Analytical Geometry and Linear Algebra II, Lab 8

Fourier Series

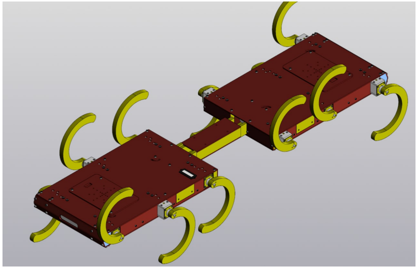
Fast Fourier Transform (FFT)

Discrete Fourier Transform (DFT)

Innopolis University

How I spent last weekend

(a) Minecraft RAGE club meeting



(b) StriRus robot

(c) Museum of Modern Art

Gilbert Strang's Goal VS My Goal

Gilbert Strang's Goal

Is to give you a knowledge how to calculate Discrete Fourier Transform (DFT) by hand. It's an application for using Complex Numbers and Matrices.

My Goal

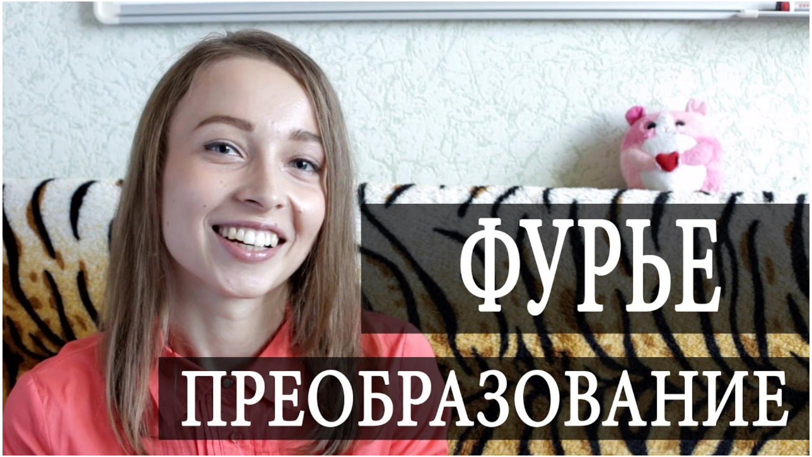
Is to give you the application and the concept why do we need it. It won't be on the exam.

Outline

1. Fourier Series, intuition
2. From Fourier Series to DFT
3. Fast Fourier Transform algorithm

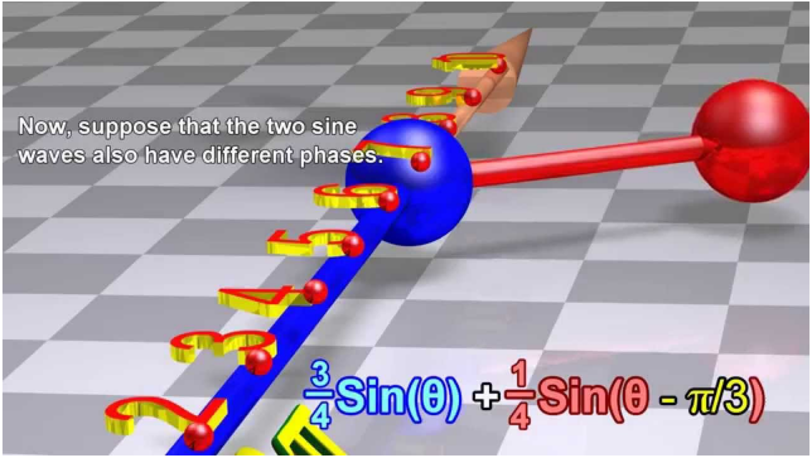
How to imagine Fourier Transform

Video (rus)



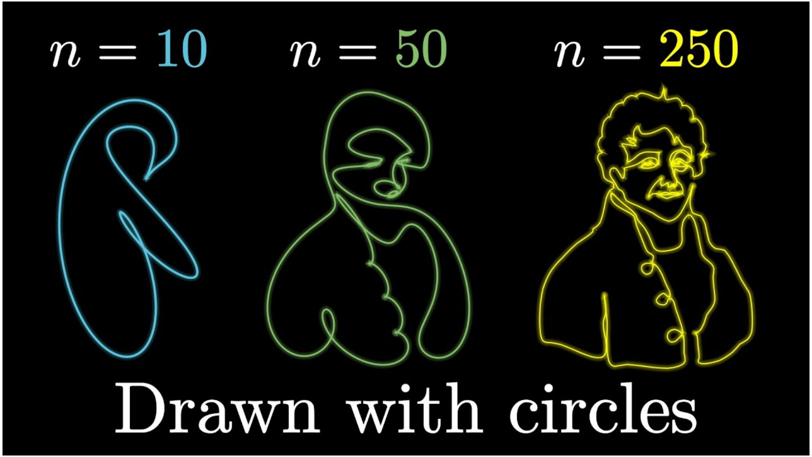
How to sum up sines (Spectrum)

Video



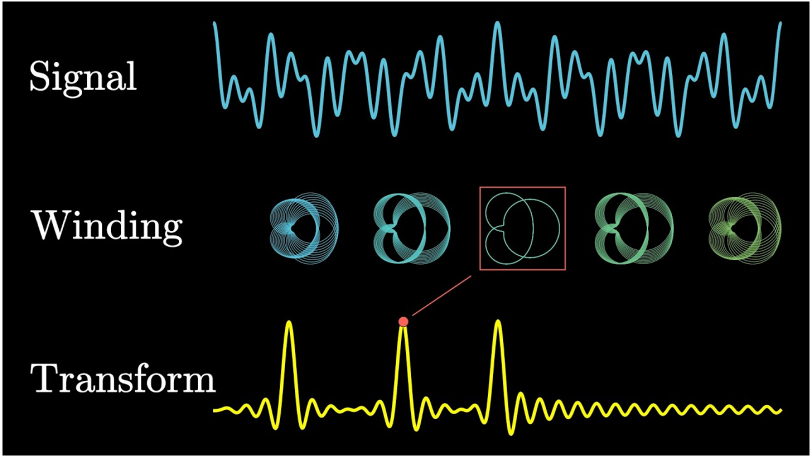
Draw pictures using Fourier Transform

Video



DFT: explanation using sound domain (watch at home)

Video



From Fourier Series to DFT

Fourier Series (Infinite) 1

$$\underbrace{f(x) = \sum_{n=0}^{\infty} a_n \cos(nx) + \sum_{n=1}^{\infty} b_n \sin(nx)}_{\text{Sine-cosine form}} = \underbrace{\frac{a_0}{2} + \sum_{n=1}^{\infty} a_n \cos(nx - \varphi)}_{\text{Amplitude-phase form}}$$

variables

$$\underbrace{\sum_{n=-\infty}^{\infty} c_n e^{inx}}_{\text{Exponential form}}$$

Because we are working in circle, values range is [0,1]

Let's make it finite. Most of real applications require discrete form 2

$$\sum_{n=0}^{N-1} c_n e^{inx} = c_0 + c_1 e^{ix} + c_2 e^{2ix} \dots c_{N-1} e^{(N-1)x_i} = y$$

It's [0,2pi], but we can use any via transforming [a,b] -> [0,2pi]

There are two ways of constructing new points based on the range of known points:

Interpolation (result should pass through all known points (amount of known points(equations) equal to variables), 3

$$F\begin{bmatrix} c_0 \\ c_1 \\ \vdots \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \end{bmatrix}$$

Interpolation example: line

$$\underbrace{\begin{bmatrix} x_1 & 1 \\ x_2 & 1 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} k \\ b \end{bmatrix}}_{c} = \underbrace{\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}}_{y}$$

Regression (result may pass through point (Least Squares) (less variables, than known points)

Usually Fourier is used as interpolation function

Let's write it using DF function

$$F = \begin{bmatrix} 1 & e^{ix_0} & e^{2ix_0} & \dots \\ 1 & e^{ix_1} & e^{2ix_1} & \dots \\ \vdots & & & \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \dots \\ 1 & \omega^2 & \omega^4 & \\ 1 & \vdots & & \\ 1 & & & \end{bmatrix}$$

Trigonometric funcs are periodic, hence $\rightarrow x = \frac{2\pi}{N} \rightarrow \omega = e^{i\frac{2\pi}{N}}$ $\quad F_{ij} = \omega^{ij}$

Lab 8: Discrete Fourier Transform

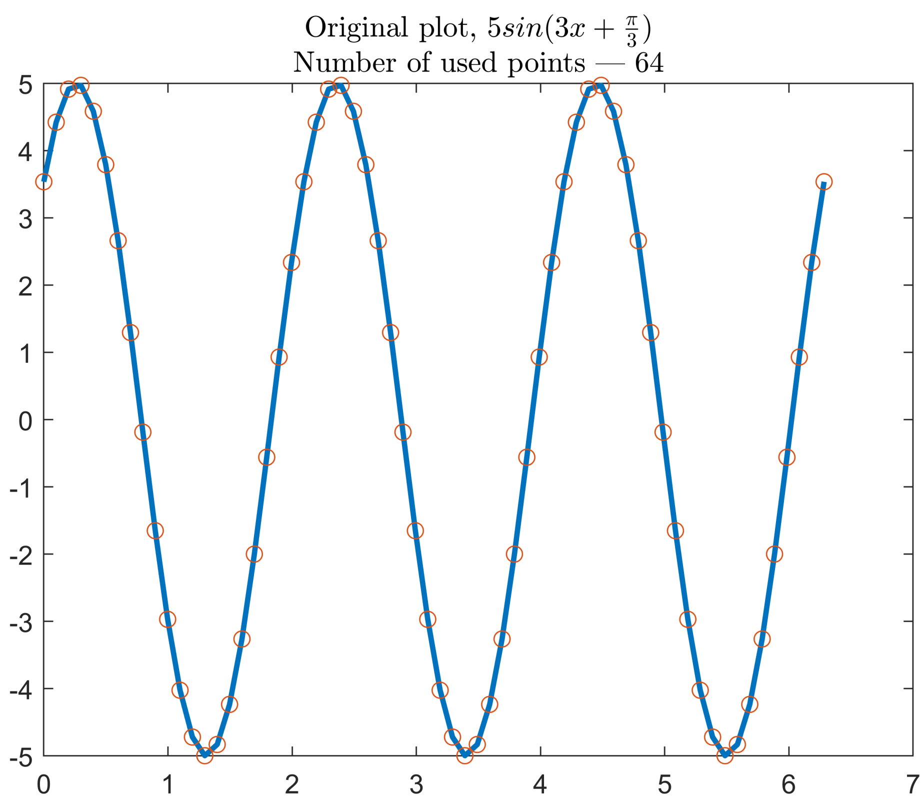
Simple periodic function

Let's choose the function: $5sin(3x+\frac{\pi}{3})$

```
n = 2^6
```

```
n = 64
```

```
x = linspace(0,2*pi,n);
y = 5*sin(3*x+pi/4);
plot(x,y,"LineWidth",2)
title(["Original plot, $5sin(3x+\frac{\pi}{3})$","Number of used points --- " + num2str(n)],"In
hold on
scatter(x,y)
hold off
```



Interpolation, using Fourier Transform

```
% Let's make F_4 for checking that it is what we need

F_4 = conj(dftmtx(4)) % because they use different formula inside
```

```
F_4 = 4×4 complex
   1.0000 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i
```

```
   1.0000 + 0.0000i   0.0000 + 1.0000i  -1.0000 + 0.0000i   0.0000 - 1.0000i
   1.0000 + 0.0000i  -1.0000 + 0.0000i   1.0000 + 0.0000i  -1.0000 + 0.0000i
   1.0000 + 0.0000i   0.0000 - 1.0000i  -1.0000 + 0.0000i   0.0000 + 1.0000i
```

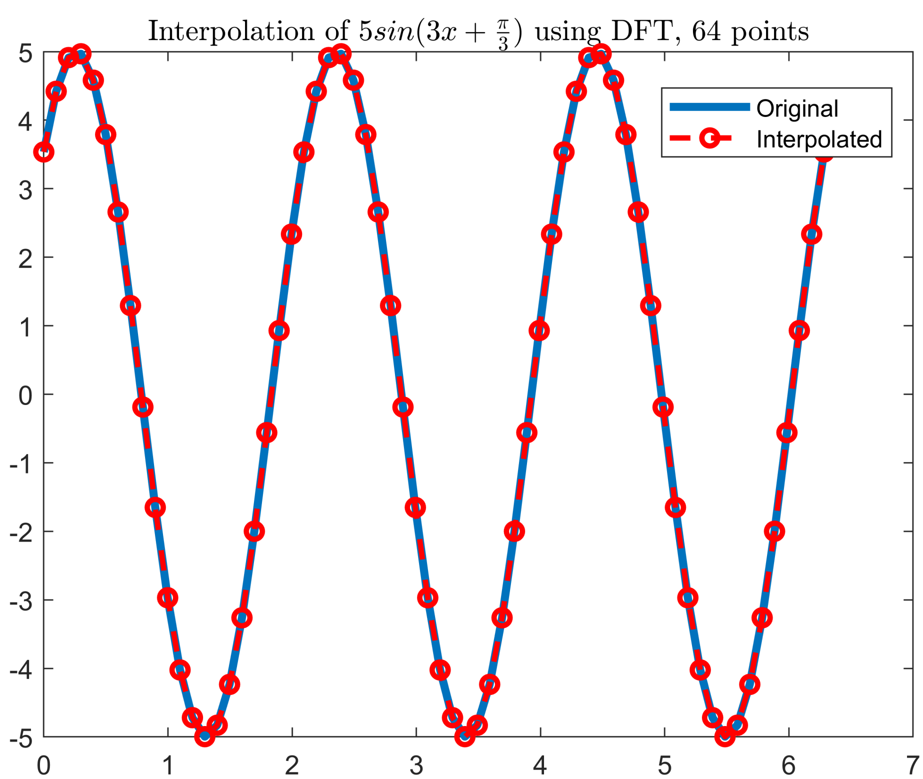
```
F_64 = conj(dftmtx(n));

% Let's find our args of functions
C = linsolve(F_64,y');

Y_new = zeros(n,1);
for t=1:n
    temp = 0;
    for j=1:n
        temp = C(j)*exp(2*pi/n*(t-1)*(j-1)*1i);
        Y_new(t) = Y_new(t) + temp;
    end
end
error_interpolation = rms(y'-Y_new)
```

```
error_interpolation = 4.2429e-14
```

```
p=plot(x,y,x,real(Y_new),'r--o');
p(1).LineWidth = 3;
p(2).LineWidth = 2;
title("Interpolation of $5sin(3x+\frac{\pi}{3})$ using DFT, " + num2str(n) + " points",'interpr
legend("Original","Interpolated")
```

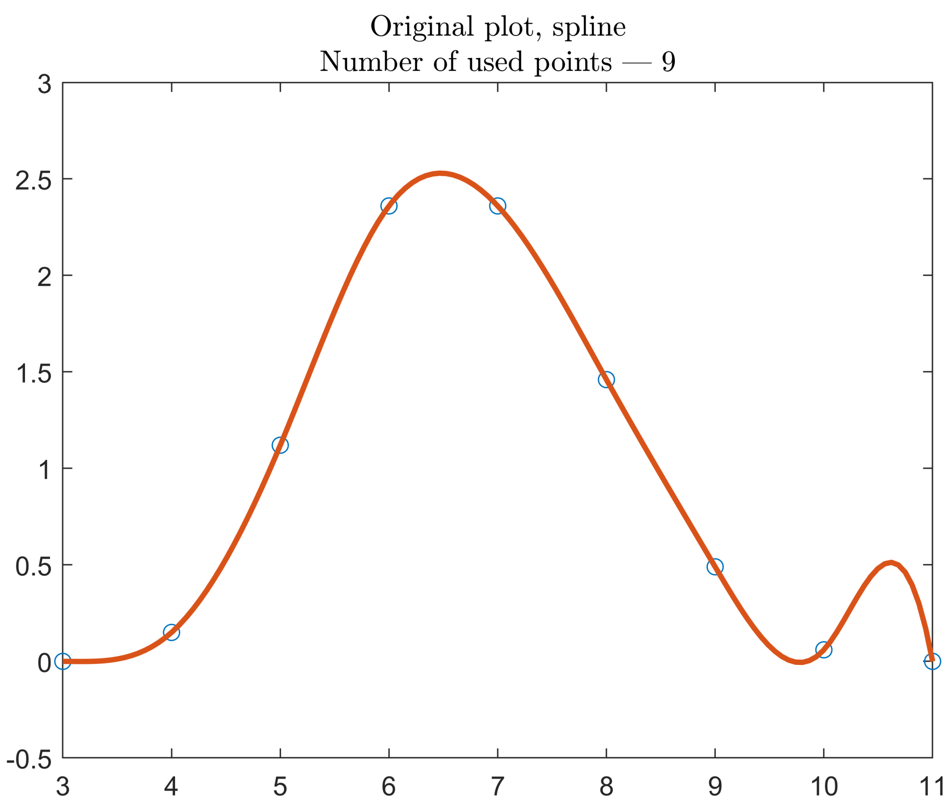


Interpolation, complex example

```
% boundaries
a = 3;
b = 11;
x_exp2 = a:b;
y_exp2 = [0 .15 1.12 2.36 2.36 1.46 .49 .06 0];
cs = spline(x_exp2,[0 y_exp2 -3]);
n_spline = 128
```

```
n_spline = 128
```

```
xx_spline = linspace(a,b,n_spline);
yy_spline = ppval(cs,xx_spline);
p_spline = plot(x_exp2,y_exp2,'o',xx_spline,yy_spline,'-');
p_spline(2).LineWidth=2;
title(["Original plot, spline","Number of used points --- 9"],"Interpreter","latex")
```




```
% Transform from a-b to 0-2pi
mapping = linsolve([0 1;2*pi 1],[a;b])
```

```
mapping = 2×1
    1.2732
    3.0000
```

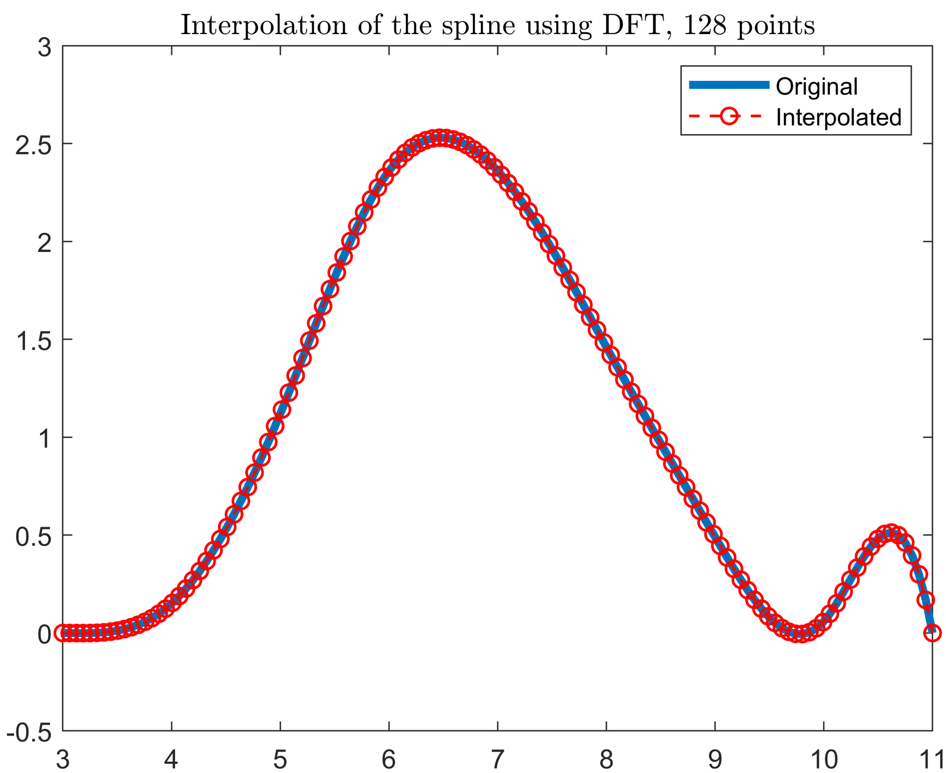
```
F_128 = conj(dftmtx(n_spline));

% Let's find our args of functions
C_spline = linsolve(F_128,yy_spline');

Y_new_spline = zeros(n_spline,1);
for t=1:n_spline
    temp = 0;
    for j=1:n_spline
        temp = C_spline(j)*exp(2*pi/n_spline*(t-1)*(j-1)*1i);
        Y_new_spline(t) = Y_new_spline(t) + temp;
    end
end
error_interpolation_spline = rms(y'-Y_new)
```

```
error_interpolation_spline = 4.2429e-14
```

```
p=plot(xx_spline,yy_spline,xx_spline,real(Y_new_spline),'r--o');
p(1).LineWidth = 3;
p(2).LineWidth = 1;
title("Interpolation of the spline using DFT, " + num2str(n_spline) + " points",'interpreter',
legend("Original","Interpolated")
```



From Fourier Series to DFT

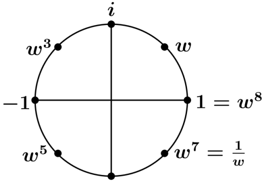
Properties

Fourier Matrix $F_4 = \begin{bmatrix} 1 & \mathbf{1} & 1 & 1 \\ 1 & \boldsymbol{i} & i^2 & i^3 \\ 1 & \boldsymbol{i^2} & i^4 & i^6 \\ 1 & \boldsymbol{i^3} & i^6 & i^9 \end{bmatrix}$ **DFT Matrix** $\overline{F_4} =$ **powers of** $-i$

F_N and $\overline{F_N}$
N by N **matrices**

Replace $i = e^{2\pi i/4}$
by $w = e^{2\pi i/N}$

$\boxed{\begin{array}{l} F_{jk} = w^{jk} = e^{2\pi ijk/N} \\ \text{Columns } k = 0 \text{ to } N-1 \end{array}}$



$w = e^{2\pi i/8} \qquad w^8 = 1$

$1 + w + w^2 + \cdots + w^{N-1} = 0$

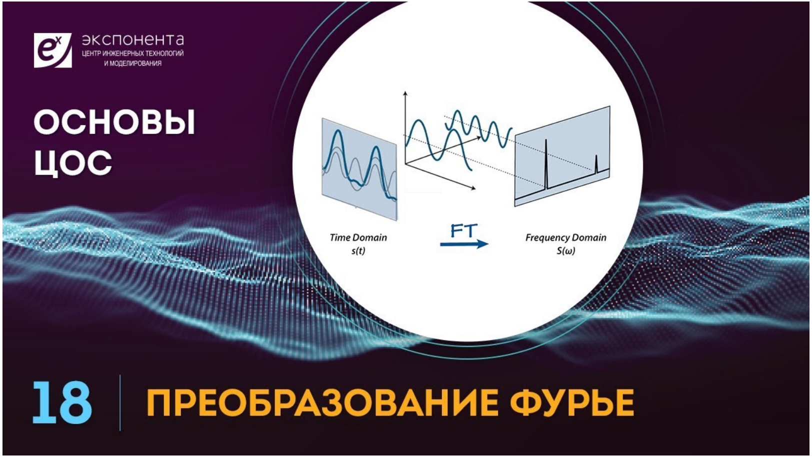
$\boxed{\overline{F_N} F_N = NI}$ Then $F_N/\sqrt{N}$ is a unitary matrix. It has orthonormal columns

$N = 2$
$w = e^{\pi i} = -1$

$F_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad \overline{F}_2 F_2 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} = NI$

DFT application: Sound (rus)

Video

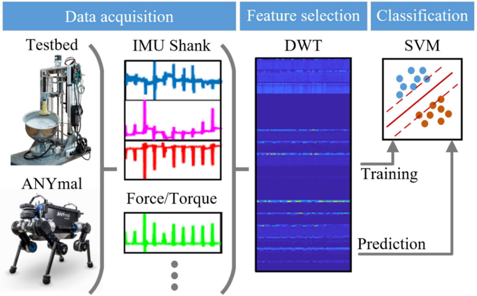


DFT application: Terrain Classification

Feature Extraction step

Problem

How to put such data (2) in ML algorithm (for instance SVM)?



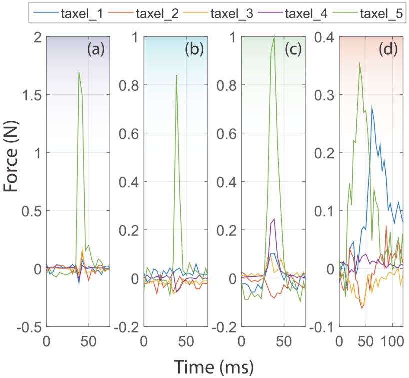


Figure 2: Individual taxel forces recorded on different surfaces at 10 Hz stride frequency

Fast Fourier Transform

Problem Statement

Direct matrix multiplication of **c** by F_N needs $\mathbf{N^2}$ multiplications.

FFT factorization – $\mathbf{\frac{1}{2}N\log_2 N}$ multiplications.

Benefit: $N = 2^{10} = 1024$, $N^2 = 1$ million, FFT – 5000

Constraint of FFT: N should be equal to 2^n

Fast Fourier Transform

Algorithm

Step 1: From 1024 to 512

$$\begin{bmatrix} F_{1024} \end{bmatrix} = \begin{bmatrix} I & D \\ I & -D \end{bmatrix} \begin{bmatrix} F_{512} & 0 \\ 0 & F_{512} \end{bmatrix} \begin{bmatrix} P \end{bmatrix},$$

where D is a diagonal matrix of F_{1024}, but we took only half of it ($512x512$);

P – permutation matrix: for P_{1024} puts columns 0,2,…,1022 ahead of 1,3,…1023.

$$Example\text{:}\ P_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Step 2: From 512 to 256

Step 3…: From 256 to 128 … **Recursion** continues to small N: **log$_2$N** steps.

Task 1

All entries in the factorization of F_6 involve powers of $w =$ sixth root of 1:

$$F_6 = \begin{bmatrix} I & D \\ I & -D \end{bmatrix} \begin{bmatrix} F_3 & \\ & F_3 \end{bmatrix} \begin{bmatrix} P \end{bmatrix}.$$

Write these factors with $1, w, w^2$ in D and $1, w^2, w^4$ in F_3. Multiply!

Task 1

All entries in the factorization of F_6 involve powers of w = sixth root of 1:

$$F_6 = \begin{bmatrix} I & D \\ I & -D \end{bmatrix} \begin{bmatrix} F_3 & \\ & F_3 \end{bmatrix} \begin{bmatrix} P \end{bmatrix}.$$

Write these factors with $1, w, w^2$ in D and $1, w^2, w^4$ in F_3. Multiply!

Answer $D_{3,3} = e^{4\pi i/3}$ is also correct. It depend of w equation (with minus or not).

$$D = \begin{bmatrix} 1 & & \\ & e^{2\pi i/6} & \\ & & e^{4\pi i/6} \end{bmatrix} \text{ and } F_3 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & e^{2\pi i/3} & e^{4\pi i/3} \\ 1 & e^{4\pi i/3} & e^{2\pi i/3} \end{bmatrix}.$$

Task 2

$$Fc = y \qquad \begin{array}{ccccccccc} c_0 & + & c_1 & + & c_2 & + & c_3 & = & 2 \\ c_0 & + & ic_1 & + & i^2c_2 & + & i^3c_3 & = & 4 \\ c_0 & + & i^2c_1 & + & i^4c_2 & + & i^6c_3 & = & 6 \\ c_0 & + & i^3c_1 & + & i^6c_2 & + & i^9c_3 & = & 8. \end{array}$$

Solve the 4 by 4 system if the right-hand sides are $y_0 = 2$, $y_1 = 0$, $y_2 = 2$, $y_3 = 0$. In other words, solve $F_4c = y$.

Task 2

$$
Fc = y \qquad
\begin{array}{ccccccccc}
c_0 & + & c_1 & + & c_2 & + & c_3 & = & 2 \\
c_0 & + & ic_1 & + & i^2c_2 & + & i^3c_3 & = & 4 \\
c_0 & + & i^2c_1 & + & i^4c_2 & + & i^6c_3 & = & 6 \\
c_0 & + & i^3c_1 & + & i^6c_2 & + & i^9c_3 & = & 8.
\end{array}
$$

Solve the 4 by 4 system if the right-hand sides are $y_0 = 2$, $y_1 = 0$, $y_2 = 2$, $y_3 = 0$. In other words, solve $F_4c = y$.

Answer

$$c = (1, 0, 1, 0).$$

Task 3

Find all solutions to the equation $e^{ix} = -1$, and all solutions to $e^{i\theta} = i$.

Task 3

Find all solutions to the equation $e^{ix} = -1$, and all solutions to $e^{i\theta} = i$.

Answer

$e^{ix} = -1$ for $x = (2k+1)\pi$, $e^{i\theta} = i$ for $\theta = 2k\pi + \pi/2$, k is integer.

Task 4

What are F^2 and F^4 for the 4 by 4 Fourier matrix F?

Task 4

What are F^2 and F^4 for the 4 by 4 Fourier matrix F?

Answer

$$F^2 = \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 \\ 0 & 0 & 4 & 0 \\ 0 & 4 & 0 & 0 \end{bmatrix}, \quad F^4 = \begin{bmatrix} 16 & 0 & 0 & 0 \\ 0 & 16 & 0 & 0 \\ 0 & 0 & 16 & 0 \\ 0 & 0 & 0 & 16 \end{bmatrix} = 4^2 I.$$

Reference material

- Fourier Series
- Lecture 26, 2nd part
- *”Linear Algebra and Applications”, pdf pages 221–234*
  Fast Fourier Transform
- *”Introduction to Linear Algebra”, pdf pages 456–462*
  Fast Fourier Transform
- *”Introduction to Linear Algebra”, pdf pages 501–506*
  Fourier Series: Linear Algebra for Functions

Deserve “A” grade!

– Oleg Bulichev

o.bulichev@innopolis.ru

@Lupasic

Room 105 (Underground robotics lab)