Recovering Lost Sensor Data through Compressed Sensing

Zainul Charbiwala

Collaborators: Younghun Kim, Sadaf Zahedi, Supriyo Chakraborty, Ting He (IBM), Chatschik Bisdikian (IBM), Mani Srivastava
The Big Picture

Lossy Communication Link
The Big Picture

Lossy Communication Link
The Big Picture

Lossy Communication Link
The Big Picture

Lossy Communication Link

How do we recover from this loss?
How do we recover from this loss?

- Retransmit the lost packets
The Big Picture

Lossy Communication Link

How do we recover from this loss?
- Retransmit the lost packets
How do we recover from this loss?

- Retransmit the lost packets
- Proactively encode the data with some protection bits
The Big Picture

How do we recover from this loss?

- Retransmit the lost packets
- Proactively encode the data with some protection bits
The Big Picture

How do we recover from this loss?

- Retransmit the lost packets
- Proactively encode the data with some protection bits
The Big Picture

How do we recover from this loss?

• Retransmit the lost packets
• Proactively encode the data with some protection bits
• Can we do something better?
The Big Picture - Using Compressed Sensing
The Big Picture - Using Compressed Sensing
The Big Picture - Using Compressed Sensing
The Big Picture - Using Compressed Sensing

Generate Compressed Measurements

Lossy Communication Link

CSEC
The Big Picture - Using Compressed Sensing

Generate Compressed Measurements

Lossy Communication Link

CSEC
The Big Picture - Using Compressed Sensing

How does this work?
The Big Picture - Using Compressed Sensing

How does this work?
- Use knowledge of signal model and channel
The Big Picture - Using Compressed Sensing

How does this work?

- Use knowledge of signal model and channel
- CS uses randomized sampling/projections
How does this work?

- Use knowledge of signal model and channel
- CS uses randomized sampling/projections
- Random losses look like additional randomness!
How does this work?

- Use knowledge of signal model and channel
- CS uses randomized sampling/projections
- Random losses look like additional randomness!

Rest of this talk focuses on describing “How” and “How Well” this works
Talk Outline

- A Quick Intro to Compressed Sensing
- CS Erasure Coding for Recovering Lost Sensor Data
- Evaluating CSEC’s cost and performance
- Concluding Remarks
Why Compressed Sensing?

- Physical Signal
- Sampling
- Compression
- Communication
- Application

Computationally expensive
Why Compressed Sensing?

Computationally expensive

Shifts computation to a capable server
Compressed Sensing - Some Intuition
Compressed Sensing - Some Intuition

How do you acquire this signal?
Compressed Sensing - Some Intuition

How do you acquire this signal?
- Nyquist rate - twice the bandwidth
Compressed Sensing - Some Intuition

How do you acquire this signal?

- Nyquist rate - twice the bandwidth
- But what if you knew more about the signal?
Compressed Sensing - Some Intuition

How do you acquire this signal?

- Nyquist rate - twice the bandwidth
- But what if you knew more about the signal?
Compressed Sensing - Some Intuition

How do you acquire this signal?

- Nyquist rate - twice the bandwidth
- But what if you knew more about the signal?
- CS enables signal acquisition based on information content
Transform Domain Analysis
Transform Domain Analysis

- We usually acquire signals in the time or spatial domain
Transform Domain Analysis

- We usually acquire signals in the time or spatial domain
- By looking at the signal in another *domain*, the signal may be represented more compactly
Transform Domain Analysis

- We usually acquire signals in the time or spatial domain.
- By looking at the signal in another **domain**, the signal may be represented more compactly.

- Eg: a sine wave can be expressed by 3 parameters: frequency, amplitude and phase.
Transform Domain Analysis

- We usually acquire signals in the time or spatial domain.
- By looking at the signal in another *domain*, the signal may be represented more compactly.

  - Eg: a sine wave can be expressed by 3 parameters: frequency, amplitude and phase.
  - Or, in this case, by the index of the FFT coefficient and its complex value.
Transform Domain Analysis

- We usually acquire signals in the time or spatial domain
- By looking at the signal in another **domain**, the signal may be represented more compactly

- Eg: a sine wave can be expressed by 3 parameters: frequency, amplitude and phase.
- Or, in this case, by the index of the FFT coefficient and its complex value
- Sine wave is *sparse* in frequency domain

![Graph of sine wave](image)
Acquiring a Sine Wave
Acquiring a Sine Wave

- Assume we’re interested in acquiring a single sine wave $x(t)$ in a noiseless environment
Acquiring a Sine Wave

- Assume we’re interested in acquiring a single sine wave \( x(t) \) in a noiseless environment.
- An infinite duration sine wave can be expressed using three parameters: frequency \( f \), amplitude \( a \) and phase \( \phi \).
Acquiring a Sine Wave

- Assume we’re interested in acquiring a single sine wave $x(t)$ in a noiseless environment.

- An infinite duration sine wave can be expressed using three parameters: frequency $f$, amplitude $a$ and phase $\phi$.

- Question: What’s the best way to find the parameters?
Acquiring a Sine Wave
Acquiring a Sine Wave

Technically, to estimate three parameters one needs three good measurements
Acquiring a Sine Wave

- Technically, to estimate three parameters one needs three good measurements

- Questions:
Acquiring a Sine Wave

- Technically, to estimate three parameters one needs three good measurements

- Questions:
  - What are “good” measurements?
Acquiring a Sine Wave

- Technically, to estimate three parameters one needs three good measurements

Questions:
- What are “good” measurements?
- How do you estimate $f, a, \phi$ from three measurements?
Compressed Sensing
Compressed Sensing

- With three samples: \( z_1, z_2, z_3 \) of the sine wave at times \( t_1, t_2, t_3 \)
Compressed Sensing

- With three samples: $z_1, z_2, z_3$ of the sine wave at times $t_1, t_2, t_3$
- We know that any solution of $f, a$ and $\phi$ must meet the three constraints and spans a 3D space:
Compressed Sensing

- With three samples: $z_1, z_2, z_3$ of the sine wave at times $t_1, t_2, t_3$
- We know that any solution of $f, a$ and $\phi$ must meet the three constraints and spans a 3D space:

$$z_i = x(t_i) = a \sin(2\pi ft_i + \phi)$$
$$\forall i \in \{1, 2, 3\}$$
Compressed Sensing

- With three samples: $z_1, z_2, z_3$ of the sine wave at times $t_1, t_2, t_3$
- We know that any solution of $f, a$ and $\phi$ must meet the three constraints and spans a 3D space:
  \[ z_i = x(t_i) = a\sin(2\pi ft_i + \phi) \]
  \[ \forall i \in \{1,2,3\} \]
- Feasible solution space is much smaller
Compressed Sensing

- With three samples: $z_1, z_2, z_3$ of the sine wave at times $t_1, t_2, t_3$

- We know that any solution of $f, a$ and $\phi$ must meet the three constraints and spans a 3D space:

  $$ z_i = x(t_i) = a \sin(2\pi ft_i + \phi) $$

  $$ \forall i \in \{1, 2, 3\} $$

- Feasible solution space is much smaller

- As the number of constraints grows from more measurements, the feasible solution space shrinks

- Exhaustive search over this space reveals the right answer knowing presence of one sine
We could also represent $f, a$ and $\phi$ as a very long, but mostly empty FFT coefficient vector.
Formulating the Problem

- We could also represent $f$, $a$ and $\phi$ as a very long, but mostly empty FFT coefficient vector.
We could also represent $f, a$ and $\phi$ as a very long, but mostly empty FFT coefficient vector.
Formulating the Problem

- We could also represent $f$, $a$ and $\phi$ as a very long, but mostly empty FFT coefficient vector.
Formulating the Problem

- We could also represent $f$, $a$ and $\phi$ as a very long, but mostly empty FFT coefficient vector.
Formulating the Problem

- We could also represent $f$, $a$ and $\phi$ as a very long, but mostly empty FFT coefficient vector.

\[
y = ae^{-j2\pi ft + \phi}
\]

Sine wave. Amplitude represented by color.

\[
\Psi \quad \text{(Fourier Transform)}
\]
Sampling Matrix

- We could also write out the sampling process in matrix form
Sampling Matrix

We could also write out the sampling process in matrix form
We could also write out the sampling process in matrix form.
We could also write out the sampling process in matrix form:

\[ Z \Phi = \chi \]
We could also write out the sampling process in matrix form:

\[ Z = \Phi \chi \]

Three non-zero entries at some "good" locations.
We could also write out the sampling process in matrix form.

\[ Z = \Phi \chi \]

Three non-zero entries at some "good" locations.
Sampling Matrix

- We could also write out the sampling process in matrix form.

\[ Z = \Phi \]

Three non-zero entries at some “good” locations
Exhaustive Search

- Objective of exhaustive search:
  - Find an estimate of the vector $y$ that meets the constraints and is the most compact representation of $x$ (also called the sparsest representation)

- Our search is now guided by the fact that $y$ is a sparse vector

- Rewriting constraints:

$$z = \Phi x$$

$$y = \Psi x$$

$$z = \Phi \Psi^{-1} y$$

Constraints from measurements
Exhaustive Search

Objective of exhaustive search:

Find an estimate of the vector $y$ that meets the constraints and is the most compact representation of $x$ (also called the sparsest representation)

Our search is now guided by the fact that $y$ is a sparse vector

Rewriting constraints:

\[
\begin{align*}
    z &= \Phi x \\
    y &= \Psi x \\
    z &= \Phi \Psi^{-1} y
\end{align*}
\]

Constraints from measurements

\[
\begin{align*}
    \hat{y} &= \arg\min_{\tilde{y}} \|\tilde{y}\|_{\ell_0} \\
    \text{s.t.} & \quad z = \Phi \Psi^{-1} \tilde{y} \\
    \|y\|_{\ell_0} & \triangleq |\{i : y_i \neq 0\}| 
\end{align*}
\]
Exhaustive Search

- Objective of exhaustive search:
  - Find an estimate of the vector $y$ that meets the constraints and is the most compact representation of $x$ (also called the sparsest representation)

- Our search is now guided by the fact that $y$ is a sparse vector

- Rewriting constraints:
  \[
  \begin{align*}
  z &= \Phi x \\
  y &= \Psi x \\
  z &= \Phi \Psi^{-1} y
  \end{align*}
  \]

- Constraints from measurements

\[
\hat{y} = \arg\min_{\tilde{y}} \|\tilde{y}\|_{\ell_0}
\]

s.t. $z = \Phi \Psi^{-1} \tilde{y}$

$\|y\|_{\ell_0} \triangleq |\{i : y_i \neq 0\}|$

This optimization problem is NP-Hard!
**$l_1$ Minimization**

- Approximate the $l_0$ norm to an $l_1$ norm

\[
\hat{y} = \arg \min \|\tilde{y}\|_{l_1}
\]

\[
\|y\|_{l_1} = \sum_i |y_i|
\]

- This problem can now be solved efficiently using linear programming techniques

- This approximation was not new

- The big leap in Compressed Sensing was a theorem that showed that **under the right conditions, this approximation is exact!**
Some CS Results

- **Theorem**: If $k$ samples of a length $n$ signal are acquired uniformly randomly (if each sample is equiprobable) and reconstruction is performed in the Fourier basis:

\[ s \leq C' \cdot \frac{k}{\log^4(n)} \quad \text{w.h.p.} \]

[Rudelson06]

- Where $s$ is the sparsity of the signal
Handling Missing Data - Traditional Approach

Physical Signal → Sampling → Compression

\[ x \in \mathbb{R}^n \]
\[ z = I_n x \]
\[ y = \Psi_z \]

Compressed domain samples
Handling Missing Data - Traditional Approach

\[
x \in \mathbb{R}^n \quad z = \mathbf{I}_n x \quad y = \Psi z_{n \times n}
\]

When communication channel is lossy:

Missing samples
Handling Missing Data - Traditional Approach

When communication channel is lossy:
  • Use retransmissions to recover lost data
Handling Missing Data - Traditional Approach

\[ x \in \mathbb{R}^n \quad z = \mathbf{I}_n x \quad y = \Psi z \]

When communication channel is lossy:
- Use retransmissions to recover lost data
- Or, use error (erasure) correcting codes
Handling Missing Data - Traditional Approach

\[ x \in \mathbb{R}^n \]

\[ z = I_n x \]

\[ y = \Psi z_{\mathbb{R}^n} \]

Compressed domain samples

Communication

Missing samples
Handling Missing Data - Traditional Approach

\[ x \in \mathbb{R}^n \quad z = I_n x \quad y = \Psi z \]

[Diagram showing the process of handling missing data]

Physical Signal \rightarrow Sampling \rightarrow Compression \rightarrow Compressed domain samples

\rightarrow Channel Coding \rightarrow Communication \rightarrow Channel Decoding

\rightarrow Recovered compressed domain samples
Handling Missing Data - Traditional Approach

Physical Signal

Sampling

$z = I_n x$

$y = \Psi z$

$x \in \mathbb{R}^n$

Compressed domain samples

Compression

Channel Coding

$w = \Omega y$

$w_l = Cw$

Recovered compressed domain samples

$m \times n$

Communication

$z \in \mathbb{R}^n$

$m > n$

$\hat{y} = (C\Omega)^+ w_l$

Channel Decoding
Handling Missing Data - Traditional Approach

- Physical Signal
- Sampling: $x \in \mathbb{R}^n$
- $z = I_n x$
- Compression: $y = \Psi z$
- Done at application layer
- Channel Coding: $w = \Omega y$
- Communication: $w_l = Cw$
- Channel Decoding: $\hat{y} = (C\Omega)^+ w_l$
- Recovered compressed domain samples

$m \times n$
$m > n$
Handling Missing Data - Traditional Approach

Physical Signal

Sampling

$z = I_n x$

Compression

$y = \Psi z$

Done at application layer

Channel Coding

$w = \Omega y$

$m \times n$

$m > n$

Done at physical layer

Can’t exploit signal characteristics

Communication

$w_l = Cw$

Channel Decoding

$\hat{y} = (C\Omega)^+ w_l$

Recovered compressed domain samples
CS Erasure Coding Approach

\[ x \in \mathbb{R}^n \]

Physical Signal

Compressive Sampling

\[ z = \Phi x \]

Compressed domain samples

Communication

Decoding

\[ y = \arg \min_{\tilde{y}} \| \tilde{y} \|_{\ell_1} \]

\[ z_l = Cz \]

\[ z_l = C\Phi \Psi^{-1} \tilde{y} \]

s.t. \[ z_l \in \mathbb{R}^n \]

k x n

k < n
CS Erasure Coding Approach

Physical Signal

Compressive Sampling

$z = \Phi x$

$z_l = Cz$

Communication

Decoding

$y = \arg\min_{\tilde{y}} \|\tilde{y}\|_{\ell_1}$

s.t. $z_l = C\Phi\Psi^{-1}\tilde{y}$

Compressed domain samples

$z \in \mathbb{R}^n$

$z = \Phi x$

$z_l = Cz$

$y = \arg\min_{\tilde{y}} \|\tilde{y}\|_{\ell_1}$

s.t. $z_l = C\Phi\Psi^{-1}\tilde{y}$

Compressed domain samples

$z \in \mathbb{R}^n$

$z = \Phi x$

$z_l = Cz$

$y = \arg\min_{\tilde{y}} \|\tilde{y}\|_{\ell_1}$

s.t. $z_l = C\Phi\Psi^{-1}\tilde{y}$
CS Erasure Coding Approach

Physical Signal → Compressive Sampling → Communication → Decoding

\[ x \in \mathbb{R}^n \]

\[ z = \Phi x \]

\[ z_d = Cz \]

\[ y = \arg \min_{\tilde{y}} \|\tilde{y}\|_{\ell_1} \]

s.t. \[ z_d = C\Phi\Psi^{-1}\tilde{y} \]

Over-sampling in CS is Erasure Coding!
Effects of Missing Samples on CS

\[ z = \Phi \ x \]
Effects of Missing Samples on CS

\[ z = \Phi \]  

Missing samples at the receiver
Effects of Missing Samples on CS

\[ z = \Phi x \]

- Missing samples at the receiver
- Same as missing rows in the sampling matrix
Effects of Missing Samples on CS

What happens if we over-sample?
Effects of Missing Samples on CS

What happens if we over-sample?
Effects of Missing Samples on CS

What happens if we over-sample?
Effects of Missing Samples on CS

\[ z = \Phi x \]

What happens if we over-sample?

- Can we recover the lost data?
Effects of Missing Samples on CS

What happens if we over-sample?

- Can we recover the lost data?
- How much over-sampling is needed?
Extending CS Results

- **Claim**: When $m > k$ samples are acquired uniformly randomly and communicated through a memoryless binary erasure channel that drops $m-k$ samples, the received $k$ samples are still equiprobable.

- Implies that bound on sparsity condition should hold.

- If bound is tight, over-sampling rate ($m-k$) is same as loss rate

[This paper]
Features of CS Erasure Coding

- No need of additional channel coding block
- Redundancy achieved by oversampling
- Recovery is resilient to incorrect channel estimates
  - Traditional channel coding fails if redundancy is inadequate
- Decoding is free if CS was used for compression anyway
Features of CS Erasure Coding

- No need of additional channel coding block
- Redundancy achieved by oversampling
- Recovery is resilient to incorrect channel estimates
- Traditional channel coding fails if redundancy is inadequate
- Decoding is free if CS was used for compression anyway

Intuition:
- Channel Coding spreads information out over measurements
- Compression (Source Coding) - compact information in few measurements
- CSEC - spreads information while compacting!
Signal Recovery Performance Evaluation

Create Signal → CS Sampling → Lossy Channel → CS Recovery → Reconstruction Error?
In Memoryless Channels

Baseline performance - No Loss

Pr[Exact Recovery] vs. Sparsity (s)

- No Loss
- 20% Loss
- CSEC 7% Oversampling
- CSEC 15% Oversampling
- CSEC 20% Oversampling
In Memoryless Channels

Baseline performance - No Loss

20 % Loss - Drop in recovery probability
In Memoryless Channels

Baseline performance - No Loss

20% Oversampling - complete recovery

20% Loss - Drop in recovery probability

Pr[ Exact Recovery ]

No Loss
20% Loss
CSEC 7% Oversampling
CSEC 15% Oversampling
CSEC 20% Oversampling

Sparsity (s)
In Memoryless Channels

Baseline performance - No Loss

20% Oversampling - complete recovery

20% Loss - Drop in recovery probability

Less than 20% Oversampling - recovery does not fail completely

Pr[Exact Recovery] vs. Sparsity (s)
In Bursty Channels

Baseline performance - No Loss

Pr[Exact Recovery] vs Sparsity (s)

- No Loss – baseline
- 20% Loss
- CSEC 20% Oversampling
- CSEC+Interleaving
In Bursty Channels

Baseline performance - No Loss

20% Loss - Drop in recovery probability

Pr[ Exact Recovery ]

Sparsity (s)

No Loss – baseline
20% Loss
CSEC 20% Oversampling
CSEC+Interleaving
In Bursty Channels

Baseline performance - No Loss

20% Loss - Drop in recovery probability

20% Oversampling - doesn’t recover completely

Pr[Exact Recovery]
In Bursty Channels

Baseline performance - No Loss

20% Loss - Drop in recovery probability

20% Oversampling - doesn’t recover completely

Oversampling + Interleaving - Still incomplete recovery

Pr[ Exact Recovery ]

Sparsity (s)

No Loss
20% Loss
CSEC 20% Oversampling
CSEC+Interleaving
In Bursty Channels

- Recovery incomplete because of low interleaving depth
- Recovery better at high sparsity because bursty channels deliver bigger packets on average, but with higher variance
In Bursty Channels

- Recovery incomplete because of low interleaving depth
- Recovery better at high sparsity because bursty channels deliver bigger packets on average, but with higher variance

Graph showing the impact of oversampling and interleaving on recovery probability. The graph compares different loss scenarios and oversampling methods. The conclusion is that oversampling alone is not sufficient for complete recovery in bursty channels, and interleaving is needed to improve recovery rates.
In Real 802.15.4 Channel

Baseline performance - No Loss

Pr[Exact Recovery] vs Sparsity (s)

- No Loss
- 15% Loss
- CSEC 6% Oversampling
- CSEC 11% Oversampling
- CSEC 15% Oversampling
In Real 802.15.4 Channel

Baseline performance - No Loss

15% Loss - Drop in recovery probability

Pr[Exact Recovery] vs. Sparsity (s)

- No Loss
- 15% Loss
- CSEC 6% Oversampling
- CSEC 11% Oversampling
- CSEC 15% Oversampling
In Real 802.15.4 Channel

Baseline performance - No Loss

15% Loss - Drop in recovery probability

15% Oversampling - complete recovery
In Real 802.15.4 Channel

Baseline performance - No Loss

15% Loss - Drop in recovery probability

15% Oversampling - complete recovery

Less than 15% Oversampling - recovery does not fail completely
Cost of CSEC

- No robustness guarantees
Cost of CSEC

No robustness guarantees

All options equally robust (w.h.p.)
Cost of CSEC

No robustness guarantees

All options equally robust (w.h.p.)

2.5x lower energy

No robustness guarantees

All options equally robust (w.h.p.)

Energy/block (mJ)

m=256  S-n-S
m=10   C-n-S
m=64   CS
k=320  S-n-S+RS
k=16   C-n-S+RS
k=80   CSEC

Sense and Send
Sense, Compress (FFT) and Send
CS and Send (1/4th rate)
Sense and Send with Reed Solomon
Sense, Compress and Send with RS
CSEC and Send
Summary

- Oversampling is a valid erasure coding strategy for compressive reconstruction
- For binary erasure channels, an oversampling rate equal to loss rate is sufficient
- CS erasure coding can be rate-less like fountain codes
  - Allows adaptation to varying channel conditions
- Can be computationally more efficient on transmit side than traditional erasure codes
Closing Remarks

- CSEC spreads information out while compacting
  - No free lunch syndrome: Data rate requirement is higher than if using good source and channel coding independently
  - But, then, computation cost is higher too
- CSEC requires knowledge of signal model
  - If signal is non-stationary, model needs to be updated during recovery
  - This can be done using over-sampling too
- CSEC requires knowledge of channel conditions
  - Can use CS streaming with feedback
Thank You