely large, so it is important that we have a fast algorithm for applying it without having to compute it

explicitly. Indeed, there is a fairly straightforward algorithm using FFT's. A rough outline is as follows:

- To compute the frame coefficients of a signal (apply analysis matrix Ψ^T)
  - Initialize the output vector as empty
  - At each scale,
    - At each shift,
      - Multiply input vector by windowing function
      - Pad with zeros to oversample in frequency
      - Apply FFT, and appropriate transformation to make it real-valued
      - Concatenate the results to the output vector
- To compute a signal based on its frame coefficients (apply synthesis matrix Ψ)
  - Initialize the output vector as a zero vector of the correct size
  - At each scale,
    - At each shift,
      - Take frame coefficients and delete them from the input vector
      - Apply appropriate transformation and IFFT
      - Delete excess data (transpose of padding with zeros)
      - Multiply by the windowing function
      - Add the result to the output vector

Note that we can effectively oversample in scale, space, and in frequency, making the dictionary richer. When deciding how much to oversample, there are a few trade-offs involved. If the dictionary is too small (for example, if we only included windows of much larger scales than the support of the signal) then it will not be effective at capturing the signal, and fidelity of reconstruction will suffer, or fail completely. Obviously, making the dictionary extraordinarily large will make the computational complexity of the reconstruction intractable. But even if computation is not an issue, adding too many elements to the dictionary will impair the fidelity of reconstruction by adding extraneous elements to the dictionary.

In our experiments, when working on a vector of length 2^9, we use windowing scales 2^6 through 2^9, then oversampling by a factor of 2 in space, and 2 in frequency, resulting in a total of 22144 frame vectors.

3.3.2 Variations of L1 Recovery

In this section we briefly discuss two variations on the canonical l_1 recovery program designed to improve the recovery of sparse signals. After discussing the theory and motivation behind these approaches, we present preliminary experiments supporting the potential benefits in signal recovery, particularly when these two modifications are implemented simultaneously.

3.3.2.1 Synthesis Vs. Analysis

There is a subtle variation in the canonical l_1 recovery program for overcomplete dictionaries. If we believe the dictionary Ψ is a good representation for the signals we are trying to acquire, there are two ways to formulate the problem. The first is standard l_1 synthesis:

$$\min_{\alpha} \|\alpha\|_1 \quad \text{such that} \quad \Phi\Psi\alpha = y.$$

This program looks for the sparsest combination of the columns of Ψ that matches the observed measurements y. (To make things simple, we are using equality constraints here, but the main points of this discussion will not change if we relax $\Phi\Psi\alpha = y$ to $\|\Phi\Psi\alpha - y\|_2 \le \varepsilon$ or $\|\Psi^T\Phi^T(\Phi\Psi\alpha - y)\|_\infty \le \varepsilon$.) The other option for recovery is to solve the l_1 analysis problem

$$\min_{x} \|\Psi^T x\|_1 \quad \text{such that} \quad \Phi x = y,$$

which looks for the signal $x(t)$ whose forward transform coefficients $\Psi^T x$ are as sparse as possible. If the dictionary Ψ is orthogonal, these programs are equivalent. For overcomplete Ψ (when Ψ has more rows than columns) though, they are different. To see this, we rewrite the analysis problem as

$$\min_{\alpha} \|\alpha\|_1 \quad \text{such that} \quad \Phi\Psi\alpha = y, \quad \Psi^T\Psi\alpha = \alpha.$$

Thus the l_1 analysis problem is more tightly constrained than the l_1 synthesis problem.

In the experiments below (see Section 3.3.3), where Ψ is a dictionary of modulated pulses at different shifts (and thus its columns are highly correlated), it seems that the additional constraints that l_1 analysis imposes are a significant aid in the recovery process. A theoretical understanding of this phenomena is a subject of current research.

3.3.2.2 Reweighting L1

We discuss a slight reformulation of $l_1$minimization that improves the recovery of sparse signals. Historically, the motivation for choosing the l_1 norm in the optimization

(1) $$\min \|\alpha\|_1 \ subject\ to\ y = \Phi\Psi\alpha$$

comes from the fact that it provides a convex relaxation of a more desirable but intractable optimization problem

(2) $$\min \|\alpha\|_0 \ subject\ to\ y = \Phi\Psi\alpha$$

where α is a coefficient vector, Ψ is a sparse basis or dictionary transforming the sparse coefficients to a signal $x = \Psi\alpha$, and Φ is the measurement matrix taking the signal to the measurements $y = \Phi x$. For the present discussion let us assume α is Nx1, $\Psi$is NxN, and Φ is MxN with M<N.

Now, let W denote an NxN matrix with nonzero entries $w_1, w_2, \ldots, w_N$ along the diagonal and zeros elsewhere. We note that in (2), the norm $\|\alpha\|_0$ can be replaced by the reweighted norm $\|W\alpha\|_0$ without changing the solution. However, minimizing the corresponding relaxation

$$\textbf{(3)} \qquad \min \|W\alpha\|_1 \ \textit{subject to}\ y = \Phi\Psi\alpha$$

will not return the same solution as the unweighted norm $\|\alpha\|_1$ in (1).

Our goal is to obtain a set of weights W that actually improve upon the l_1 norm, making it behave more like the l_0 norm. The key difference between the l_1 and l_0 norms is the dependence on magnitude --- larger coefficients are penalized more heavily than smaller coefficients, unlike the democratic penalization of the l_0 norm. Ideally the weights W could correct for this imbalance, by setting w_n inversely proportional to α_n. Of course, the true coefficients α are not known in advance, but an iterative procedure can be used that alternates between estimating $\hat{\alpha}$ and redefining the weights. The algorithm is as follows

1. Set the iteration i = 1. Set $w_n^{(1)} = 1$ for n = 1,2,…,N.
2. Let $\hat{\alpha}^{(i)}$ be the solution to (3) with weights $w_n = w_n^{(i)}$.
3. Set $w_n^{(i+1)} = \dfrac{1}{\left|\hat{\alpha}_n^{(i)}\right| + \varepsilon}$.
4. Increment i and go to step 2.

The parameter ε in step 3 should be set slightly smaller than the expected entries of α. As demonstrated below, the recovery process is somewhat robust to the choice of ε.

As an alternative interpretation of this reweighted l_1 recovery algorithm, it is interesting to note that every iteration of the above algorithm is guaranteed to decrease [12] the following function of $\hat{\alpha}$: $\sum_{n=1}^{N} \log\left(\left|\hat{\alpha}_n\right| + \varepsilon\right)$. This objective function is concave (not convex) but more closely resembles the l_0 norm. Unfortunately the algorithm is guaranteed only to converge to a local (not necessarily global) minimum.

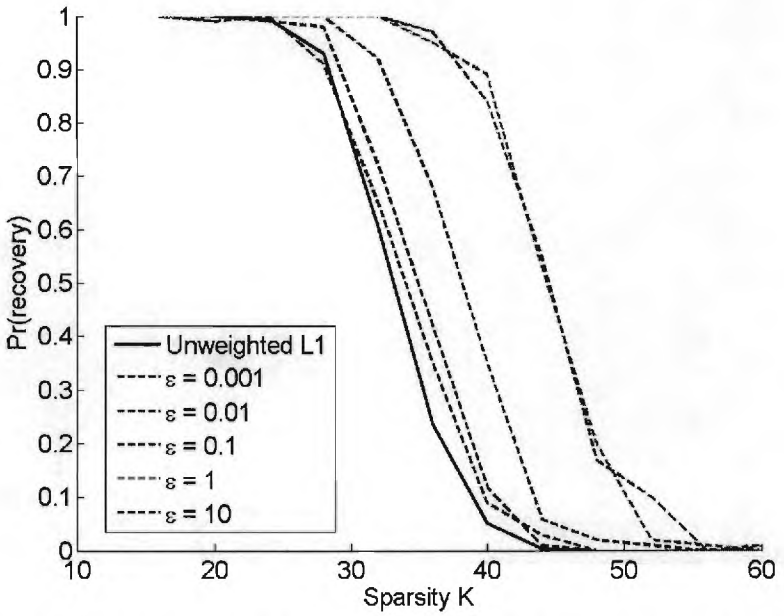


Figure 27: Sparse signal recovery from M = 100 random measurements of a length N = 256 signal. The probability of successful recovery depends on the sparsity level K; the dashed curves represent a reweighted l_1 algorithm that outperforms the traditional unweighted l_1 approach (solid curve).

Figure 27 shows a simple experiment with N = 256. For several values of K, we construct K-sparse signals randomly in the time domain (Ψ is the identity matrix), assigning zero-mean, unit-variance Gaussian coefficient values to K nonzero elements in random positions. For each signal we set M = 100 and construct a random M \times N measurement matrix Φ with iid Gaussian entries. The solid black curve denotes the probability (over 100 trials with random x and Φ) of correctly recovering the K-sparse signal from the measurements $y = \Phi x$ using the unweighted l_1 algorithm. We see that, using M = 100 measurements and the unweighted l_1 algorithm, we can correctly recover K-sparse signals with very high probability when $K \approx 25 = M/4$.

Figure 27 also shows the results of *reweighting* l_1; the dashed curves represent several different values for the parameter ε. We see a marked improvement over the unweighted l_1 algorithm --- with the proper choice of ε, the ability to recover sparse signals has increased from $K \approx 25 = M/4$ to $K \approx 33 = M/3$.

Preliminary work has also shown promising results from reweighted versions of other CS recovery algorithms. For example, in the case of noisy measurements we have experimented with a reweighted version of the quadratically constrained l_1 program

$$\min \|W\alpha\|_1 \ \ subject\ to \ \|y - \Phi\Psi\alpha\|_2 < \gamma .$$

For moderate noise levels, reweighting typically yields signal estimates with 30-40% lower mean-square error (MSE) than the unweighted algorithm.

3.3.3 Combined experiments

We have also tested our variations of l_1 recovery on recovering a synthetic modulated Gaussian pulse from compressive measurements. The pulse x has length 128 and is shown in Figure 28. From this signal we construct a 30x128 measurement matrix Φ having iid Gaussian entries and obtain the 30 measurements $y = \Phi x$.

From these measurements we use several variations of the l_1 algorithm to recover the pulse; in each case we use a sparse Gabor dictionary Ψ. Figure 29 shows plots of the reconstruction *errors* of the recovered signal relative to the true signal x, the numbers quoted indicate the l_2 reconstruction error as a percentage of the l_2 norm of the original signal x.

We see in this experiment that analysis-based l_1 outperforms the traditional synthesis-based l_1 recovery, and that reweighting each of these methods further improves the performance. The best performance, by a factor of 10 in both l_2 error and absolute error, is achieved by combining these benefits: using a *reweighted analysis-based $l_1$recovery method.*

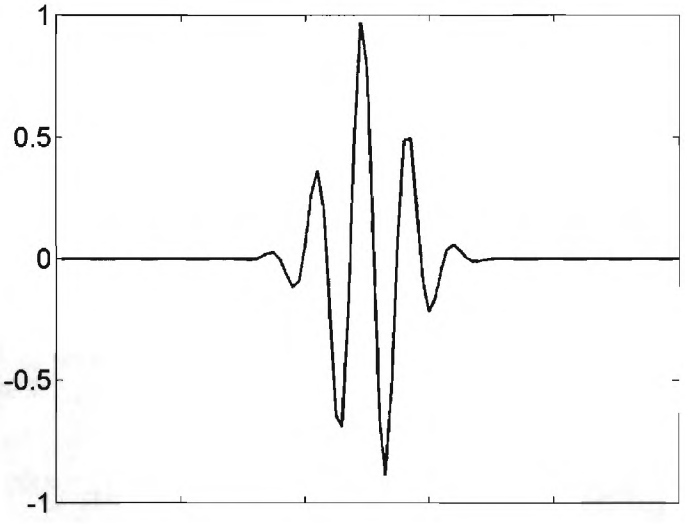


Figure 28: Synthetic modulated Gaussian pulse for testing variations of l_1 recovery.

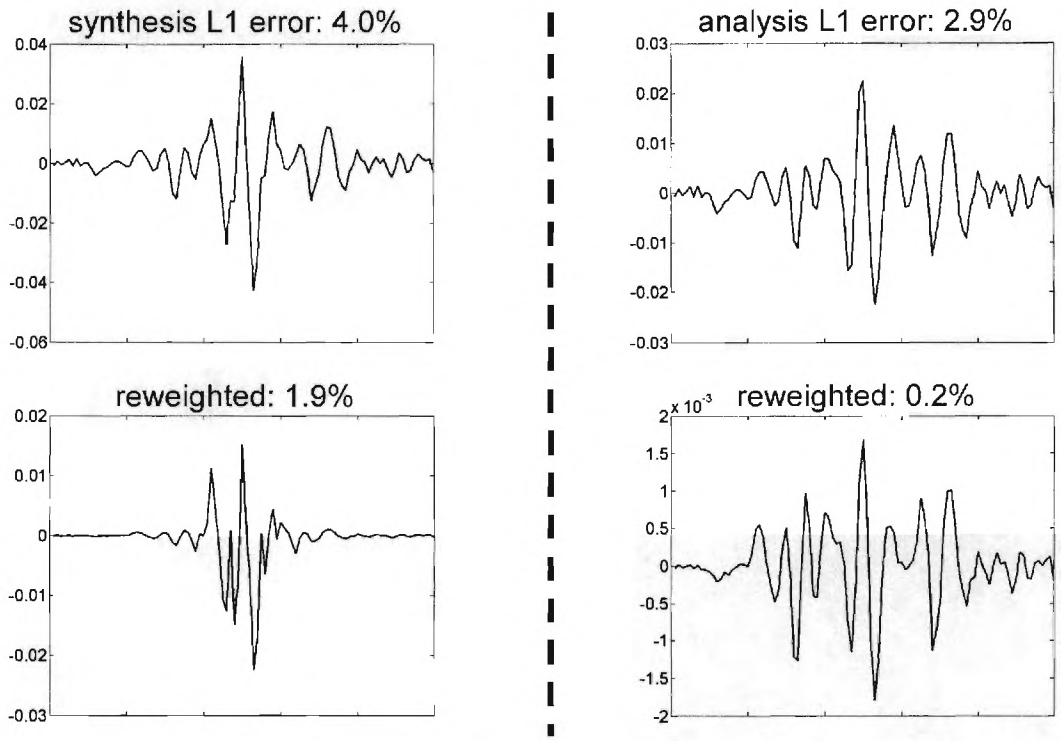


Figure 29: Reconstruction error plots for synthetic Gaussian pulse using variations on l_1 recovery.

3.4 Analysis of Circuit Simulation Results

3.4.1 Discrete System Characterization

Of course, the reconstruction algorithms require knowing the measurement matrix Φ. Note that the resolution of the domain of Φ can be any sampling rate, but for simplicity in our experiments, we took it to be the same as the Nyquist rate (which is also chip rate of the random polarity changes). So to analyze the results of the SPICE simulations, we construct Φ exactly as outlined above.

1. Load the signs of the chip sequences for each channel
2. Load the impulse response
3. For each sample of each channel, we take the time-reversed impulse response, shifted to the sample location and multiply that by the chip sequence for that channel to get the measurement vector φ_n.

To get the measurement data, we just take the output from the integrator simulations and subsample at the locations corresponding to the above measurement vectors. Ideally these should equal the inner products $\langle x, \varphi_n \rangle$, where x is the input signal subsampled at the Nyquist rate.

The measurement process is not precisely the same as a true binary random ensemble, but in our experiments, it is close enough to still give good results. Exactly how many channels are necessary and how high the sample rates should be has not been entirely resolved. Obviously, more channels and higher sampling rates will result in higher fidelity.

Note that with the entire set of simulation output, we can experiment by choosing the sampling rate arbitrarily. Also, we can choose to limit our measurements to less than the full set of channels. If we do either of these, then we need to build the measurement matrix (in steps 1-3 above) appropriately for use in the reconstruction algorithm.

We have run experiments with both 32 channels at a low sampling rate and 8 channels at a high sampling rate, (such that the total number of measurements in each case was the same). For one specific experiment, using a single Gaussian pulse input and synthesis based reconstruction on noisy SPICE data, the 8-channel measurements resulted in an 11.7% error, whereas the 32-channel measurements resulted in a 5.8% error. This supports the expected trend – as the number of channels increases, the measurement process more faithfully mimics a true binary random ensemble, which should improve performance.

3.4.2 Two-Pulse Reconstruction Experiments

Using the procedures outlined in previous sections, we have run experiments on data from RPI spice simulations. For thoroughness, every combination of 4 different variables was run: whether to use synthesis or analysis, whether to use data from 8 channels or 32 channels, which window shape to use, and whether to use reweighting or not. The relative error of all cases is shown in the following table:

		Iterated Sine Windowing	Gaussian Windowing	Iterated Sine Windowing, Weighted	Gaussian Windowing, Weighted
Synthesis-Based	8 Channel	8.37%	8.58%	7.91%	8.13%
	32 Channel	37.26%	6.28%	36.79%	4.85%
Analysis-Based	8 Channel	6.70%	6.73%	6.52%	6.48%
	32 Channel	17.33%	20.55%	4.26%	4.77%

In some cases, the reconstruction algorithm failed to completely resulting in an abnormally large error. The best possible reconstruction, with a relative mean square error of 4.26%, is shown in Figure 30 along with the original signal. This was based on 4 measurements each from 32 channels, using a reweighted analysis based reconstruction, and iterated sine windows.

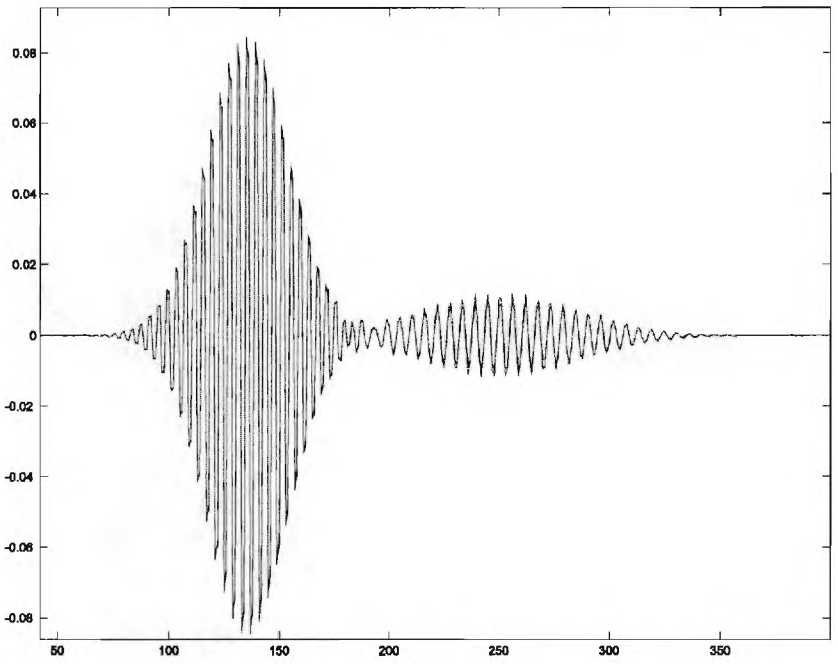


Figure 30. Best reconstruction from RPI SPICE simulation data (green) vs. Original signal (blue). Based on data from 32-channels (4 measurements each). Algorithm used reweighted analysis, iterated sine windowing for the frame.

4 Future Plans

As mentioned in the introduction, our plans for the future include the design and hardware implementation of a high-speed non-uniform sampling architecture and of a random pre-integration architecture operating in the GHz range. A critical component of these projects is of course the development of efficient hardware implementation of data processing architectures for COMS. All along this program, sparse recovery computations (COMS) were performed in an off-line fashion: the output of a nonuniform sampling ADC was captured and delivered in batches to l_1-MAGIC, a MATLAB-based software package—developed by the Caltech team—which performed the computations. One of our future goals is to demonstrate the feasibility of real-time operation at high sample rates. In this section, we assemble the evidence that this goal can be achieved within the lifetime of a possible

follow-up project through a combination of improvements in algorithms, software, and hardware.

Before we begin, we would like to emphasize that the definition of "real time" depends on the application. In static medical imaging, for instance, current performance levels might be sufficient. For the Analog to Information program, however, the sparse recovery algorithm takes on the role of signal acquisition, so to be most useful COMS should require an amount of time which is not too long compared to the duration of the signal itself. It goes without saying that the faster we can solve the sparse recovery problem, the wider the variety of applications we can pursue.

Now we believe that our future projects hinges on the strongest possible expertise in the area of large scale optimization, and in the area of digital and analog signal processing, and this is the reason why our team has started a collaboration with the group led by Professor Stephen Boyd from Stanford University and with Professor Paul Hasler from the Georgia Institute of Technology. Professor Boyd and Professor Hasler are world leaders in the aforementioned areas. In concrete terms, our collaboration started on January 16th 2007, when Caltech organized a meeting between the Caltech-NG team and Paul Hasler and Michael Grant, a senior member of the Boyd group. This meeting generated a lot of enthusiasm and the confidence that together, we could achieve something significant.

Our interest in collaborating is genuine; the paragraphs below were written jointly with Michael Grant and Paul Hasler.

4.1 Speeding up COMS

How do we improve the performance of COMS? More specifically, how do we accelerate the solution of sparse recovery problem

$$\min \|x\|_1 \quad subject\ to\ Ax = b \tag{4}$$

or of the Dantzig selector (DS) model (the noise aware version of the above program)

$$\min \|x\|_1 \quad subject\ to \left\|A^*(Ax - y)\right\|_\infty \le \varepsilon \ ? \tag{5}$$

Here and below, A is the sensing or the sampling matrix relating the features of the signal we wish to reconstruct to the measured data. If an interior-point algorithm is applied, the computational effort is dominated by the execution of matrix-vector products of the form Av and A^*w. All other calculations, including the determination of step lengths and evaluation of stopping criteria, constitute a tiny fraction of the effort. Thus the acceleration of such an algorithm must necessarily focus on two goals:

- reducing the number of matrix-vector products that are required; and
- computing those products as fast as possible.

The first of these goals requires fundamental improvements in the underlying algorithm; the second is more a function of lower-level software, firmware, and/or hardware performance. We address them in order.

4.2 Algorithmic improvements

Compressive Sampling is attracting an enormous amount of attention and each week, a flurry of new articles are released on this subject. For examples, researchers in optimization are proposing improvements over the standard solvers that one finds in l_1-MAGIC. Recent developments for models related to (4) and (5) suggest that we should be able to reduce the several thousand matrix-vector products l_1-MAGIC is using by an order of magnitude or more. Some of the ideas to achieve this are rather straightforward. Modern interior point methods produce a sequence of guesses $x^{(n)}$ converging to the solution. Each update $x^{(n)}$ is obtained from the previous guess $x^{(n-1)}$ by solving a large system of linear equations (this is the famous Newton step). Therefore, the speed of the optimization algorithm is tied to the ability to solve very large systems of linear equations; for the problems we are interested in, the sizes of these linear systems prohibit direct methods, and we need to employ iterative methods such as the method of Conjugate Gradients. It is well known that the accuracy and speed of such iterative solvers depend upon the conditioning of the linear system we wish to invert. The problem is that near the solution, such systems are typically ill-conditioned. One way out is to use preconditioning.

In this direction, Stanford University researchers S.-J. Kim, K. Koh, *et. al.* have developed a new method for solving large-scale l_1-regularized least squares (L1RLS) problems [5]. Like l_1-MAGIC, their method is based upon a standard interior-point method, and uses a conjugate gradient (CG) method to compute search directions. But Kim *et. al.* have developed two key contributions to improve performance: a fast and effective preconditioner to reduce the number of CG iterations required, and a more effective method of controlling the algorithm itself.

With a straightforward MATLAB implementation of their algorithm, they demonstrate considerable improvement in performance over other methods applied to L1RLS problems—not just l_1-MAGIC but also very specialized methods such as HOMOTOPY [2]. For example, a 256K-point medical imaging example was solved in only 130 CG iterations, as opposed to 6000 CG iterations for l_1-MAGIC.

Our problem formulations may be a little more complex than L1RLS, but the models do share a significant amount of structure; and the higher-level algorithmic improvements developed by Kim *et. al.* can be translated rather easily to our problems. Thus we can reasonably expect that by developing a similarly effective preconditioner, we can obtain relative performance gains comparable to those achieved for L1RLS. Indeed, the Stanford team is eager to assist in this effort.

All of this is extremely encouraging, because for the non-uniform sampling architecture, our projections indicate that the algorithms would run 20 to 30 times faster than the current implementation. More importantly, the signal reconstruction would take only a few hundred FFTs, which is an important step toward real time computations.

4.3 High-speed digital implementations of COMS

In the case where the signal in question is sampled in the time domain and compressible in the Fourier domain, the rows of A are selected orthogonal rows of a Fourier transform matrix. (Even in the RPI architecture, most computations reduce to taking FFTs.) In such a case, FFT methods can be applied to accelerate the computation of matrix-vector products. Highly optimized software libraries are available to perform FFTs on general-purpose computer systems and DSP chips. IP cores are readily available to perform the same operations on FPGAs and ASICs.

To illustrate what kind of performance improvements are possible, we have assembled FFT benchmark data for various software/hardware scenarios in Table 2 below. We have chosen two data lengths, $2^{16} = 256^2 = 65536$ and $2^{20} = 1024^2 = 1048576$ points, and have presented the performance of a single FFT computation in in terms of both speed (in Gflops) and time (in milliseconds). The first scenario employs double-precision calculations in MATLAB, the method that l_1-MAGIC currently employs. For all others, single-precision floating point performance is quoted.

Table 2: Benchmarks for various single-precision floating-point FFT implementations

hardware/software		speed Gflops	time ms	source
Intel Core Duo 2.16GHz	65536	0.19	13.9	authors
MATLAB 7.3 (double precision)	1048576	0.22	235	
Intel Core Duo 2.16GHz	65536	0.77	3.4	authors
MATLAB 7.3	1048576	0.37	139	
Intel Xeon 3.0GHz	65536	11.0	0.24	[3]
Intel IIP library FFT	1048576	5.0	10.4	
350MHz TI C67x DSP	65536	1.0	2.6	[9]
	1048576	1.0	52.5	
Texas Memory Systems	65536	3.2	0.82	[10]
TM-44 Custom DSP	1048576	3.3	15.7	
200MHz Virtex-4 FPGA	65536	4.0	0.65	[1]
4DSP FFT core	1048576	5.0	10.5	
NVIDIA 7900 GPU	65536	6.1	0.43	[4]
GPUFFTW software	1048576	6.1	8.6	
3.2GHz Cell Processor	65536	41.8	0.063	[11]
	1048576	35.9	1.5	

The table reveals several interesting points. First of all, the overhead imposed by MATLAB seems considerable, and goes well beyond differences in processor speed. Some of this difference can be easily explained. For example, the Intel IIP library can utilize multiple cores simultaneously, while MATLAB 7.3 cannot. Furthermore, MATLAB is performing

additional steps, such as bit-reversal reordering, that would not be necessary in a custom code.

The GPU results are more compelling than they may appear to be; the processor used for those benchmarks is several months old and has been superceded by a significantly faster model. In practice, a GPU are treated as a coprocessor: the primary CPU is retained for higher-level algorithmic tasks, and the GPU is assigned that portion of the computational burden for which its vectorized architecture is best suited (e.g. FFTs). Care must be taken in such an arrangement to control memory bandwidth consumption [6].

The Cell processor produces the best results by a 4x-6x margin. Not surprisingly, there has been considerable interest in the Cell computing architecture for scientific computing. The paper cited above cites performance estimates for other computational tasks besides the FFT [11]. The Stanford Folding@Home project, which has for some time used GPUs to accelerate their protein folding simulations [8], is now developing a version of their software to run on the Cell-based Playstation 3 [7].

Neither the DSP- nor FPGA-based solutions produce better performance, though in lower-power applications they may be preferable. If an application demanded it, the sparse recovery algorithm could be implemented on a custom ASIC. Such a solution could achieve FFT performance that exceeds the best software/hardware combinations above, while simultaneously lowering power consumption.

4.4 Analog solutions

An interesting question is whether analog signal processing could relieve some of the pressure on COMS. In addition to increasing computational speed, another motivation for analog processing is the *efficiency* in terms of the number of computations performed per unit of power consumed. This could potentially enable a wide range of portable and power constrained applications. Interestingly, some areas of analog signal processing, including vector-matrix multiplication, analog filterbanks, Gaussian mixture model classifiers, and hidden Markov model classifiers have experimentally demonstrated at least a power efficiency improvement of 1000 over custom digital solutions [13].

COMS requires a physical implementation that solves constrained convex optimization problems. Analog solutions to such problems usually begin by creating an energy surface that is minimized by the physics of the analog computation; that is, the solution converges to the lowest energy state. One early formulation involves building Hopfield networks to solve linearly constrained optimization problems [14]; this allows us to envision several potential circuit implementations. Because we wish to solve a large set of coupled ODEs (in time), often approximating a PDE, we have different methods for modifying and accelerating the convergence of these analog systems. In particular, the time constant(s) for system convergence can continually vary because of global convergence of the analog physical system.

For many COMS problems, scaling the constrained linear optimization solvers will be challenging. In general, for m system inputs and n output variables, the physical solution typically requires $O(nm)$ computations per iteration and requires $O(nm)$ area to store the

constraints. In the analog environment, the same area required to store the constraints performs the resulting vector-matrix multiplication, so a parallel, mesh-based architecture is fairly natural in these approaches. The challenges of scaling become clear for an image or signal reconstruction problem where one might have m = 10,000 inputs (e.g., samples) and n = 1,000,000 unknowns (e.g., a 1-megapixel image or a few milliseconds of a 1-GHz bandwidth signal). This would result in over 10Gflops of computations per iteration and require over 10^9 analog storage elements. The resulting IC would be similar to a 40Gbit EEPROM storage device, which is close to the currently available (i.e., the 4Gbyte iPod Shuffle); therefore the approach would seem to be possible with the best technologies available over the next 2-3 years. The challenge is to develop algorithms over the next few years that would take advantage of this direct EEPROM-like scaling for the entire constrained linear optimization.

One would imagine two initial stages of development before scaling an architecture to these sizes. First, one would develop small constrained optimization algorithms that can be compiled in current analog signal processing hardware, namely the Large-Scale Field Programmable Analog Arrays that have been developed at the Georgia Institute of Technology [13,15]. Second, one would develop a moderate constrained optimization algorithm, for example to recover a signal of length 4,096 (which can be easily generated by existing hardware) from 128 samples. In this case, the number $n \times m$ of flops per iteration and memory elements is about 512,000. Ideally, one would gain fundamental insight from these studies to develop architectures capable of reconstructing higher-dimensional signals.

4.5 Conclusions

Just looking at digital implementations for the moment, a rough and optimistic estimate of the performance gains achievable can be obtained by supposing that the relative gains achieved by Kim et. al. for L1RLS can be duplicated for DS, and that a Cell-based FFT implementation can maintain the 36Gflop performance quoted above. Combining these factors yields a speedup of 6000*35.9/130* 0.22 = 7531x! Although we concede that this improvement seems quite optimistic, this helps supporting the claim that COMS computations that now take minutes with l_1-MAGIC can ultimately be performed in seconds or even fractions of seconds with currently available hardware.

With improvements in hardware performance that are certain to come over the few years to come, that speedup factor may not be so optimistic after all.

5 References

[1] 4DSP Inc. IEEE-754 floating point FFT/IFFT IP core. http://www.4dsp.com/fft.htm

[2] D. Donoho and Y. Tsaig. Fast solution of l_1-norm minimization problems when the solution may be sparse. Technical report, Department of Statistics, Stanford University, October 2006. http://www.stanford.edu/~tsaig/Papers/FastL1.pdf

[3] M. Frigo and S. Johnson. benchFFT (web site). http://www.fftw.org/benchfft

[4] N. Govindaraju and D. Manocha. GPUFFTW: High performance power-of-two FFT library using graphics processors. http://gamma.cs.unc.edu/GPUFFTW

[5] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. A method for large-scale l_1-regularized least squares problems with applications in signal processing and statistics. February 2007. Submitted to IEEE Journal on Selected Topics in Signal Processing. http://www.stanford.edu/~boyd/l1_logistic_reg.html

[6] J. Owens, S. Sengupta, and D. Horn. Assessment of graphic processing units (GPUs) for department of defense (DoD) digital signal processing (DSP) applications. Technical Report ECE-CE-2005-3, October 2005. http://graphics.cs.ucdavis.edu/publications/func/return_pdf?pub_id=866

[7] V. Pande et al. Folding@Home on the PS3, December 2006. http://folding.stanford.edu/FAQ-PS3.html

[8] V. Pande et al. Folding@Home on ATI GPU's: a major step forward, January 2007. http://folding.stanford.edu/FAQ-ATI.html

[9] Texas Instruments Incorporated. TMS320C67x DSP library programmer's reference guide (rev. B), March 2006. http://www.ti.com/litv/pdf/spru657b

[10] Texas Memory Systems, Inc. TM-44 function execution times. http://www.superdsp.com/products/xp30/tm44timings.pdf

[11] S. Williams, J. Shalf, L. Oliker, S. Kamil, P. Husbands, and K. Yelick. The potential of the Cell processor for scientific computing. In Proceedings of the 3rd ACM Conference on Computing Frontiers, pages 9–20, 2006. http://www.cs.berkeley.edu/~samw/projects/cell/CF06.pdf

[12] M. Fazel, H. Hindi, and S. Boyd. Log-det heuristic for matrix rank minimization with applications to Hankel and Euclidean distance matrices, In Proceedings of the American Control Conference, 2003. http://www.cds.caltech.edu/~maryam/acc03_final.html

[13] C. M. Twigg and P. E. Hasler. Large-scale FPAA devices for signal processing applications. In IEEE International Conference on Accoustics Speech and Signal Processing, 2007.

[14] D. W. Tank and J. J. Hopfield. Simple neural optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit. IEEE Transactions on Circuits and Systems, 33(5):533–541, 1986.

[15] C. M. Twigg and P. E. Hasler. A large-scale reconfigurable analog signal processor (RASP) IC. In IEEE Proceedings of the Custom Integrated Circuits Conference, pages 5–8, Sept. 2006. http://users.ece.gatech.edu/~ctwigg/papers/ctwigg_2006_cicc.pdf