Transform Methods & Signal Processing

lecture 07

Matthew Roughan
<matthew.roughan@adelaide.edu.au>

Discipline of Applied Mathematics
School of Mathematical Sciences
University of Adelaide

October 26, 2009

The Fast Fourier Transform

... Its discrete version is computed by the Fast Fourier Transform, which is the most important algorithm of the last century.

Gilbert Strang, MIT

http://www.icase.edu/colloq-v3/data/colloq.Strang.Gilbert.2000.12.15.html

This lecture covers some of the theory that we have neglected so far. In particular: Fast Fourier Transforms, some theory related to random processes and the spectral representation, and the Parseval, Rayleigh and Plancheral theorems. We further consider the theoretical basis for the Fourier transform by considering how and why we could generalize it.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.1/77

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.2/77

Multiplication

How should we multiply large integers x and y?

http://en.wikipedia.org/wiki/Multiplication_algorithm

Karatsuba (1962): Divide and conquer! i.e. write

$$\begin{aligned} x &= x_1 W^m + x_2 \\ y &= y_1 W^m + y_2 \end{aligned}$$

And the product becomes

$$xy = x_1 y_1 W^{2m} + (x_1 y_2 + x_2 y_1) W^m + x_2 y_2$$

For example 1011×205 with $W = 10$ and $m = 2$

$$\left.\begin{aligned} 1011 &= 10 \times 10^2 + 11 \\ 205 &= 2 \times 10^2 + 5 \end{aligned}\right\} \Rightarrow x \times y = 20 \times 10^4 + (50+22)10^2 + 55$$

The trick can be used recursively (e.g. for $x_1 \times y_1$)

Fast Fourier Transform

- DFT, as presented so far takes a matrix multiplication
  - N^2 complex multiplies
- FFT is based on divide and conquer
  - much faster $O(N \log N)$
- Not quite as simple as Karatsuba
  - use symmetries of the transform

Assume that multiplications are the expensive computation. Note that the new calculation takes 4 multiplies, instead of 1. However, the new multiplication involves numbers with half as many digits, and so is potentially much simpler. For instance, if we reduce the multiplication down to a single digit times a single digit, we could use a fast table lookup to perform the multiplication.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.3/77

Could do the DFT by the matrix multiplication $X = Ax$, which is an $N \times N$ matrix times a $N \times 1$ vector.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.4/77

The Fast Fourier Transform

Could do the DFT by the matrix multiplication $X = Ax$.
This takes $O(N^2)$ operations, which could get quite large.
Cooley-Tukey (radix-2) algorithm is $O(N \log_2 N)$ for data of length 2^k later.

- FFT first use by Gauss
- being used in by physicists in X-ray scattering in 1940
- various other users in proprietary settings
- Cooley-Tookey, 1965
- versions not dependent on data length 2^k.

Fast Fourier Transform

FFT has many version, but here we do radix-2
Take advantage of symmetry properties in the data
Write the DFT and IDFT

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk} \qquad \text{and} \qquad x(n) = \frac{1}{N} \sum_{n=0}^{N-1} X(k) W_N^{-nk}$$

$0 \leq k < N$ and $W_N = e^{-i2\pi/N}$.

Symmetries

$$\begin{aligned} W_N^{k+N/2} &= -W_N^k \\ W_N^{k+N} &= W_N^k \end{aligned}$$

The FFT seems to have been invented a few times, reputedly before Cooley and Tukey, for instance see anecdotal reports at
`http://www.lns.cornell.edu/spr/2002-08/msg0043452.html`

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.5/77

To see the first property note

$$\begin{aligned} W_N^{k+N/2} &= e^{-i2\pi(k+N/2)/N} \\ &= e^{-i2\pi k/N - i2\pi/2} \\ &= e^{-i2\pi k/N - i\pi} \\ &= e^{-i2\pi k/N} e^{-i\pi} \\ &= -e^{-i2\pi k/N} \\ &= -W_N^k \end{aligned}$$

I leave the second one as an exercise.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.6/77

Radix-2

For $k < N/2$

$$\begin{aligned} X(k) &= \sum_{n=0}^{N-1} x(n) W_N^{nk} \\ &= \sum_{n=0}^{N/2-1} x(2n) W_N^{2nk} + \sum_{n=0}^{N/2-1} x(2n+1) W_N^{(2n+1)k} \\ &= \sum_{n=0}^{N/2-1} x(2n) W_{N/2}^{nk} + W_N^k \sum_{n=0}^{N/2-1} x(2n+1) W_{N/2}^{nk} \\ &= F(k) + W_N^k G(k) \end{aligned}$$

Where $F(k)$ is the DFT of the sequence $\{x(0), x(2), \ldots, x(2N-2)\}$, and $G(k)$ is the DFT of the sequence $\{x(1), x(3), \ldots, x(2N-1)\}$.

Radix-2

For $k >= N/2$, exploit the periodicity of the DFT, i.e. we know that $F(k+N/2) = F(k)$ and $G(k+N/2) = G(k)$, and $W_N^{k+N/2} = -W_N^k$ so that $X(k+N/2) = F(k) - W_N^k G(k)$, and the DFT can be computed by

$$\begin{aligned} X(k) &= F(k) + W_N^k G(k), \quad k = 0, 1, \ldots, N/2-1 \\ X(k+N/2) &= F(k) - W_N^k G(k), \quad k = 0, 1, \ldots, N/2-1 \end{aligned}$$

where $F(k)$ and $G(k)$ are the DFTs given by

$$\begin{aligned} F(k) &= \sum_{n=0}^{N/2-1} x(2n) W_{N/2}^{nk} \\ G(k) &= \sum_{n=0}^{N/2-1} x(2n+1) W_{N/2}^{nk} \end{aligned}$$

- Step 1: separate even and odd terms from the sum
- Step 2: factorize W_N^k out of the second sum
- Step 3: note that

$$W_N^{2nk} = e^{-i2\pi 2nk/N} = e^{-i2\pi nk/(N/2)} = W_{N/2}^{nk}$$

and define

$$\begin{aligned} F(k) &= \sum_{n=0}^{N/2-1} x(2n) W_{N/2}^{nk} \\ G(k) &= \sum_{n=0}^{N/2-1} x(2n+1) W_{N/2}^{nk} \end{aligned}$$

If we used matrix multiplication to compute F and G we would require

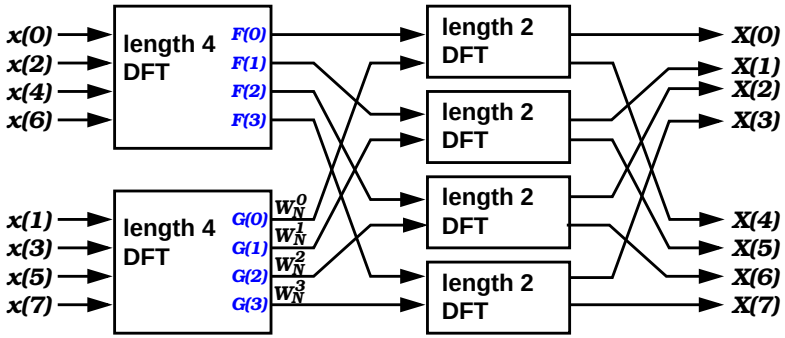
$$2 \times O((N/2)^2) = O(N^2/2)$$

multiplications. Adding them together (with the extra factor) takes fewer operations, so we can ignore these for the moment.

However, note that we can use the approach **recursively**: i.e., we apply the same idea to F and to G.

Radix-2

Procedure: split the data in two (odd terms and even terms) and FFT the two sequences, then add (with an additional factor for odd terms).



Can repeat recursively

Depends on the length of the series being even.

Radix-2 as factorization

Can view the DFT as a matrix transform $X = Ax$, where (for a length 8 sequence)

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & W & W^2 & W^3 & W^4 & W^5 & W^6 & W^7 \\ 1 & W^2 & W^4 & W^6 & W^8 & W^{10} & W^{12} & W^{14} \\ 1 & W^3 & W^6 & W^9 & W^{12} & W^{15} & W^{18} & W^{21} \\ 1 & W^4 & W^8 & W^{12} & W^{16} & W^{20} & W^{24} & W^{28} \\ 1 & W^5 & W^{10} & W^{15} & W^{20} & W^{25} & W^{30} & W^{35} \\ 1 & W^6 & W^{12} & W^{18} & W^{24} & W^{30} & W^{36} & W^{42} \\ 1 & W^7 & W^{14} & W^{21} & W^{28} & W^{35} & W^{42} & W^{49} \end{pmatrix}$$

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.9/77

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.10/77

Radix-2 as factorization

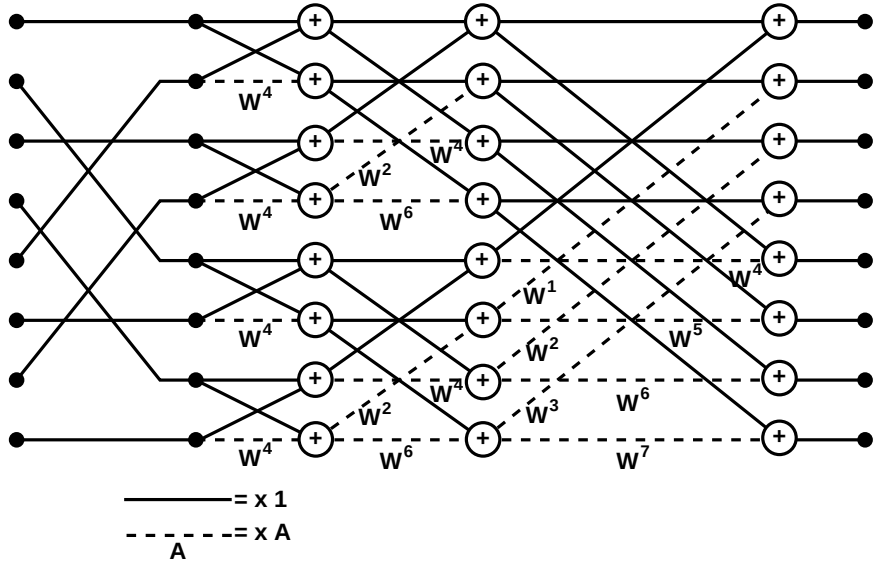
We can factorize A

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & W & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & W^2 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & W^3 \\ 1 & 0 & 0 & 0 & W^4 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & W^5 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & W^6 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & W^7 \end{pmatrix} \times \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & W^2 & 0 & 0 & 0 & 0 \\ 1 & 0 & W^4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & W^6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & W^2 \\ 0 & 0 & 0 & 0 & 1 & 0 & W^4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & W^6 \end{pmatrix}$$

$$\times \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & W^4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & W^4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & W^4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & W^4 \end{pmatrix} \times \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.11/77

Picture of factorization



Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.12/77

Radix-2 bit reversal

Note the funny ordering of the inputs. Need to efficiently order the inputs. The procedure used for radix-2 is called bit reversal.

Normal order	binary bits	bits reversed	new order
0	000	000	0
1	001	100	4
2	010	010	2
3	011	110	6
4	100	001	1
5	101	101	5
6	110	011	3
7	111	111	7

Other algorithms

Radix-2 takes $O(N \log_2 N)$ calculations, if the data length is a power of two.

- use recursive **decimation** by factors of two
- if the data isn't a power of two, we could pad with zeros, but this is adding unwanted calculations.
- better alternatives exist
- radix-N, or some alternative (e.g. see "Transforms and Fast Algorithms for Signal Analysis and Representations", Bi and Zeng, Birkhauser, 2004)

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.13/77

You probably won't ever need to code up the FFT – there is lots of good code available, e.g.,

```
http://home.get2net.dk/jjn/fft.htm
http://momonga.t.u-tokyo.ac.jp/~ooura/fft.html
http://cpan.uwinnipeg.ca/htdocs/Math-FFT/Math/FFT.html
```

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.14/77

Matlab commands for FFT

Note, indexes in Matlab run from 1 to N (not 0 to $N-1$).

$$\text{fft}(x(n)) = X(k) = \sum_{n=1}^{N} x(n) e^{-i2\pi(k-1)(n-1)/N}, \quad k = 1, \ldots, N.$$

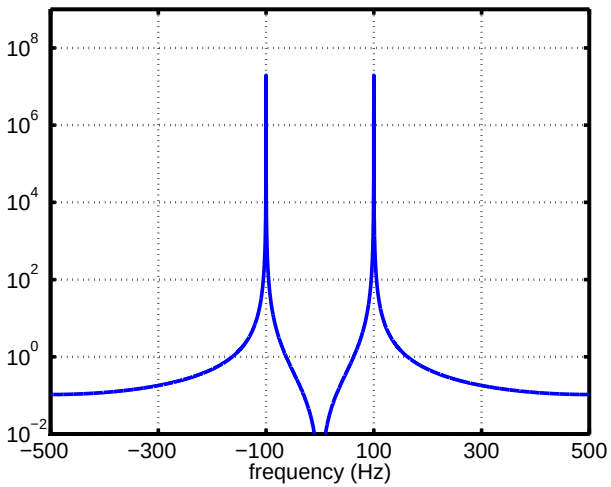
$$\text{ifft}(X(k)) = x(n) = \frac{1}{N} \sum_{k=1}^{N} X(k) e^{i2\pi(k-1)(n-1)/N}, \quad n = 1, \ldots, N.$$

$X(1)$ is the DC term, $X(n)$ is the f_s term. To plot symmetric power spectrum use, e.g.

```
f_s = 1000;
f_0 = 100;
x = 1:1/f_s:10;
y = sin(2*pi*f_0*x);
semilogy(-f_s/2+f_s/N:f_s/N:f_s/2, abs(fftshift(fft(y))).^2);
set(gca, 'ylim', 10.^[-2 9]);
xlabel('frequency (Hz)');
```


Matlab example

matlab_ex_1.m



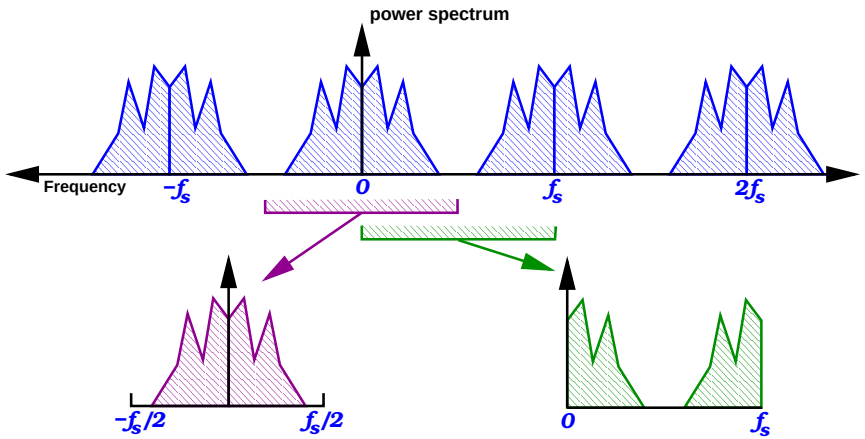
Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.15/77

```
% MATLAB_EX_1 shows a simple example of fft in practice
%
% file:       matlab_ex_1.m, (c) Matthew Roughan, Sat Aug  7 2004
% directory:   /home/mroughan/Classes/Transformations/2004/Matlab/
%
%
f_s = 1000;            % sampling frequency
f_0 = 100;             % frequency of the signal
x = 1:1/f_s:10;        % sample points
N = length(x);
y = sin(2*pi*f_0*x);   % sampled signal
semilogy(-f_s/2+f_s/N:f_s/N:f_s/2, abs(fftshift(fft(y))).^2, 'linewidth', 3);

%%%% make the axes pretty and add labels
grid on
set(gca, 'ylim', 10.^[-2 9], 'ytick',  10.^[-2:2:9], 'xtick', [-500:200:500]);
set(gca, 'linewidth', 3, 'fontsize', 18);
xlabel('frequency (Hz)');

%%%% print out a copy
print('-depsc', 'Plots/matlab_ex_1.eps');
```

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.16/77

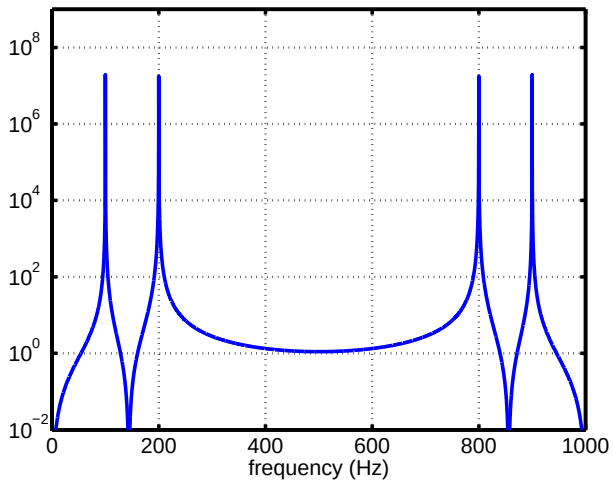
Symmetry

Discrete power spectrum is **even** and **periodic** so we can display in a number of ways.



Matlab example 2

matlab_ex_2.m



Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.17/77

```
% MATLAB_EX_1 shows a simple example of fft in practice
%
% file:       matlab_ex_1.m, (c) Matthew Roughan, Sat Aug  7 2004
% directory:   /home/mroughan/Classes/Transformations/2004/Matlab/
%
%
f_s = 1000;            % sampling frequency
x = 1:1/f_s:10;        % sample points
N = length(x);

%%%%  sampled signal
f_0 = 100;             % frequency 0 in the signal
f_1 = 200;             % frequency 1 in the signal
y = sin(2*pi*f_0*x) + sin(2*pi*f_1*x);

%%%%  FFT of data
z = fft(y);
freq = (0:N-1) * f_s/N;

%%%%  plot the data
semilogy(freq, abs(z).^2, 'linewidth', 3);

%%%% make the axes pretty and add labels
grid on
set(gca, 'ylim', 10.^[-2 9], 'ytick',  10.^[-2:2:9], 'xtick', [0:200:1000]);
set(gca, 'linewidth', 3, 'fontsize', 18);
xlabel('frequency (Hz)');

%%%% print out a copy
print('-depsc', 'Plots/matlab_ex_2.eps');
```

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.18/77

Random Processes

We've mentioned "noise" before, but in a fairly crude sense. No consideration of noise can really be done carefully without some understanding of random processes, and how Fourier analysis works on random signals, in particular we will define the spectral density of a signal.

Random processes

A discrete time random process is just a random vector $\mathbf{x} = (X_1, X_2, \ldots, X_n)$.

- in general, the x_i may have dependencies, so we need to describe the random sequence, we specify the N-th order distribution functions, for all $N \geq 1$

$$F_{\mathbf{x}}(x_n, x_{n+1}, \ldots, x_{n+N-1}) = P\{X_n \leq x_n, X_{n+1} \leq x_{n+1}, \ldots, X_{n+N-1} \leq x_{n+N-1}\}$$

- typically, we don't need to know all of this, e.g. for **White** noise, the values at each time interval are independent, so we only need to know the first order distribution functions $F_{\mathbf{x}}(x_n)$.

- "Probability and Random Processes, with Applications to Signal Processing", 3rd Ed., Henry Stark and John W. Woods, Prentice-Hall, 2002.
- "Introduction to Time Series and Forecasting", Peter J. Brockwell and Richard A. Davis, Springer-Verlag, 1996.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.19/77

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.20/77

Densities and distributions

Where the distribution function is continuous (e.g. for well-behaved continuous random variables), we can define a density function, e.g.

$$f_X(x) = \frac{dF_X}{dx}$$

with the meaning that

$$f_X(x)\,dx = P\{X \in [x, x+dx)\}$$

We can define $f_X(x)\,dx = dF_X(x)$, where the latter term is more general (applying to badly behaved random variables too). Integrals defined WRT to $dF_X(x)$ are Lebesgue-Stieltjes integrals rather than just Lebesgue integrals.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.21/77

Moments of the process

Sometimes it is enough to specify the moments of the process, e.g. the mean $\mu_X(n)$ and variance $\sigma_X^2(n)$ at time n.

$$\begin{aligned}
\mu_X(n) &= E[X(n)] \\
&= \int_{-\infty}^{\infty} x\, dF_X(x_n) \\
&= \int_{-\infty}^{\infty} x f_X(x_n)\, dx, \quad \text{where this is defined} \\
\sigma_X^2(n) &= \mathrm{Var}[X(n)] \\
&= E\left[(X(n) - \mu_X(n))^2\right] \\
&= \int_{-\infty}^{\infty} (x - \mu_X(n))^2\, dF_X(x_n)
\end{aligned}$$

Can extend definition to the n-th central moment.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.22/77

Covariance

The **covariance** of two random variables X and Y if defined by

$$\text{Cov}\{X,Y\} = E\left[(X - E\left[X\right])(Y - E\left[Y\right])\right]$$

It tells us about second-order correlations between X and Y.

The **auto-covariance** of a process is

$$R_{XX}(n;k) = \text{Cov}\{X(n), X(n+k)\}$$

and this tells us about correlations between the process at different times n, and different **lags** k.

Stationarity

- A process is **strictly stationary** if all of its distribution functions are invariant under time shifts, e.g.

$$F_{\mathbf{x}}(x_n, x_{n+1}, \ldots, x_{n+N-1}) = F_{\mathbf{x}}(x_{n+k}, x_{n+k+1}, \ldots, x_{n+k+N-1})$$

- a process is called **wide-sense**, or **weakly**, or **second-order** stationary if its mean, variance and auto-covariance are time-shift invariant, e.g.

$$\begin{aligned} \mu_X(n) &= \mu_X \\ \sigma_X^2(n) &= \sigma_X^2 \\ R_{XX}(n;k) &= R_{XX}(k) \end{aligned}$$

for all n.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.23/77

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.24/77

Marginal distribution

The marginal distribution of a stationary process is defined by the distribution of X_n, e.g.

$$F_{\mathbf{x}}(x_n)$$

which will be identical for all values of n for a stationary process.

Examples:

- Bernoulli process: the marginal distribution takes values 0 or 1 with probabilities p and $1-p$, respectively.
- random dice rolls: the marginal distribution is uniform on $\{1,2,3,4,5,6\}$.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.25/77

Gaussian processes

- are processes with a Gaussian marginal distribution

$$f_{\mathbf{x}}(x_n) = \frac{1}{\sqrt{2\pi\sigma_X^2}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu_X}{\sigma_X}\right)^2\right)$$

- completely characterized by mean, variance and auto-covariance
- hence the value of second-order stationarity!
- Gaussian processes are the "linear-time invariant" processes of the noise world
  - simple, tractable, sometimes reasonable
  - Central Limit Theorem: sums of well behaved random variables tend towards Gaussian distributions

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.26/77

Useful fact

If X and Y are independent Gaussian random variables, then their sum $X+Y$ (and difference $X-Y$) will also be Gaussian, with mean and variances

$$\begin{aligned}\mu_{X+Y} &= \mu_X + \mu_Y \\ \mu_{X-Y} &= \mu_X - \mu_Y \\ \sigma^2_{X+Y} &= \sigma^2_X + \sigma^2_Y \\ \sigma^2_{X-Y} &= \sigma^2_X + \sigma^2_Y\end{aligned}$$

We could easily generalize this to include correlations, with the only effect being a modification to σ^2_{X+Y} and σ^2_{X-Y}.

Also note that $\mu_{\alpha X} = \alpha\mu_X$, and $\sigma^2_{\alpha X} = \alpha^2\sigma^2_X$.

Spectral density

There is a direct relationship between the power spectrum of stochastic processes, and its autocovariance function. Note, the power spectrum of a single random process is also a collection of random variables, so we talk about the mean values of the power spectrum, or rather, its **spectral density**.

$$f_X(\lambda) = \sum_{h=-\infty}^{\infty} e^{-i2\pi h\lambda} R_{XX}(h)$$

Note then

$$R_{XX}(k) = \int_{-1}^{1} e^{i2\pi k\lambda} f(\lambda)\, d\lambda$$

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.27/77

NB: above holds for a mean zero process. If we have a process with non-zero mean, then we need to write:

$$W_{XX}(h) = R_{XX}(h) + \mu_X^2 \tag{1}$$

and

$$f_X(\lambda) = \sum_{h=-\infty}^{\infty} e^{-i2\pi h\lambda} W_{XX}(h)$$

Note also that for a finite sequence, we are forced to use an approximation: e.g.

$$P_X(k) \simeq \frac{1}{2N+1} \sum_{n=-N}^{N} W_{XX}(h) e^{-i2\pi kn/N} \tag{2}$$

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.28/77

Spectral density properties

The spectral density has the properties

- $f(\lambda)$ is non-negative
- $f(\lambda)$ is even
- $f(\lambda)$ is the Fourier transform of the autocovariance.
- the autocovariance is the inverse FT of $f(\lambda)$

Why the relationship?

In real-life, only we usually only get one realization of a process, so we can't measure directly the ensemble behavior, e.g. we can't directly measure $R_{XX}(k)$.

For stationary (ergodic) processes we use the fact that a time average converges to a ensemble average, for instance

$$\frac{1}{N}\sum_{n=0}^{N-k-1}[x(n)-\mu_X][x(n+k)-\mu_X] \overset{N\to\infty}{\to} R_{XX}(k)$$

↑ time average ↑ ensemble average

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.29/77

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.30/77

Wiener-Kintchine theorem

For simplicity, consider the mean zero case, e.g. $\mu_X = 0$

$$\begin{aligned} R_{XX}(k) &= \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-k-1} x(n)x(n+k) \\ &\simeq \frac{1}{N} \sum_{n=k}^{N-1} x(n-k)x(n) \\ &\simeq \frac{1}{N} x(n) * x(-n) \end{aligned}$$

Take the FT, and we get

$$\begin{aligned} \mathcal{F}\{R_{XX}(k)\} &= \frac{1}{N} X(k)X^*(k) \\ &= \frac{1}{N} |X(k)|^2 \end{aligned}$$

White noise

Typically, everyone assumes noise is "white", or "uncolored".

- Gaussian (typically implied, though not necessary)
- its spectral density is flat
  i.e., the noise includes all frequencies (up to f_s)

$$f(\lambda) = \sigma^2$$

- Uncorrelated (same as independent for Gaussian)
  - follows from duality of the spectrum and auto-covariance, i.e. flat spectrum implies delta function (at zero) for auto-covariance, and so the autocovariance is zero at non-zero lags.

Note the abuse of notation, i.e., what is really meant in the convolution (line 4) is

$$x(n) * x(-n) = \left[x * \overleftarrow{x}\right](n)$$

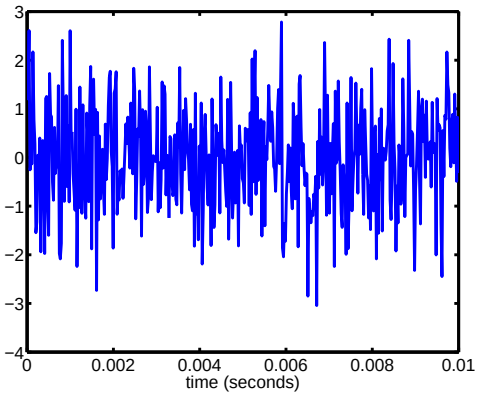
where $\overleftarrow{x}(n) = x(-n)$, i.e. the arrow denotes time reversal (in the cyclical sense). Note also that the convolution is truncated for finite N, so the result is an approximation that converges in the limit as $N \to \infty$.

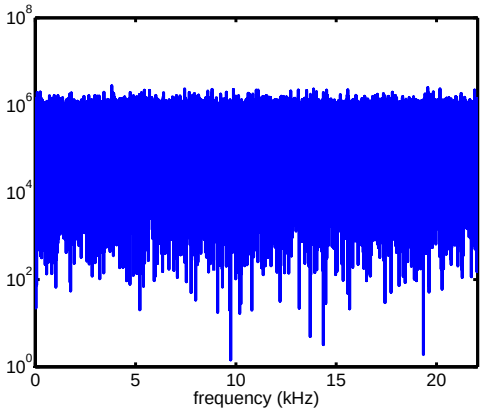
Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.31/77

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.32/77

Example: white noise

White Gaussian noise





Spectral density

The spectral density acts like the power spectrum, and so allows us to work out (statistically) what the output of noise passed through a linear time-invariant filter (or system) will look like.

- we can compute the mean of the output from stationary solution to the recurrence relation described by the filter
- we can compute the auto-covariance from the Wiener-Khintchine theorem
- if the input is Gaussian, then the output is also Gaussian
- hence it is completely characterized by its mean, variance, and auto-covariance.

White noise is what we sometimes call “static”.

```
%
% file:       noise_sounds.m, (c) Matthew Roughan, Sun Aug  1 2004
%

%%% first look at some white noise
fs = 44100;
T = 5;
x = (1:fs*T)/fs;
y = randn(size(x));

figure(10)
plot(x,y, 'linewidth', 3);
set(gca, 'xlim', [0 0.01]);
set(gca, 'linewidth', 3, 'fontsize', 18);
xlabel('time (seconds)');
print('-depsc', 'Plots/noise_white.eps');

z = fft(y);
w = abs(z).^2;
w = fftshift(w);
q = (-length(w)/2+1:length(w)/2)/T;

figure(1)
semilogy(q/1000, w, 'linewidth', 3);
set(gca, 'linewidth', 3);
set(gca, 'fontsize', 18);
set(gca, 'xlim', [0 fs/2000]);
xlabel('frequency (kHz)');
print('-depsc', 'Plots/noise_white_fft.eps');

wavwrite(y, fs, 'Plots/noise_white.wav');
```

Filtered noise

Write the filter in terms of its impulse response, e.g.

$$y(n) = \sum_{i=-\infty}^{\infty} w(i)x(n-i)$$

Expectation is linear operator so for a stationary process $x(n)$

$$\begin{aligned} E[y(n)] &= E\left[\sum_{i=-\infty}^{\infty} w(i)x(n-i)\right] \\ &= \sum_{i=-\infty}^{\infty} w(i)E[x(n-i)] \\ &= \mu_X \sum_{i=-\infty}^{\infty} w(i) \end{aligned}$$

Filtered noise

Then note that if $x(n)$ is white Gaussian noise, the RHS above is the sum of indep. Gaussian random variables, so its variance will add

$$\begin{aligned} \mathrm{Var}[y(n)] &= \mathrm{Var}\left[\sum_{i=-\infty}^{\infty} w(i)x(n-i)\right] \\ &= \sum_{i=-\infty}^{\infty} w(i)^2 \mathrm{Var}[x(n-i)] \\ &= \sigma_X^2 \sum_{i=-\infty}^{\infty} w(i)^2 \end{aligned}$$

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.35/77

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.36/77

Examples

Input process $x(n)$, with mean μ and variance σ^2

- MA filter: $y(n) = \frac{1}{N}\sum_{i=0}^{N-1} x(n-i)$, the output will have mean μ, and variance $\frac{\sigma^2}{N}$.

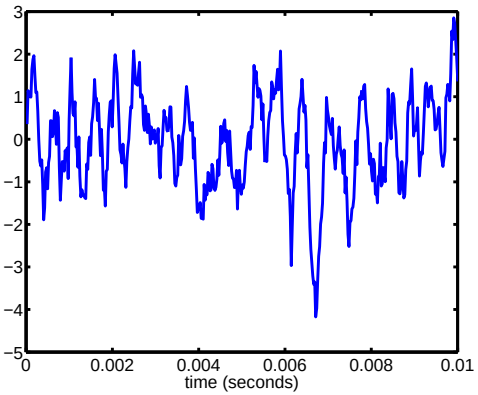
  As we expect, the MA is an unbiased estimator of the mean μ_X, and the variance is reduce by a factor of N for a filter of length N.

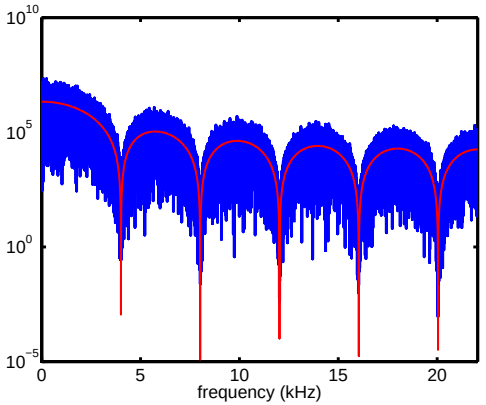
- Difference: $y(n) = x(n) - x(n-1)$, the output will have mean 0, and variance $2\sigma^2$.

  As we expect, the diff has mean zero, but larger variance (it emphasizes changes)

Example 1: filtered white noise

White noise, filtered with rectangular MA, length 11





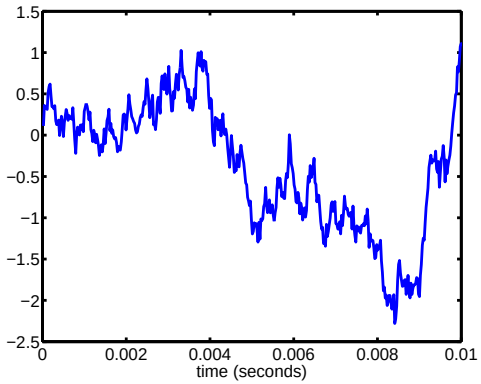
Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.37/77

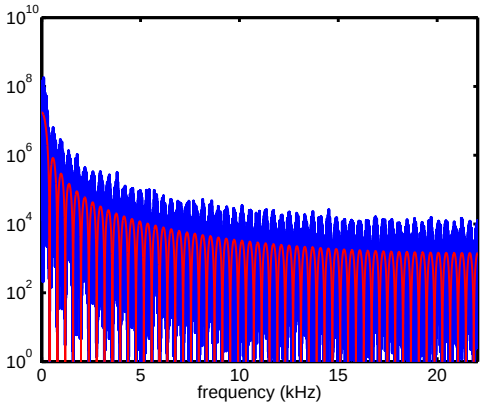
See `noise_sounds.m` again.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.38/77

Example 2: filtered white noise

White noise, filtered with rectangular MA, length 111



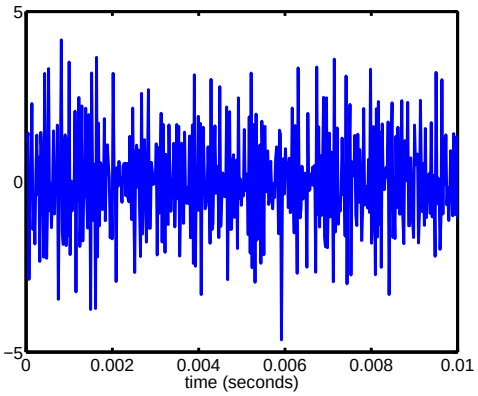


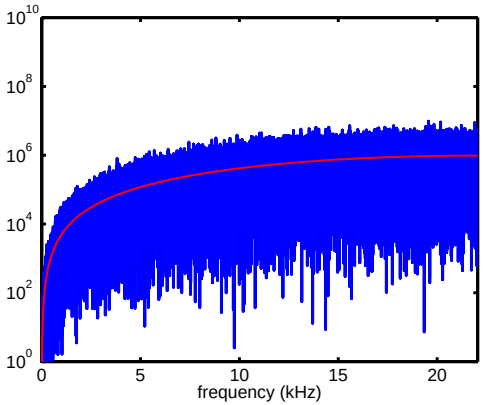
See `noise_sounds.m` again.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.39/77

Example 3: filtered white noise

White noise, filtered with a difference





See `noise_sounds.m` again.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.40/77

Color of noise

- **white:** flat spectral density see above.
- **brown:** spectral density proportional to $1/f^2$ (decrease by 6dB per octave). Named after Brownian motion to which it is related.
- **pink:** spectral density proportional to $1/f$ (decreases by 3dB per octave). This is non-trivial, but we will see more later.
- **blue:** spectral density proportional to f (increases by 3dB per octave). This is meant to be good for some types of dithering.
- plus some more...

Parseval, Rayleigh and Plancheral

There are a key set of theorems in transform theory named after various important figures in this area (Parseval, Rayleigh and Plancheral) and we would be remiss if we did not consider these along with a generalization of the Fourier transform.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.41/77

See
http://groups.google.com/group/comp.dsp/msg/3ae211b7cb0f2a76?q=colors+of+noise&hl=en&lr=&ie=UTF-8&safe=off&rnum=2
http://en.wikipedia.org/wiki/Purple_noise

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.42/77

Parseval's theorem

- $p(x)$ is a real periodic function with period T.
- Fourier **series** a_i, and b_i (see Lecture 2)

Parseval:

$$\frac{1}{T}\int_{-T/2}^{T/2} p(x)^2\,dx = a_0^2 + \frac{1}{2}\sum_{n=1}^{\infty}(a_n^2 + b_n^2)$$

Rayleigh's theorem

- continuous function $f(x)$ with
- Fourier **transform** $F(s)$

Rayleigh:

$$\int_{-\infty}^{\infty} |f(x)|^2\,dx = \int_{-\infty}^{\infty} |F(s)|^2\,ds$$

Power is conserved by the transform!

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.43/77

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.44/77

Rayleigh's theorem proof

$$\begin{aligned}
\int_{-\infty}^{\infty} |f(x)|^2\,dx &= \int_{-\infty}^{\infty} f(x)\,f^*(x)\,dx \\
&= \int_{-\infty}^{\infty} f(x)\,f^*(x)e^{-i2\pi x s'}\,dx \qquad \text{for } s'=0 \\
&= \mathcal{F}_{s'}\{f(x)\,f^*(x)\} \qquad \text{for } s'=0 \\
&= F(s')*F^*(-s') \qquad \text{for } s'=0 \\
&= \int_{-\infty}^{\infty} F(s)F^*(s-s')\,ds \qquad \text{for } s'=0 \\
&= \int_{-\infty}^{\infty} F(s)F^*(s)\,ds \\
&= \int_{-\infty}^{\infty} |F(s)|^2\,ds
\end{aligned}$$

Parseval-Rayleigh theorem

- discrete time function $x(n)$ with
- **Discrete** Fourier **transform** $X(k)$

$$N\sum_{n=0}^{N-1}|x(n)|^2 = \sum_{k=0}^{N-1}|X(k)|^2$$

Note the abuse of notation, i.e., what is really meant in the convolution (line 4) is

$$F(s')*F^*(-s') = \left[F*\overleftarrow{F}^*\right](s')$$

where $\overleftarrow{F}(s) = F(-s)$, i.e. the arrow denotes time reversal.

Parseval-Rayleigh theorem proof

$$\begin{aligned}
\sum_{n=0}^{N-1}|x(n)|^2 &= \sum_{n=0}^{N-1}x(n)x^*(n) \\
&= \sum_{n=0}^{N-1}x(n)x^*(n)e^{-i2\pi nk'/N} \qquad \text{for } k'=0 \\
&= \mathcal{F}_{k'}\{x(n)x^*(n)\} \qquad \text{for } k'=0 \\
&= X(k')*X^*(-k')/N \qquad \text{for } k'=0 \\
&= (1/N)\sum_{k=0}^{N-1}X(k)X^*(k-k') \qquad \text{for } k'=0 \\
&= (1/N)\sum_{k=0}^{N-1}X(k)X^*(k) = (1/N)\sum_{k=0}^{N-1}|X(k)|^2
\end{aligned}$$

Plancherel

- continuous functions $f(x)$ and $h(x)$ with
- Fourier transforms $F(s)$ and $H(s)$

$$\int_{-\infty}^{\infty}f(x)h^*(x)\,dx = \int_{-\infty}^{\infty}F(s)H^*(s)\,ds$$

- real valued $f(x)$ and $h(x)$ then

$$\int_{-\infty}^{\infty}f(x)h(-x)\,dx = \int_{-\infty}^{\infty}F(s)H(s)\,ds$$

- similar results for discrete time

$$N\sum_{n=0}^{N-1}x(n)y^*(n) = \sum_{k=0}^{N-1}X(k)Y^*(k)$$

The DFT of the complex conjugate can be obtained as follows

$$\begin{aligned}
DFT(x^*;k) &= \sum_{n=0}^{N-1}x^*(n)e^{-i2\pi nk/N} \\
&= \left[\sum_{n=0}^{N-1}x(n)e^{i2\pi nk/N}\right]^* \\
&= [IDFT(x;k)]^*
\end{aligned}$$

Remember the DFT duality result (from Lecture 3) $DFT(X;k) = Nx(N-k \bmod N)$ or equivalently

$$DFT(X;N-k \bmod N) = Nx(k)$$

Take inverse DFTs in the above and we get

$$X(N-k \bmod N) = N \times IDFT(x;k)$$

So the DFT of the complex conjugate of x is

$$DFT(x^*;k) = X(-k)^*/N$$

where the time reversal is taken in the cyclic sense (as is the convolution in the proof).

See also:
`http://mathworld.wolfram.com/PlancherelsTheorem.html`

Parseval-Rayleigh-Plancherel

- terminology interchanged in different literature
- e.g. Parseval sometimes used for all three versions
- we are following Bracewell's nomenclature
- important detail:
  power is conserved by the transforms
- extends to 2D

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}|f(x,y)|^2\,dx\,dy = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}|F(u,v)|^2\,du\,dv$$

- this is a common feature of many transforms

Generalized Fourier Transforms

In this section we consider a more general approach to the notion of an integral transform developed from the basics of linear algebra (which we will start by reviewing). The approach allows us to define transforms in terms of a change of basis for representing our signal, and in doing this we can see that there are actually many Fourier-like transforms that we could use, depending on what we wish to accomplish.

http://www-gap.dcs.st-and.ac.uk/~history/Mathematicians/Parseval.html

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.49/77

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.50/77

Linear filter characterization

The FT transforms naturally between two views of a filter

- in time domain by impulse response
- in freq. domain via transfer function

e.g. ARMA filter characterization

- ARMA coefficients
- poles and zeros in the complex plane

The FT (and z-transform) transforms naturally between these two views.

Revue of linear algebra

- vector space
- norms and distances
- inner products
- basis
- eigenvalues, and eigenvectors
- diagonalization

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.51/77

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.52/77

Vector spaces and function spaces

A **Vector Space** S is a non-empty collection of objects (vectors) $X, Y, \ldots$, along with two operators (addition, and scalar multiplication) that is

- closed under addition, e.g.

  For all $X, Y \in S$ we have $X + Y \in S$

- closed under scalar multiplication, e.g.

  For all $X \in S,$ and $k \in \mathbb{R}$ we have $kX \in S$

Vector spaces and function spaces

The operators have to satisfy various properties

commutivity of addition	$X + Y = Y + X$
associativity of addition	$X + (Y + Z) = (Y + X) + Z$
additive identity	$\exists 0$ such that $X + 0 = X$
additive inverse	$\forall X, \exists -X$ such that $X + (-X) = 0$
distributivity	$\alpha(X + Y) = \alpha X + \alpha Y$
distributivity	$(\alpha + \beta)X = \alpha X + \beta X$
associativity of scalar mult.	$(\alpha\beta)X = \alpha(\beta X)$
multiplicative identity	$\exists 1$ such that $1.X = X$

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.53/77

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.54/77

Examples

- **example 1**: the set of vectors $\mathbf{x} \in \mathbb{R}^n$, with the standard vector addition and scalar multiplication.
- **example 2**: the set of all continuous functions on the interval $[x_0, x_1]$, denoted
  $C[x_0,x_1] = \{ f : [x_0,x_1] \to \mathbb{R} \mid f \text{ is continuous } \}$,
  with addition and scalar multiplication defined by

$$(f+g)(x) = f(x)+g(x), \qquad (\alpha f)(x) = \alpha f(x)$$

  for any $\alpha \in \mathbb{R}$, and $f, g \in C[x_0,x_1]$.
- **example 3**: The set of square integrable functions L^2 is the set of functions $f : \mathbb{R} \to \mathbb{R}$ for which $\int_{-\infty}^{\infty} f(x)^2 dx$ exists and is finite, with the same definition of sum and scalar product as for C.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.55/77

Normed spaces

More structure is needed, in particular a way of measuring distances. A **norm** on a vector space S is a real-valued function(al) whose value at $x \in S$ is denoted $\|x\|$, and has the properties

$$\|x\| \geq 0 \tag{3}$$
$$\|x\| = 0 \text{ iff } x = 0 \tag{4}$$
$$\|\alpha x\| = |\alpha| \|x\| \tag{5}$$
$$\|x+y\| \leq \|x\| + \|y\| \text{ (the triangle inequality)} \tag{6}$$

A vector space equipped with a norm is called a **normed vector space**.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.56/77

Examples

- **example 1**: the vector space $\mathbb{R}^n$ can be equipped with the Euclidean norm defined by $\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$. Alternatively we could use the norm defined by $\|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i|$
- **example 2**: the vector space $C[x_0, x_1]$ can be equipped with norms
  $\|f\|_\infty = \sup_{x\in[x_0,x_1]} |f(x)|$
  $\|f\|_1 = \int_{x_0}^{x_1} |f(x)|\,dx$
  $\|f\|_2 = \sqrt{\int_{x_0}^{x_1} f(x)^2\,dx}$
- **example 3**: L^2 can be equipped with norm
  $\|f\|_2 = \sqrt{\int_{-\infty}^{\infty} f(x)^2\,dx}$

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.57/77

Examples (cont.)

- **example 4**: Define $C^n[x_0, x_1]$ to be the set of functions that have at least n continuous derivatives on $[x_0, x_1]$. Note

$$C^n[x_0,x_1] \subset C^{n-1}[x_0,x_1] \subset \cdots \subset C^1[x_0,x_1] \subset C[x_0,x_1]$$

  $C^n[x_0, x_1]$ is a vector space, and $\|f\|_\infty$, $\|f\|_1$, and $\|f\|_2$ are all possible norms on this space. Other norms

$$\|f\|_{\infty,j} = \sum_{k=0}^{j} \sup_{x\in[x_0,x_1]} |f^{(k)}(x)|$$

  for $j \leq n$ on $C^n[x_0, x_1]$.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.58/77

Norms

- denote a normed vector space $(S, \|\cdot\|)$.
- Two norms $\|\cdot\|_a$ and $\|\cdot\|_b$ are said to be equivalent if there exists positive numbers α and β such that for all $x \in S$

$$\alpha\|x\|_a \leq \|x\|_b \leq \beta\|x\|_a$$

- In finite dimensional spaces all norms are equivalent, but not in infinite dimensional spaces.
- Norms define **distances** between elements of space

$$d(f,g) = \|f-g\|$$

Inner products

An **inner product** is a function $\langle\cdot,\cdot\rangle : S \times S \to \mathbb{R}$, i.e. it maps two elements from a vector space S to a real number, such that for any $f,g,h \in S$ and $\alpha \in \mathbb{R}$.

$$\langle f, f\rangle \geq 0 \tag{7}$$
$$\langle f, f\rangle = 0 \text{ iff } f = 0 \tag{8}$$
$$\langle f, h+g\rangle = \langle f, h\rangle + \langle f, g\rangle \tag{9}$$
$$\langle f, g\rangle = \langle g, f\rangle \tag{10}$$
$$\langle \alpha f, g\rangle = \alpha\langle f, g\rangle \tag{11}$$

A vector space with an inner product is called an **inner product space**.
We can use $\sqrt{\langle f, f\rangle}$ as a norm.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.59/77

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.60/77

Examples

- **example 1**: for the set of vectors $\mathbf{x} \in \mathbb{R}^n$, with the standard vector addition and scalar multiplication, we can define the inner product

$$\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{n} x_i y_i$$

- **example 2**: for the set of square integrable functions L^2 is the set of functions $f : \mathbb{R} \to \mathbb{R}$ for which $\int_{-\infty}^{\infty} f(x)^2 \, dx$ exists and is finite, we can define the inner product

$$\langle f, g \rangle = \int_{-\infty}^{\infty} f(x) g(x) \, dx$$

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.61/77

Orthogonality

- **Orthogonal:** two elements of a vector space S are orthogonal if $f \neq g$ implies $\langle f, g \rangle = 0$.
- this is a generalization of the idea of perpendicular vectors
- the key is that f, when **projected** onto g is zero (and visa versa).
- inner product acts like a projection operation
- a set of elements of a vector space are orthogonal iff each pair is orthogonal.
- a set of elements $\{f_i\}$ of a vector space are **orthonormal** if $< f_i, f_j >= \delta_{ij}$ the Kronecker delta.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.62/77

Basis

A basis for a vector space S is a set of elements $f_i \in S$ such that

- the elements $\{f_i\}$ span S, i.e., any element of S can be written as a linear combination of the f_i.
- the elements $\{f_i\}$ are linearly independent, i.e. we cannot write any element f_i as a linear combination of the other elements of the basis.

A basis is useful because

- it provides a way of representing elements of S
  - uniquely
  - efficiently (no redundancy)
  - constructively (Gram-Schmidt)

Generalized Fourier transform

Take a **complete orthonormal** system of functions $\{\phi_i(x)\}$, such that they form a basis for L^2. e.g.

$$\langle \phi_i, \phi_j \rangle = \delta_{ij}$$

for some inner product $\langle \cdot, \cdot \rangle$.

A function $f(x)$ may be represented as

$$f(x) = \sum_{i=0}^{\infty} a_i \phi_i(x)$$

Then we can define the Generalized Fourier Transform by

$$a_i = \langle f, \phi_i \rangle$$

More generally (than span) we want a basis to be "complete" in the sense that for every piecewise continuous function $f(x)$, the minimum square error

$$E_n = \left\| f - \left(\sum_{i=0}^{n} a_i \phi_i \right) \right\|,$$

goes to zero as $n \to \infty$.

For definitions based on the L^2 norm see
`http://mathworld.wolfram.com/OrthogonalFunctions.html`
`http://mathworld.wolfram.com/CompleteOrthogonalSystem.html`

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.63/77

Note that now the transform has been generalized with respect to

- the underlying space on which we operate
- the set of basis functions
- the definition of inner product

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.64/77

Generalized Fourier transform

$$f(x) = \sum_{i=0}^{\infty} a_i \phi_i(x)$$

$$\begin{aligned} \langle f, \phi_j \rangle &= \left\langle \sum_{i=0}^{\infty} a_i \phi_i, \phi_j \right\rangle \\ &= \sum_{i=0}^{\infty} a_i \langle \phi_i, \phi_j \rangle \\ &= \sum_{i=0}^{\infty} a_i \delta_{ij} \\ &= a_j \end{aligned}$$

Generalized Fourier transform

$$a_i = \langle f, \phi_i \rangle$$

$$f(x) = \sum_{i=0}^{\infty} a_i \phi_i(x) = \sum_{i=0}^{\infty} \langle f, \phi_i \rangle \phi_i(x)$$

NB: where i is replaced by a continuous index term s, we might write

$$\begin{aligned} f(x) &= \int a(s) \phi(x; s)\, ds \\ &= \int \langle f, \phi(x; s) \rangle \phi(x; s)\, ds \\ a(s) &= \langle f, \phi(x; s) \rangle \end{aligned}$$

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.65/77

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.66/77

Generalization of P-R

$$\begin{aligned}
\langle f, f \rangle &= \left\langle \sum_{i=0}^{\infty} a_i \phi_i(x), \sum_{j=0}^{\infty} a_j \phi_j(x) \right\rangle \\
&= \sum_{i=0}^{\infty} a_i \left\langle \phi_i(x), \sum_{j=0}^{\infty} a_j \phi_j(x) \right\rangle \\
&= \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} a_i a_j \langle \phi_i(x), \phi_j(x) \rangle \\
&= \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} a_i a_j \delta_{ij} \\
&= \sum_{i=0}^{\infty} a_i^2
\end{aligned}$$

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.67/77

Alternate transforms

- Mostly use L^2 norm to define inner product

$$< f, g >= \int_{-\infty}^{\infty} f(x) g^*(x)\, dx$$

- alternate basis functions are then equivalent to alternative kernels in integrals
- Also $\langle f, f \rangle$ then corresponds to energy in the signal f

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.68/77

Examples of integral transforms

Name	kernel $g(\cdot)$	transform of $f(t)$
Identity	$\delta(s-t)$	$F(s)=\int_{-\infty}^{\infty} f(t)\,\delta(s-t)\,dt$
Fourier	e^{-ist}	$F(s)=\int_{-\infty}^{\infty} f(t)\,e^{-ist}\,dt$
Laplace	e^{-st}, for $t\geq 0$	$F(s)=\int_{0}^{\infty} f(t)\,e^{-st}\,dt$
Hilbert	$\frac{1}{\pi(s-t)}$	$F(s)=\int_{-\infty}^{\infty} f(t)\,\frac{1}{\pi(s-t)}\,dt$
Mellin	t^{z-1}	$F(z)=\int_{0}^{\infty} f(t)\,t^{z-1}\,dt$
Fourier Cosine	$\cos(st)$	$F(s)=\int_{-\infty}^{\infty} f(t)\,\cos(st)\,dt$

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.69/77

Examples of integral transforms

Name	basis functions
Identity	Delta functions $\delta(s-t)$
Fourier	Complex exponentials $e^{-ist}=\cos(st)+i\sin(st)$
Laplace	Real exponentials e^{-st}
Hilbert	Hyperbola $\frac{1}{\pi(s-t)}$
Mellin	Power functions t^{z-1}
Fourier Cosine	Cosines $\cos(st)$

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.70/77

Transforms

Simple transforms are **changes of basis**

- in $\mathbb{R}^n$ we can write these $\mathbf{x} = A\mathbf{y}$
- in more complex spaces, the transform can be represented by an operator, e.g. $\mathcal{F}(f) = g$.

Why would one change basis?

Diagonalization!

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.71/77

Eigenvalues and Eigenvectors

Take a square $n \times n$ matrix A, then a non-zero vector in $\mathbf{x} \in \mathbb{R}^n$ is called an **eigenvector** if it satisfies

$$A\mathbf{x} = \lambda\mathbf{x}$$

for some scalar λ, which is called an **eigenvalue** of A.

$\mathbf{x}$ is said to be the **eigenvector** corresponding to λ.

Similarly for any vector space S and operator $\mathcal{A} : S \to S$, we can define eigenvalues and eigenvectors, such that they satisfy

$$\mathcal{A}f = \lambda f$$

where $f \in S$ is non-zero, and $\lambda \in \mathbb{R}$.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.72/77

Diagonalization

Definition: A matrix is **diagonalizable** if there exists an invertible matrix P such that

$$P^{-1}AP = D$$

where D is a diagonal matrix, e.g.

$$D = \begin{bmatrix} d_1 & & & 0 \\ & d_2 & & \\ & & \ddots & \\ 0 & & & d_n \end{bmatrix}$$

We say that P **diagonalizes** A.

Notation: $D = \text{diag}(d_1, d_2, \ldots, d_n)$.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.73/77

Diagonalization

Given an $n \times n$ matrix A with n linearly independent eigenvectors $\{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$, then A will be diagonalizable, and P has as columns the eigenvectors. The diagonal matrix D will then have the corresponding eigenvalues along its diagonal. That is

$$P^{-1}AP = D$$

where $A\mathbf{v}_k = \lambda_k \mathbf{v}_k$ and

$$P = \left[\begin{array}{c|c|c|c} \vdots & \vdots & & \vdots \\ \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \\ \vdots & \vdots & & \vdots \end{array}\right] \text{ and } D = \begin{bmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{bmatrix}$$

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.74/77

Diagonalization

Diagonalization is useful for studying dynamical systems

- it decouples the elements
- makes it possible to study a bunch of independent simple (1D) systems, rather than one complex, high-dimensional system.
- long term behavior of system comes down to a single dominant eigenvalue/eigenvector.

In a continuous state space, the same is true, though now the basis set is uncountably infinite.

FT as diagonalization

Consider a linear-time invariant system (or filter) as an operator on the vector $\mathbf{x}$

- Impulse response defines a linear operator $\mathcal{A}f = a * f$, where a is the impulse response.
- Complex exponentials are eigenvectors of $\mathcal{A}$, e.g. $\mathcal{A}e^{i2\pi st} = A(s)e^{i2\pi st}$,
  - $A(s)$ is the transfer function.
- The Fourier transform is a diagonalization operation.
- The diagonalization is performed using an inner product with the eigenvectors (which form an orthonormal basis)

Dynamic systems: e.g. write the state of the system at time k as $\mathbf{x}_k$, and assume the system is governed by equations

$$\mathbf{x}_{k+1} = A\mathbf{x}_k$$

We can use induction to show that

$$\mathbf{x}_k = A^k\mathbf{x}_0$$

where $\mathbf{x}_0$ is the initial state. Now

$$A^k = \left(PDP^{-1}\right)^k = PD^kP^{-1}$$

and we can easily show that

$$D^k = \mathrm{diag}(\lambda_1^k, \lambda_2^k, \ldots, \lambda_n^k)$$

so we now have a simple method for computing the state at time k. Also, the eigenvalues tell us something about the long-term stability of the system as for $|\lambda_i| < 1$ the term λ_i^k will go to zero, and if $|\lambda_i| > 1$ the term λ_i^k will go to infinity.

Transform Methods & Signal Processing (APP MTH 4043): lecture 07 – p.76/77

What if?

What happens if we aren't interested in linear, time-invariant systems?

- many real systems are non-linear
- many real systems have transients (i.e. they are not time-invariant)
- Is the Fourier transform still the right approach?