Making Do with Less: An Introduction to Compressed Sensing 6 Frequency Analysis

Kurt Bryan

July 14, 2023

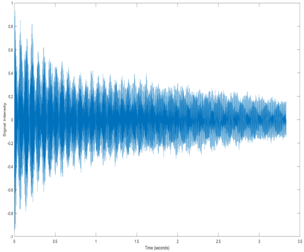
Inspiration

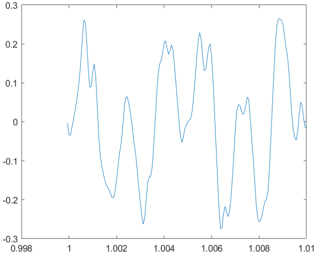
I have a metal gong in my kitchen:

It makes a pleasing sound when you hit it.

Inspiration

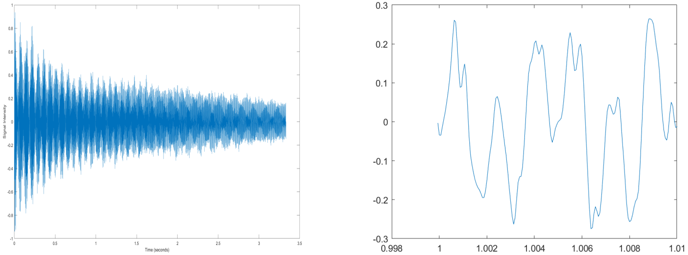
A plot of the sound intensity as captured by my iPhone:





Inspiration

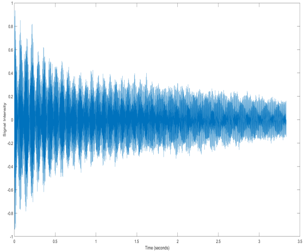
A plot of the sound intensity as captured by my iPhone:

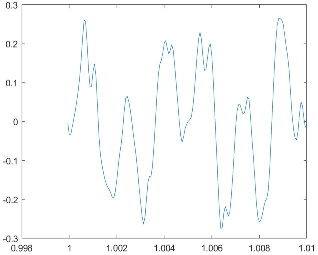


Goal 1: Analyze the frequencies present in this audio signal.

Inspiration

A plot of the sound intensity as captured by my iPhone:





Goal 1: Analyze the frequencies present in this audio signal.
Goal 2: Use compressed sensing to make the process more efficient.

Taylor Series

In calculus you may have seen that a function $f(t)$ can often be approximated by a polynomial

$$f(t) \approx a_0 + a_1 t + a_2 t^2 + a_3 t^3 + \cdots a_n t^n$$

if the a_k are chosen as $a_k = f^{(k)}(0)/k!$. This is a *Taylor polynomial* for f.

Taylor Series

In calculus you may have seen that a function $f(t)$ can often be approximated by a polynomial

$$f(t) \approx a_0 + a_1 t + a_2 t^2 + a_3 t^3 + \cdots a_n t^n$$

if the a_k are chosen as $a_k = f^{(k)}(0)/k!$. This is a *Taylor polynomial* for f.

In the limit $n \to \infty$ we get a *Taylor series* for $f(t)$, and for many functions $f(t)$ the series actually converges to the function.

Taylor Series

In calculus you may have seen that a function $f(t)$ can often be approximated by a polynomial

$$f(t) \approx a_0 + a_1 t + a_2 t^2 + a_3 t^3 + \cdots a_n t^n$$

if the a_k are chosen as $a_k = f^{(k)}(0)/k!$. This is a *Taylor polynomial* for f.

In the limit $n \to \infty$ we get a *Taylor series* for $f(t)$, and for many functions $f(t)$ the series actually converges to the function.

But polynomials aren't the only way to approximate functions...

Fourier Series

A good chunk of mathematics and engineering is based on one of the great ideas of 19th century mathematics:

> *Functions can be decomposed into sums of sines and/or cosines of various frequencies.*

Fourier Cosine Series

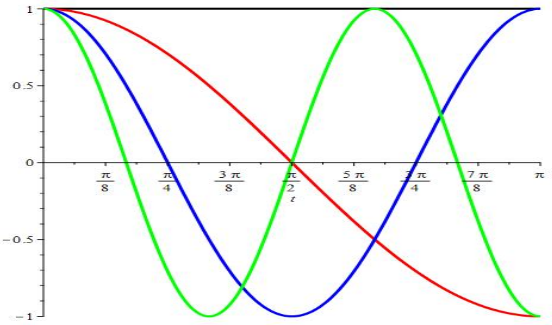
Fourier series come in many flavors. We'll be interested in *Fourier Cosine Series*: Any reasonable function $f(t)$ defined on $[0, \pi]$ can be well-approximated as a sum of cosines,

$$
\begin{aligned}
f(t) \approx \; & a_0 \\
& + a_1 \cos(t) \\
& \quad + a_2 \cos(2t) \\
& \qquad + \cdots \\
& \qquad\quad + a_N \cos(Nt)
\end{aligned}
$$

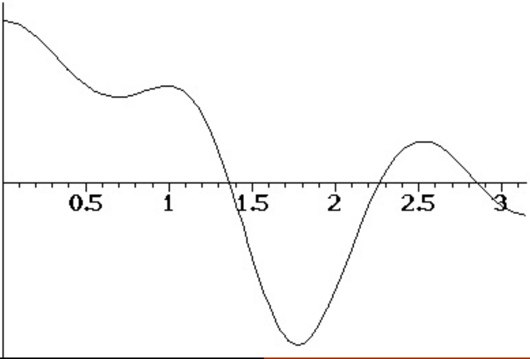
if we pick the a_k correctly (and take N large enough).

Fourier Cosine Series

The functions $\cos(0t), \cos(t), \cos(2t), \cos(3t)$ on $0 \leq t \leq \pi$ look like

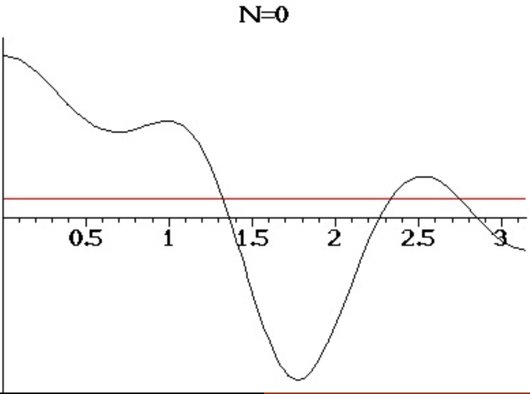


A Function to Approximate



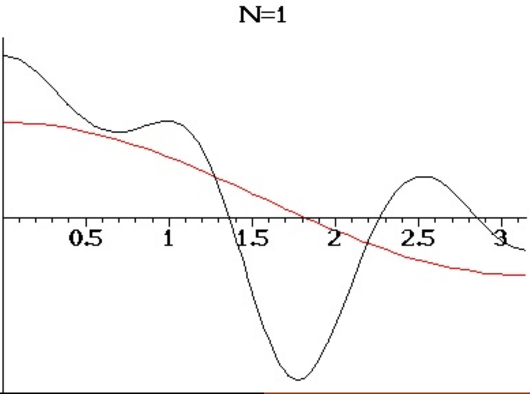
Cosine Series Example

$f(t) \approx 4.70$



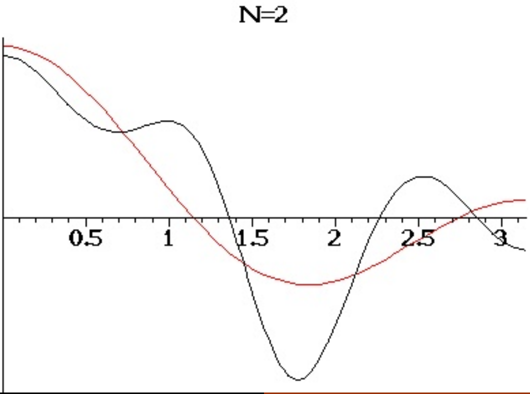
Cosine Series Example

$$f(t) \approx 4.70 + 19.1\cos(t)$$



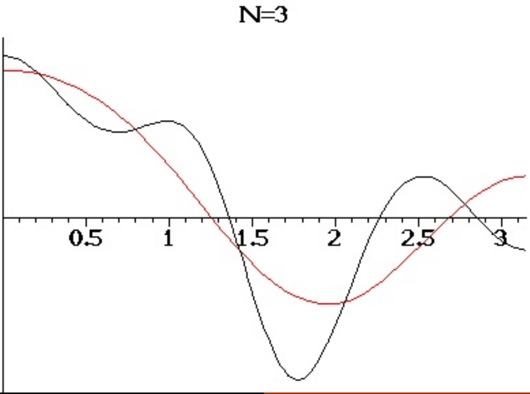
Cosine Series Example

$$f(t) \approx 4.70 + 19.1\cos(t) + 19.0\cos(2t)$$



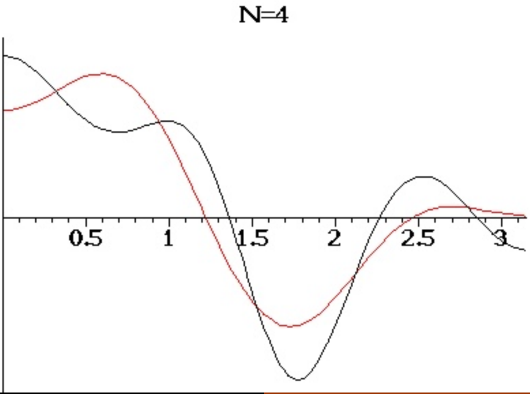
The Cosine Series

$$f(t) \approx 5.97 + 19.1\cos(t) + 19.0\cos(2t) - 5.88\cos(3t)$$



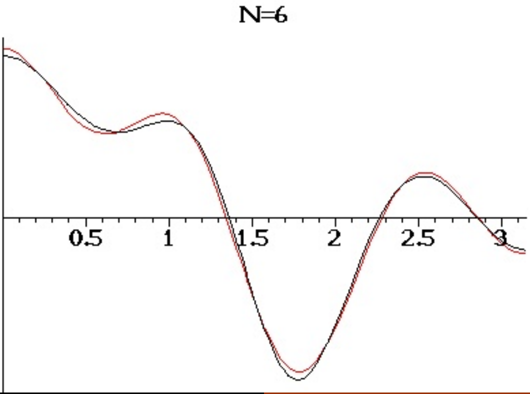
The Cosine Series

$$f(t) \approx 5.97 + 19.1\cos(t) + 19.0\cos(2t) - 5.88\cos(3t) - 9.92\cos(4t)$$



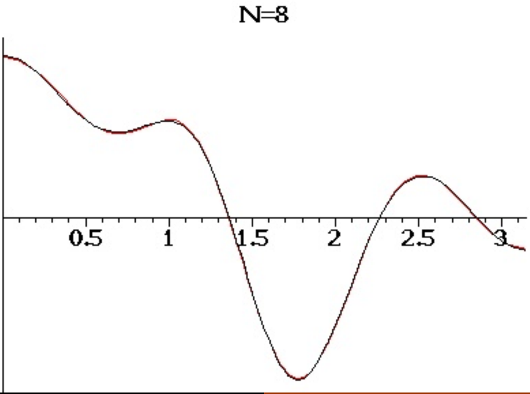
The Cosine Series

$$+ \cdots + 12.4\cos(5t) + 2.97\cos(6t)$$



The Cosine Series

$$+\cdots - 1.70\cos(7t) - 0.53\cos(8t)$$



The Cosine Coefficients

It can be shown that we must take

$$a_k \;=\; \frac{2}{\pi}\int_0^{\pi} f(t)\cos(kt)\,dt$$

The Cosine Coefficients

It can be shown that we must take

$$a_k = \frac{2}{\pi}\int_0^{\pi} f(t)\cos(kt)\,dt$$

If f and f' are bounded and piecewise continuous functions on $[0,\pi]$ then

$$f(t) \approx \frac{a_0}{2} + a_1\cos(t) + a_2\cos(2t) + \cdots + a_N\cos(Nt)$$

and as $N \to \infty$ the sum converges to $f(t)$ for any t at which f is continuous.

Cosine Series on More General Intervals

On a more general interval $0 \leq t \leq T$ we can write

$$f(t) = \frac{a_0}{2} + a_1 \cos(\pi t/T) + a_2 \cos(2\pi t/T) + \cdots + a_k \cos(k\pi t/T) + \cdots$$

by taking

Cosine Series on More General Intervals

On a more general interval $0 \leq t \leq T$ we can write

$$f(t) = \frac{a_0}{2} + a_1 \cos(\pi t/T) + a_2 \cos(2\pi t/T) + \cdots + a_k \cos(k\pi t/T) + \cdots$$

by taking

$$a_k = \frac{2}{T} \int_0^T f(t) \cos(k\pi t/T)\, dt.$$

Cosine Series on More General Intervals

On a more general interval $0 \leq t \leq T$ we can write

$$f(t) = \frac{a_0}{2} + a_1 \cos(\pi t/T) + a_2 \cos(2\pi t/T) + \cdots + a_k \cos(k\pi t/T) + \cdots$$

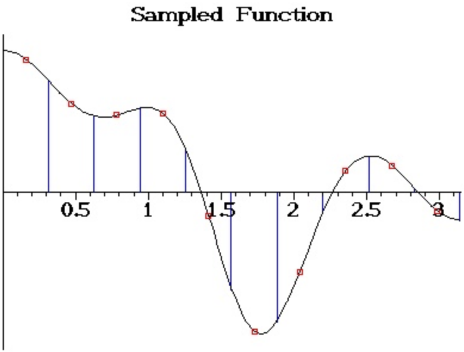
by taking

$$a_k = \frac{2}{T}\int_0^T f(t)\cos(k\pi t/T)\,dt.$$

Since $\cos(k\pi t/T)$ oscillates at frequency $\frac{k}{2T}$ Hz, a_k quantifies how much of this frequency is present.

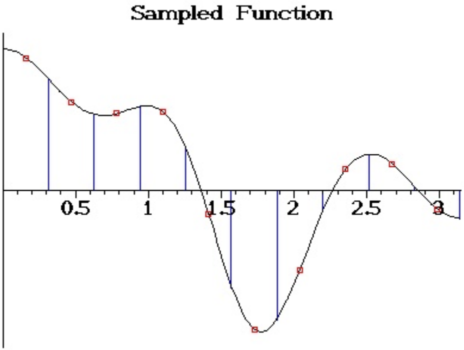
Sampling and Discretization

Signals aren't presented to the computer as functions, but as sampled function values:



Sampling and Discretization

We might break $[0, T]$ into N subintervals, sample f at each interval midpoint $t_0, t_1, \ldots, t_{N-1}$, by taking $t_i = \frac{(i+1/2)T}{N}$:



Sampling and Discretization

Replace the function $f(t)$ with the sampled version of f, the vector

$$\mathbf{f} = \langle f(t_0), f(t_1), \ldots, f(t_{N-1}) \rangle,$$

or more succinctly

$$\mathbf{f} = \langle f_0, f_1, \ldots, f_{N-1} \rangle$$

where $f_i = f(t_i)$ is the function sampled at time $t = t_i$.

Sampling and Discretization

We had

$$f(t) = \sum_k a_k \cos(k\pi t/T)$$

with

$$a_k = \frac{2}{T}\int_0^T f(t)\cos(k\pi t/T)\,dt.$$

Sampling and Discretization

We had

$$f(t) = \sum_k a_k \cos(k\pi t/T)$$

with

$$a_k = \frac{2}{T}\int_0^T f(t)\cos(k\pi t/T)\,dt.$$

Now we have a sampled version of $f(t)$, a vector $\mathbf{f}$, to express as a sum.

Sampling and Discretization

We had

$$f(t) = \sum_k a_k \cos(k\pi t/T)$$

with

$$a_k = \frac{2}{T}\int_0^T f(t)\cos(k\pi t/T)\,dt.$$

Now we have a sampled version of $f(t)$, a vector $\mathbf{f}$, to express as a sum.

What takes the place of the functions $\cos(k\pi t/T)$? What formula holds for the coefficients a_k?

Sampling and Discretization

Replace each function $\cos(k\pi t/T)$ with its sampled version at times $t = t_0, \ldots, t_{N-1}$ to form vectors

$$\mathbf{v}_k = \langle \cos(k\pi t_0/T), \cos(k\pi t_1/T), \ldots, \cos(k\pi t_{N-1}/T) \rangle.$$

Sampling and Discretization

Replace each function $\cos(k\pi t/T)$ with its sampled version at times $t = t_0, \ldots, t_{N-1}$ to form vectors

$$\mathbf{v}_k = \langle \cos(k\pi t_0/T), \cos(k\pi t_1/T), \ldots, \cos(k\pi t_{N-1}/T) \rangle.$$

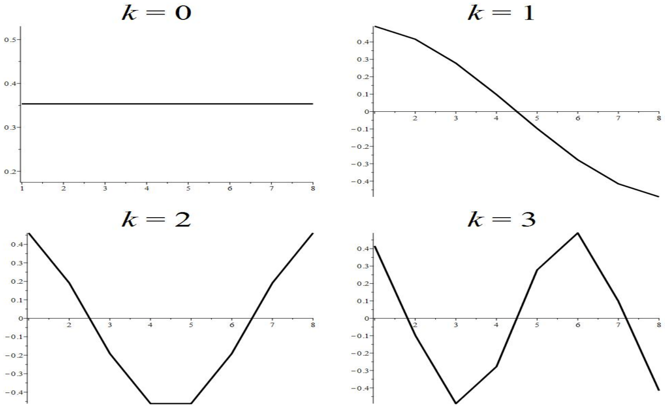
We'll then try to express

$$\mathbf{f} = \sum_k c_k \mathbf{v}_k$$

for some appropriate coefficients c_k.

Sampled Cosine Basis Vectors

The vectors $\mathbf{v}_k$ look like their analog $\cos(k\pi t/T)$ counterparts.



Symmetries and Discretization

The infinite sum

$$f(t) = \frac{a_0}{2} + a_1 \cos(\pi t/T) + a_2 \cos(2\pi t/T) + \cdots$$

is replaced by a sum

$$\mathbf{f} = c_0\mathbf{v}_0 + c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots$$

Symmetries and Discretization

The infinite sum

$$f(t) = \frac{a_0}{2} + a_1 \cos(\pi t/T) + a_2 \cos(2\pi t/T) + \cdots$$

is replaced by a sum

$$\mathbf{f} = c_0\mathbf{v}_0 + c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots$$

How do we compute the c_k? How many terms do we need?

Symmetries and Discretization

A bit of computation with some trig identities shows that the vectors

$$\mathbf{v}_0, \dots, \mathbf{v}_{N-1}$$

in $\mathbb{R}^N$ are pairwise orthogonal and so form an orthogonal basis for $\mathbb{R}^N$!

Symmetries and Discretization

A bit of computation with some trig identities shows that the vectors

$$\mathbf{v}_0, \dots, \mathbf{v}_{N-1}$$

in $\mathbb{R}^N$ are pairwise orthogonal and so form an orthogonal basis for $\mathbb{R}^N$!

Also, $\mathbf{v}_k = \mathbf{v}_{k+2N}$ and $\mathbf{v}_k = -\mathbf{v}_{k+N}$, so the vectors $\mathbf{v}_k$ outside the range $0 \leq k \leq N-1$ are redundant.

Symmetries and Discretization

Any vector $\mathbf{f}$ can be expressed as a linear combination

$$\mathbf{f} = c_0\mathbf{v}_0 + c_1\mathbf{v}_1 + \cdots + c_{N-1}\mathbf{v}_{N-1}$$

of the $\mathbf{v}_k$: Take the dot product of each side above with c_k to find $\mathbf{f} \cdot \mathbf{v}_k = c_k\mathbf{v}_k \cdot \mathbf{v}_k$ or

Symmetries and Discretization

Any vector $\mathbf{f}$ can be expressed as a linear combination

$$\mathbf{f} = c_0\mathbf{v}_0 + c_1\mathbf{v}_1 + \cdots + c_{N-1}\mathbf{v}_{N-1}$$

of the $\mathbf{v}_k$: Take the dot product of each side above with c_k to find $\mathbf{f} \cdot \mathbf{v}_k = c_k\mathbf{v}_k \cdot \mathbf{v}_k$ or

$$c_k = \frac{\mathbf{f} \cdot \mathbf{v}_k}{\mathbf{v}_k \cdot \mathbf{v}_k}.$$

Symmetries and Discretization

Any vector $\mathbf{f}$ can be expressed as a linear combination

$$\mathbf{f} = c_0\mathbf{v}_0 + c_1\mathbf{v}_1 + \cdots + c_{N-1}\mathbf{v}_{N-1}$$

of the $\mathbf{v}_k$: Take the dot product of each side above with c_k to find $\mathbf{f} \cdot \mathbf{v}_k = c_k\mathbf{v}_k \cdot \mathbf{v}_k$ or

$$c_k = \frac{\mathbf{f} \cdot \mathbf{v}_k}{\mathbf{v}_k \cdot \mathbf{v}_k}.$$

Summary: Any vector $\mathbf{f}$ in $\mathbb{R}^N$ can be decomposed into a sum of the $\mathbf{v}_k$ by taking the c_k as indicated.

Symmetries and Discretization

Any vector $\mathbf{f}$ can be expressed as a linear combination

$$\mathbf{f} = c_0\mathbf{v}_0 + c_1\mathbf{v}_1 + \cdots + c_{N-1}\mathbf{v}_{N-1}$$

of the $\mathbf{v}_k$: Take the dot product of each side above with c_k to find $\mathbf{f} \cdot \mathbf{v}_k = c_k\mathbf{v}_k \cdot \mathbf{v}_k$ or

$$c_k = \frac{\mathbf{f} \cdot \mathbf{v}_k}{\mathbf{v}_k \cdot \mathbf{v}_k}.$$

Summary: Any vector $\mathbf{f}$ in $\mathbb{R}^N$ can be decomposed into a sum of the $\mathbf{v}_k$ by taking the c_k as indicated.

And recall, $\mathbf{v}_k$ is a discrete version of $\cos(k\pi t/T)$ with frequency $k/(2T)$ Hz.

The Discrete Cosine Transform

It's most common to rescale the $\mathbf{v}_k$ to be unit vectors.

The Discrete Cosine Transform

It's most common to rescale the $\mathbf{v}_k$ to be unit vectors. In this case we can write

$$\mathbf{f} = \sum_{k=0}^{N-1} c_k \mathbf{v}_k$$

with

$$c_k = \mathbf{f} \cdot \mathbf{v}_k$$

for $0 \leq k \leq N-1$.

The Discrete Cosine Transform

It's most common to rescale the $\mathbf{v}_k$ to be unit vectors. In this case we can write

$$\mathbf{f} = \sum_{k=0}^{N-1} c_k \mathbf{v}_k$$

with

$$c_k = \mathbf{f} \cdot \mathbf{v}_k$$

for $0 \leq k \leq N-1$.

The computation of $c_0, \ldots, c_{N-1}$ from the dot products with $\mathbf{f}$ is called the *Discrete Cosine Transform* of $\mathbf{f}$; the reconstruction of $\mathbf{f}$ in the sum above is the *Inverse Discrete Cosine Transform*.

The DCT Basis Vectors in $\mathbb{R}^4$

For illustration, when $N = 4$ here are the $\mathbf{v}_k$ vectors:

$$
\begin{aligned}
\mathbf{v}_0 &= \langle 0.5, 0.5, 0.5, 0.5 \rangle \\
\mathbf{v}_1 &= \langle 0.6533, 0.2706, -0.2706, -0.6533 \rangle \\
\mathbf{v}_2 &= \langle 0.5, -0.5, -0.5, 0.5 \rangle \\
\mathbf{v}_3 &= \langle 0.2706, -0.6533, 0.6533, -0.2706 \rangle.
\end{aligned}
$$

The DCT Basis Vectors in $\mathbb{R}^4$

For illustration, when $N = 4$ here are the $\mathbf{v}_k$ vectors:

$$\begin{aligned}
\mathbf{v}_0 &= \langle 0.5, 0.5, 0.5, 0.5 \rangle \\
\mathbf{v}_1 &= \langle 0.6533, 0.2706, -0.2706, -0.6533 \rangle \\
\mathbf{v}_2 &= \langle 0.5, -0.5, -0.5, 0.5 \rangle \\
\mathbf{v}_3 &= \langle 0.2706, -0.6533, 0.6533, -0.2706 \rangle.
\end{aligned}$$

These 4 vectors form an orthonormal basis for $\mathbb{R}^4$.

The Discrete Cosine Transform

If $\mathbf{c} = \langle c_0, \ldots, c_{N-1} \rangle$ then the equation $\mathbf{f} = \sum_{k=0}^{N-1} c_k \mathbf{v}_k$ can be expressed as a matrix-vector multiplication

$$\mathbf{f} = \mathbf{M}\mathbf{c}$$

where $\mathbf{M}$ is the orthogonal matrix with columns $\mathbf{v}_0, \ldots, \mathbf{v}_{N-1}$.

The Discrete Cosine Transform

If $\mathbf{c} = \langle c_0, \ldots, c_{N-1} \rangle$ then the equation $\mathbf{f} = \sum_{k=0}^{N-1} c_k \mathbf{v}_k$ can be expressed as a matrix-vector multiplication

$$\mathbf{f} = \mathbf{M}\mathbf{c}$$

where $\mathbf{M}$ is the orthogonal matrix with columns $\mathbf{v}_0, \ldots, \mathbf{v}_{N-1}$.

We also have

$$\mathbf{c} = \mathbf{M}^{-1}\mathbf{f}.$$

Observations

Observations

- If $\mathbf{f}$ stems from a signal $f(t)$ sampled uniformly at N points on $0 \leq t \leq T$ then the sampling interval is T/N. The *sampling rate* is N/T Hz.

Observations

- If **f** stems from a signal $f(t)$ sampled uniformly at N points on $0 \leq t \leq T$ then the sampling interval is T/N. The *sampling rate* is N/T Hz.
- The function $\cos(k\pi t/T)$ makes $k/2$ oscillations on $0 \leq t \leq T$, a frequency of $k/(2T)$ Hz.

Observations

- If $\mathbf{f}$ stems from a signal $f(t)$ sampled uniformly at N points on $0 \leq t \leq T$ then the sampling interval is T/N. The *sampling rate* is N/T Hz.
- The function $\cos(k\pi t/T)$ makes $k/2$ oscillations on $0 \leq t \leq T$, a frequency of $k/(2T)$ Hz.
- The sampled version $\mathbf{v}_k$ thus corresponds to $k/(2T)$ Hz. The highest, $k = N - 1$, corresponds to $(N-1)/(2T) \approx N/(2T)$ Hz (if N is large).

Observations

- If $\mathbf{f}$ stems from a signal $f(t)$ sampled uniformly at N points on $0 \leq t \leq T$ then the sampling interval is T/N. The *sampling rate* is N/T Hz.
- The function $\cos(k\pi t/T)$ makes $k/2$ oscillations on $0 \leq t \leq T$, a frequency of $k/(2T)$ Hz.
- The sampled version $\mathbf{v}_k$ thus corresponds to $k/(2T)$ Hz. The highest, $k = N - 1$, corresponds to $(N-1)/(2T) \approx N/(2T)$ Hz (if N is large).
- The highest frequency we can estimate is $(N-1)/(2T)$, about half the sampling rate. This is called the *Nyquist Frequency*.

The Nyquist Sampling Criterion

This analysis gives rise to one of the mantras of traditional signal processing:

"In order to estimate the spectrum of a signal with frequency content from 0 to r Hz, we must sample at a frequency of at least $2r$ Hz."

The Nyquist Sampling Criterion

This analysis gives rise to one of the mantras of traditional signal processing:

"In order to estimate the spectrum of a signal with frequency content from 0 to r Hz, we must sample at a frequency of at least $2r$ Hz."

This is one reason for the traditional sampling rate of $44,100$ Hz used in CD's—in theory, frequencies up to $22,050$ Hz (up to and a bit above human hearing) can be reproduced.

DCT of the Gong Audio Signal

The gong audio signal consists of 16000 samples taken at a sampling rate of 16000 Hz, total signal length $T = 1$ second.

DCT of the Gong Audio Signal

The gong audio signal consists of 16000 samples taken at a sampling rate of 16000 Hz, total signal length $T = 1$ second.

We can compute c_k for $0 \leq k \leq 15999$ (c_k corresponds to frequency $k/(2T) = k/2$ here, highest frequency $15999/(2T) = 7999.5 \approx 8000$ Hz.

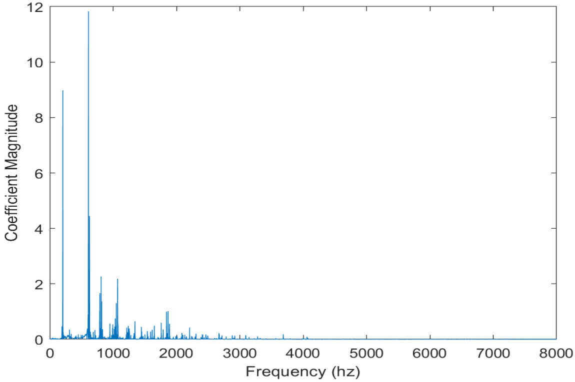
DCT of the Gong Audio Signal

The gong audio signal consists of 16000 samples taken at a sampling rate of 16000 Hz, total signal length $T = 1$ second.

We can compute c_k for $0 \leq k \leq 15999$ (c_k corresponds to frequency $k/(2T) = k/2$ here, highest frequency $15999/(2T) = 7999.5 \approx 8000$ Hz.

A plot of $|c_k|$ versus frequency $k/2$ is shown on the next slide.

DCT of the Gong Audio Signal



The Conventional Approach

The conventional approach to frequency analysis is this: We have a signal containing frequencies up to r Hz.

The Conventional Approach

The conventional approach to frequency analysis is this: We have a signal containing frequencies up to r Hz.

To resolve the frequency content of the signal we sample at uniform time intervals, at a frequency of at least $2r$ Hz. This is a lot of data if r is large.

The Conventional Approach

The conventional approach to frequency analysis is this: We have a signal containing frequencies up to r Hz.

To resolve the frequency content of the signal we sample at uniform time intervals, at a frequency of at least $2r$ Hz. This is a lot of data if r is large.

Is it possible to collect less data and still get what we need?

The Conventional Approach

Many signals contains only a few large frequencies of interest and everything else is small in magnitude (effectively zero).

The Conventional Approach

Many signals contains only a few large frequencies of interest and everything else is small in magnitude (effectively zero).

That is, the vector $\mathbf{c}$ encoding the DCT coefficients is sparse.

The Conventional Approach

Many signals contains only a few large frequencies of interest and everything else is small in magnitude (effectively zero).

That is, the vector **c** encoding the DCT coefficients is sparse.

Maybe we can leverage sparsity to our advantage? For example, sample less data.

The Conventional Approach

Many signals contains only a few large frequencies of interest and everything else is small in magnitude (effectively zero).

That is, the vector $\mathbf{c}$ encoding the DCT coefficients is sparse.

Maybe we can leverage sparsity to our advantage? For example, sample less data.

How then should we sample the signal?

The Conventional Approach

Many signals contains only a few large frequencies of interest and everything else is small in magnitude (effectively zero).

That is, the vector $\mathbf{c}$ encoding the DCT coefficients is sparse.

Maybe we can leverage sparsity to our advantage? For example, sample less data.

How then should we sample the signal? Randomly!

Example

Consider a signal

$$f(t) = 0.15\cos(23\pi t) - 0.53\cos(49\pi t) + 0.19\cos(234\pi t)$$

defined on the interval $0 \leq t \leq 1$.

Example

Consider a signal

$$f(t) = 0.15\cos(23\pi t) - 0.53\cos(49\pi t) + 0.19\cos(234\pi t)$$

defined on the interval $0 \leq t \leq 1$.

It's highest frequency is 117 Hz, dictating a traditional sampling rate of at least 234 Hz. That's 234 samples.

Example

Consider a signal

$$f(t) = 0.15\cos(23\pi t) - 0.53\cos(49\pi t) + 0.19\cos(234\pi t)$$

defined on the interval $0 \leq t \leq 1$.

It's highest frequency is 117 Hz, dictating a traditional sampling rate of at least 234 Hz. That's 234 samples.

Notice that $f(t)$ itself—or any sampled version thereof—is NOT sparse.

Example

Consider a signal

$$f(t) = 0.15\cos(23\pi t) - 0.53\cos(49\pi t) + 0.19\cos(234\pi t)$$

defined on the interval $0 \leq t \leq 1$.

It's highest frequency is 117 Hz, dictating a traditional sampling rate of at least 234 Hz. That's 234 samples.

Notice that $f(t)$ itself—or any sampled version thereof—is NOT sparse. But in the frequency domain f is sparse—it has only three nonzero frequencies.

Example

A general function $f(t)$ on $0 \leq t \leq 1$ has a Fourier Cosine series

$$f(t) = \sum_{k=0}^{\infty} c_k \cos(k\pi t).$$

Example

A general function $f(t)$ on $0 \leq t \leq 1$ has a Fourier Cosine series

$$f(t) = \sum_{k=0}^{\infty} c_k \cos(k\pi t).$$

Think of $\mathbf{c} = \langle c_0, c_1, \ldots, \rangle$ as a vector; truncate the sum at some large value N.

Example

A general function $f(t)$ on $0 \leq t \leq 1$ has a Fourier Cosine series

$$f(t) = \sum_{k=0}^{\infty} c_k \cos(k\pi t).$$

Think of $\mathbf{c} = \langle c_0, c_1, \ldots, \rangle$ as a vector; truncate the sum at some large value N. For our $f(t)$ the vector $\mathbf{c}$ is sparse.

Example

A general function $f(t)$ on $0 \leq t \leq 1$ has a Fourier Cosine series

$$f(t) = \sum_{k=0}^{\infty} c_k \cos(k\pi t).$$

Think of $\mathbf{c} = \langle c_0, c_1, \ldots, \rangle$ as a vector; truncate the sum at some large value N. For our $f(t)$ the vector $\mathbf{c}$ is sparse. Sample f at random times $t_1, t_2, \ldots, t_m$, so we have data

$$f(t_j) = \sum_{k=0}^{N-1} c_k \cos(k\pi t_j), \quad 1 \leq t \leq m.$$

Example

But

$$f(t_j) = \sum_{k=0}^{N-1} c_k \cos(k\pi t_j)$$

for $1 \leq t \leq m$ is a system of m equations in N unknowns, with a sparse solution vector $\mathbf{c} = \langle c_0, \ldots, c_{N-1} \rangle$!

Example

But

$$f(t_j) = \sum_{k=0}^{N-1} c_k \cos(k\pi t_j)$$

for $1 \leq t \leq m$ is a system of m equations in N unknowns, with a sparse solution vector $\mathbf{c} = \langle c_0, \ldots, c_{N-1} \rangle$!

We can try any CS algorithm to seek a sparse solution to $\mathbf{Ac} = \mathbf{d}$ where $d_j = f(t_j)$ and $\mathbf{A}$ is the $m \times N$ matrix with entries

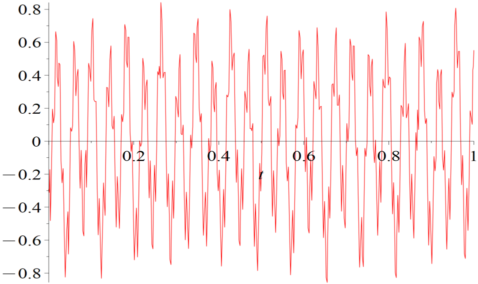
$$A_{jk} = \cos(k\pi t_j).$$

Example

To illustrate, let

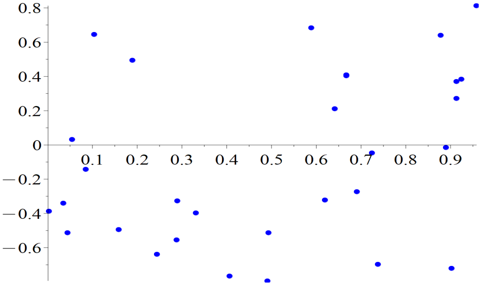
$$f(t) = 0.15\cos(23\pi t) - 0.53\cos(49\pi t) + 0.19\cos(234\pi t).$$

Then $c_{23} = 0.15$, $c_{49} = -0.53$, and $c_{234} = 0.19$.



Example

We sample $f(t)$ at 30 random times:



to generate data $\mathbf{d} = \langle f(t_1), \ldots, f(t_{30}) \rangle$.

Example

Solve $\mathbf{Ac} = \mathbf{d}$ with OMP (sparsity bound 5, cosine sum truncated at upper limit $N = 500$).

Example

Solve $\mathbf{Ac} = \mathbf{d}$ with OMP (sparsity bound 5, cosine sum truncated at upper limit $N = 500$).

The recovery is exact with close to 100 percent probability.

Example

Solve $\mathbf{Ac} = \mathbf{d}$ with OMP (sparsity bound 5, cosine sum truncated at upper limit $N = 500$).

The recovery is exact with close to 100 percent probability.

Can we do this with the gong audio signal?

Recovering the Gong Audio Signal

Forget the DCT—let's go back to continuous time.

Recovering the Gong Audio Signal

Forget the DCT—let's go back to continuous time.

In the analog continuous time world the gong signal on the time interval $0 \leq t \leq 1$ has a Fourier cosine series

$$f(t) = \sum_{k=0}^{\infty} c_k \cos(k\pi t).$$

Recovering the Gong Audio Signal

Forget the DCT—let's go back to continuous time.

In the analog continuous time world the gong signal on the time interval $0 \leq t \leq 1$ has a Fourier cosine series

$$f(t) = \sum_{k=0}^{\infty} c_k \cos(k\pi t).$$

But as we've seen, many of the c_k are quite small.

Recovering the Gong Audio Signal

Forget the DCT—let's go back to continuous time.

In the analog continuous time world the gong signal on the time interval $0 \leq t \leq 1$ has a Fourier cosine series

$$f(t) = \sum_{k=0}^{\infty} c_k \cos(k\pi t).$$

But as we've seen, many of the c_k are quite small.

Suppose we sample $f(t)$ at random times t_j, $1 \leq j \leq 2000$ to obtain data $f_j = f(t_j)$.

Recovering the Gong Audio Signal

Let's cap the sum for $f(t)$ at an upper limit of 8000 (corresponds to frequency 4000 Hz, should be high enough).

Recovering the Gong Audio Signal

Let's cap the sum for $f(t)$ at an upper limit of 8000 (corresponds to frequency 4000 Hz, should be high enough). We have

$$f(t) = \sum_{k=0}^{8000} c_k \cos(k\pi t).$$

Recovering the Gong Audio Signal

Let's cap the sum for $f(t)$ at an upper limit of 8000 (corresponds to frequency 4000 Hz, should be high enough). We have

$$f(t) = \sum_{k=0}^{8000} c_k \cos(k\pi t).$$

Sampling at times $t_1, \ldots, t_{2000}$ yields

$$f_j = \sum_{k=0}^{8000} c_k \cos(k\pi t_j).$$

Recovering the Gong Audio Signal

Let's cap the sum for $f(t)$ at an upper limit of 8000 (corresponds to frequency 4000 Hz, should be high enough). We have

$$f(t) = \sum_{k=0}^{8000} c_k \cos(k\pi t).$$

Sampling at times $t_1, \ldots, t_{2000}$ yields

$$f_j = \sum_{k=0}^{8000} c_k \cos(k\pi t_j).$$

Consider this a system of 2000 equations in 8001 unknowns, $c_0, \ldots, c_{8000}$.

Recovering the Gong Audio Signal

In matrix form we have $\mathbf{Ac} = \mathbf{f}$ where

$$\mathbf{c} = \langle c_0, c_1, \ldots, c_{8000} \rangle$$

is sparse (sort of),

$$\mathbf{f} = \langle f(t_1), \ldots, f(t_{2000}) \rangle$$

and the measurement matrix $\mathbf{A}$ is 2000×8001 with entries $A_{jk} = \cos(k\pi t_j)$.

Recovering the Gong Audio Signal

We solve $\mathbf{Ac} = \mathbf{d}$ with sparsity bound 500 using OMP.

Recovering the Gong Audio Signal

We solve $\mathbf{Ac} = \mathbf{d}$ with sparsity bound 500 using OMP.

Once we have an estimate of $\mathbf{c} = \langle c_0, \ldots, c_{8000} \rangle$ we can estimate

$$f(t) \approx \sum_{k=0}^{8000} c_k \cos(k\pi t)$$

for any times t we want.

Recovering the Gong Audio Signal

We solve $\mathbf{Ac} = \mathbf{d}$ with sparsity bound 500 using OMP.

Once we have an estimate of $\mathbf{c} = \langle c_0, \ldots, c_{8000} \rangle$ we can estimate

$$f(t) \approx \sum_{k=0}^{8000} c_k \cos(k\pi t)$$

for any times t we want.

See the Matlab code and listen to see how it works!

A One Pixel Camera

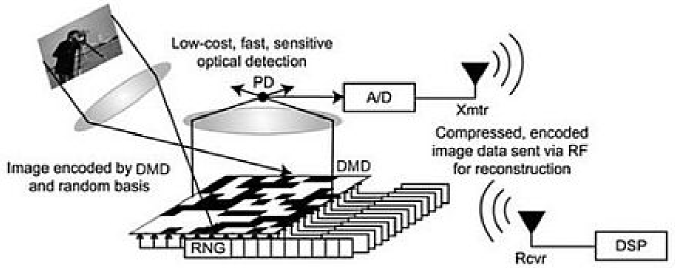
The CCD and CMOS chips in digital cameras are relatively cheap, on a per pixel basis—these materials easily convert photons at visual wavelengths into electrons. But infrared cameras cost much more per pixel.

A One Pixel Camera

The CCD and CMOS chips in digital cameras are relatively cheap, on a per pixel basis—these materials easily convert photons at visual wavelengths into electrons. But infrared cameras cost much more per pixel.

Can we find a way to generate reasonable resolution with fewer pixels (and hence lower cost)? Researchers at Rice University have demonstrated a "one pixel" camera!

A One Pixel Camera



Single Pixel Imaging and CS

Consider a grayscale ("black and white") image with n pixels, intensities $x_1, \ldots, x_n$.

Single Pixel Imaging and CS

Consider a grayscale ("black and white") image with n pixels, intensities $x_1, \ldots, x_n$.

The single pixel camera captures data $d_1, \ldots, d_m$ of the form

$$d_j = x_1 + x_4 + x_{27} + \cdots$$

for some random choice of the x_i.

Single Pixel Imaging and CS

Consider a grayscale ("black and white") image with n pixels, intensities $x_1, \ldots, x_n$.

The single pixel camera captures data $d_1, \ldots, d_m$ of the form

$$d_j = x_1 + x_4 + x_{27} + \cdots$$

for some random choice of the x_i.

It's the marble problem all over again!

Single Pixel Imaging and CS

Consider a grayscale ("black and white") image with n pixels, intensities $x_1, \ldots, x_n$.

The single pixel camera captures data $d_1, \ldots, d_m$ of the form

$$d_j = x_1 + x_4 + x_{27} + \cdots$$

for some random choice of the x_i.

It's the marble problem all over again!

We can write $\mathbf{d} = \mathbf{\Phi x}$.

Single Pixel Imaging and CS

The image $\mathbf{x}$ is not itself sparse, but it may be the transform (e.g., DCT) of a sparse vector $\mathbf{c}$ so

$$\mathbf{x} = \mathbf{M}\mathbf{c}.$$

To recover $\mathbf{x}$ from data $\mathbf{d} = \boldsymbol{\Phi}\mathbf{x}$ we

Single Pixel Imaging and CS

The image $\mathbf{x}$ is not itself sparse, but it may be the transform (e.g., DCT) of a sparse vector $\mathbf{c}$ so

$$\mathbf{x} = \mathbf{M}\mathbf{c}.$$

To recover $\mathbf{x}$ from data $\mathbf{d} = \mathbf{\Phi}\mathbf{x}$ we

1. Find a sparse solution $\mathbf{c}$ to $\mathbf{\Phi}\mathbf{M}\mathbf{c} = \mathbf{d}$ (note $\mathbf{\Phi}\mathbf{M}$ is a random $n \times N$ matrix)

Single Pixel Imaging and CS

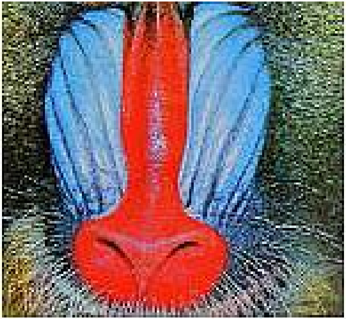
The image $\mathbf{x}$ is not itself sparse, but it may be the transform (e.g., DCT) of a sparse vector $\mathbf{c}$ so

$$\mathbf{x} = \mathbf{Mc}.$$

To recover $\mathbf{x}$ from data $\mathbf{d} = \mathbf{\Phi x}$ we

1. Find a sparse solution $\mathbf{c}$ to $\mathbf{\Phi Mc} = \mathbf{d}$ (note $\mathbf{\Phi M}$ is a random $n \times N$ matrix)
2. Reconstruct $\mathbf{x} = \mathbf{Mc}$.

One Pixel Image



Original image (left) ($N = 256 \times 256$ pixels), CS formed image based on $n = 6500$ total samples.