LAPORAN PRAKTIKUM
PENGOLAHAN SINYAL WAKTU DISKRIT

MODUL II : *TRANSFORMASI Z* DAN *TRANSFORMASI FOURIER*

DISUSUN OLEH :
Muhammad Naufal Ammar
(17101088)

Tanggal Praktikum : 26 November 2019
Asisten Praktikum :
Angga Pambudi (15101039)
Prasetyo Cahyo N (16101108)

Dosen Praktikum : Khoirun Ni'amah, S.T., M.T

LABORATORIUM MULTIMEDIA
FAKULTAS TEKNIK TELEKOMUNIKASI DAN ELEKTRO (FTTE)
INSTITUT TEKNOLOGI TELKOM
JL. D.I. PANJAITAN 128 PURWOKERTO

2019

BAB I
DASAR TEORI

A. Transformasi Z

Transformasi Z memainkan peran yang sama dalam analisis sinyal waktu *diskret* dan sistem LTI (*Invarian* Waktu *Linear*) sebagai transformasi *Laplace* dalam analisis waktu kontinu dan sistem LTI. Sebagai contoh, di dalam domain-Z (bidang–Z *kompleks*) *konvolusi* dua sinyal domain waktu *ekivalen* dengan perkalian transformasi-Z yang berhubungan. Dalam sistem *diskrit* bentuk umum dari transformasi *Fourier* adalah transformasi-Z. Transformasi-Z sangat berguna dalam menyelesaikan persamaan beda. Transformasi Z diperlukan terkait dengan mencari kemudahan dalam analisis sebuah sistem. Pada dasarnya transformasi Z adalah bentuk umum dari DTDF, transformasi ini memungkinkan dalam melakukan analisis sistem berdasarkan *Pole* dan *Zero*.

Pole : harga-harga z = pi yang menyebabkan X(z) = ∞

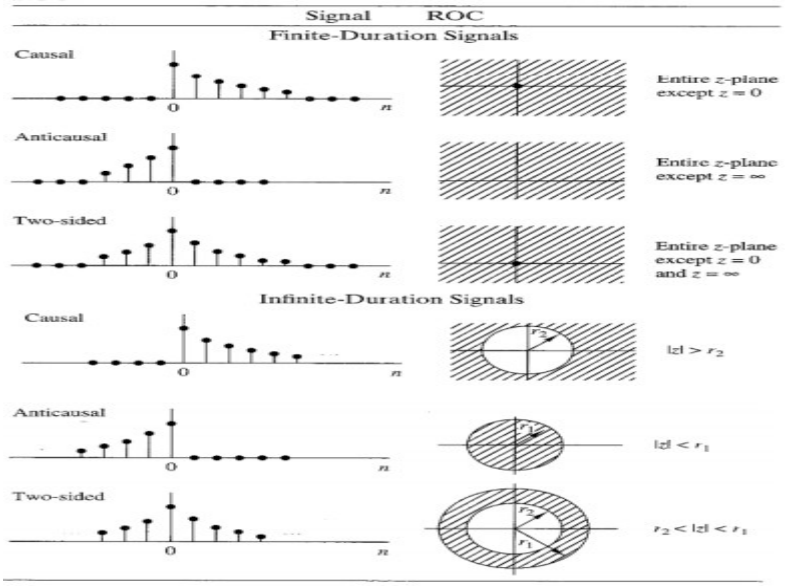
Zero : harga-harga z = zi yang menyebabkan X(z) = 0

Tabel 1.1 Sifat-Sifat Transformasi Z

Property	Sequence	z-Transform	Region of Convergence
Linearity	$ax(n) + by(n)$	$aX(z) + bY(z)$	Contains $R_x \cap R_y$
Shift	$x(n - n_0)$	$z^{-n_0}X(z)$	R_x
Time reversal	$x(-n)$	$X(z^{-1})$	$1/R_x$
Exponentiation	$\alpha^n x(n)$	$X(\alpha^{-1}z)$	$\lvert\alpha\rvert R_x$
Convolution	$x(n) * y(n)$	$X(z)Y(z)$	Contains $R_x \cap R_y$
Conjugation	$x^*(n)$	$X^*(z^*)$	R_x
Derivative	$nx(n)$	$-z\frac{dX(z)}{dz}$	R_x

Karena transformasi Z adalah deret pangkat tak berhingga, transformasi ini hanya berlaku untuk nilai-nilai yang deretnya konvergen. Daerah konvergensi *Range of Convergence* (ROC) X(z) adalah himpunan seluruh nilai z agar X(z) mencapai nilai berhingga. Jadi setiap waktu kita menyebutkan transformasi z kita menunjukan ROC-nya.

Tabel 1.2 Karakteristik sinyal dengan respon dari ROC-nya.



Tabel 1.3 Pasangan dari transformasi z

Sequence	z-Transform	Region of Convergence
$\delta(n)$	1	all z
$\alpha^n u(n)$	$\frac{1}{1-\alpha z^{-1}}$	$\lvert z\rvert > \lvert\alpha\rvert$
$-\alpha^n u(-n-1)$	$\frac{1}{1-\alpha z^{-1}}$	$\lvert z\rvert < \lvert\alpha\rvert$
$n\alpha^n u(n)$	$\frac{\alpha z^{-1}}{(1-\alpha z^{-1})^2}$	$\lvert z\rvert > \lvert\alpha\rvert$
$-n\alpha^n u(-n-1)$	$\frac{\alpha z^{-1}}{(1-\alpha z^{-1})^2}$	$\lvert z\rvert < \lvert\alpha\rvert$
$\cos(n\omega_0)u(n)$	$\frac{1-(\cos\omega_0)z^{-1}}{1-2(\cos\omega_0)z^{-1}+z^{-2}}$	$\lvert z\rvert > 1$
$\sin(n\omega_0)u(n)$	$\frac{(\sin\omega_0)z^{-1}}{1-2(\cos\omega_0)z^{-1}+z^{-2}}$	$\lvert z\rvert > 1$

B. *Transformasi Fourier*

Transformasi Fourier merupakan suatu proses yang banyak digunakan untuk memindahkan domain dari suatu fungsi atau obyek ke dalam domain frekuensi. Di dalam pengolahan citra digital, transformasi *fourier* digunakan untuk mengubah domain spasial pada citra menjadi domain *frekuensi*. Analisa-analisa dalam domain *frekuensi* banyak digunakan seperti *filtering*. Dengan menggunakan transformasi *fourier*, sinyal atau citra dapat dilihat sebagai suatu obyek dalam domain frekuensi.

Transformasi *Fourier* memiliki kelebihan :

a. Mampu menunjukkan kandungan frekuensi yang terkandung di dalam sinyal.
b. Mampu menunjukan beberapa banyak komponen frekuensi yang ada di dalam sinyal.

Kekurangan Transformasi *Fourier* :

a. Transformasi *Fourier* hanya dapat menangkap informasi apakah suatu sinyal memiliki frekuensi tertentu atau tidak, tapi tidak dapat menangkap dimana frekuensi itu terjadi.

1. *Discrete Fourier Transform* (DFT)

   Transformasi *fourier* diskrit atau disebut dengan *Discrete Fourier Transform* (DFT) adalah model transformasi *fourier* yang dikenakan pada fungsi *diskrit*, dan hasilnya juga *diskrit*. DFT didefinisikan dengan :

$$F(k)=\sum_{n=1}^{N} f(n).e^{-j2\pi knT/N}$$

2. *Fast Fourier Transform* (FFT)

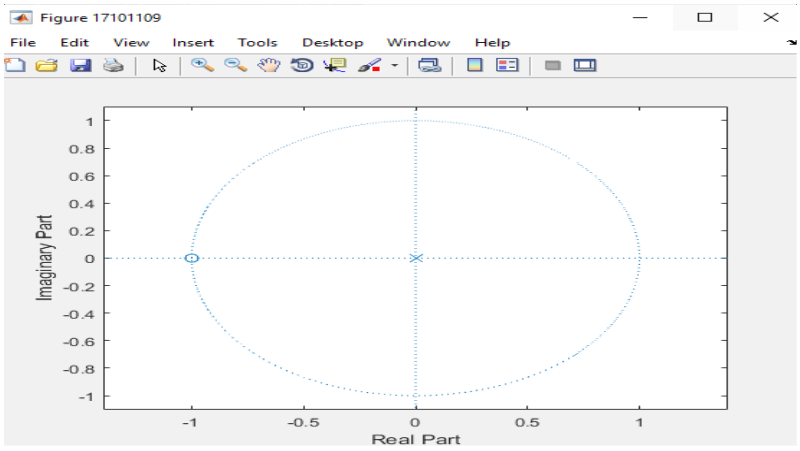
   Fast Fourier Transform (FFT) adalah teknik perhitungan cepat dari DFT. Untuk pembahasan FFT ini, akan dijelaskan FFT untuk 1D dan FFT untuk 2D. Dimana FFT 2D adalah pengembangan dari DFT 2D. Dengan kata lain FFT adalah DFT dengan teknik perhitungan yang cepat dengan memanfaatkan sifat periodikal dari transformasi *fourier*. Definisi dapat dituliskan dengan :

$$F(k)=\sum_{n=1}^{N} f(n)\cos(2\pi nkT/N)-j\sum_{n=1}^{N} f(n)\sin(2\pi nkT/N)$$

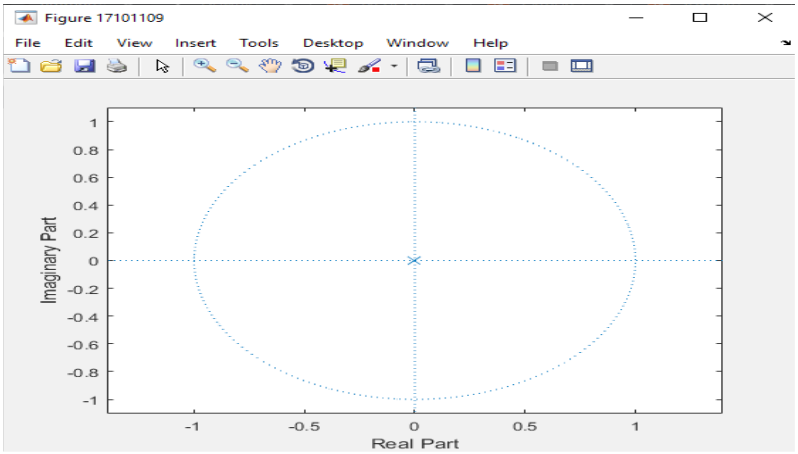
BAB II
HASIL DATA & ANALISA

Percobaan 1 :

Langkah pertama yang praktikan lakukan adalah mendeklarasikan sistem. Sistem yang diwakili oleh respon *impuls* h(n) = δ(n) + δ(n-1), dengan menuliskan *syntax* n = [0 1] ; lalu enter h_n = [1 1] ; ini berfungsi untuk menganalisis *pole* dan *zero* Setelah mendeklarasikan sistem, dengan menuliskan *syntax* n = [0 1] ; lalu enter h_n = [1 1] ;, langkah selanjutnya adalah menentukan nilai *pole* dan *zero* dengan menuliskan *syntax zplane* (h_n); setelah itu di *run*, maka akan muncul seperti gambar 2.1 yang merupakan daerah *pole* dan *zero*. Dari daerah tersebut dapat diketahui kestabilan dari sistem tersebut. Pada percobaan kedua sama saja seperti percobaan pertama hanya saja pada percobaan kedua h_n = [8 8] ; diganti dengan NIM masing-masing praktikan.



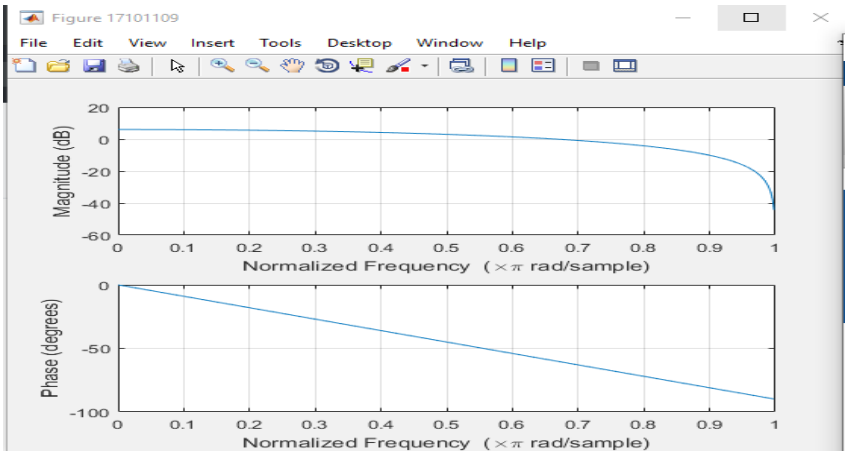
Gambar 2.1 Hasil Analisis *Pole* Dan *Zero*.



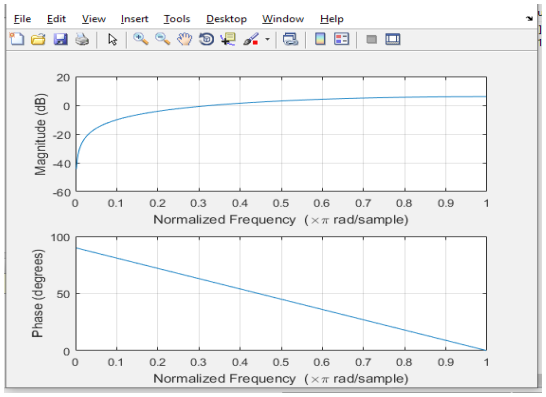
Gambar 2.2 Hasil Analisis *Pole* dan *Zero* dengan NIM

Percobaan 2 :

Untuk melihat respon frekuensi dari sistem yang telah dideklarasikan pada tahapan sebelumnya yaitu dengan menuliskan *syntax freqz* (h_n) ; fungsi ini adalah untuk melihat respon frekuensi dari sistem, lalu tekan *enter* dan kemudian akan muncul seperti gambar 2.3 yang merupakan respon frekuensi dari sistem. Dalam gambar terlihat dua buah grafik yaitu grafik respon *magnitude* yang merupakan sebuah sistem dasar untuk membuat sebuah *filter* yaitu LPF (*Low pass filter).* Dari analisa yang saya dapat bahwa jika lpf melewati angka [1 1] lalu hpf melewati angka [1 -1]



Gambar 2.3 Respon Frekuensi Dari Sistem lpf.

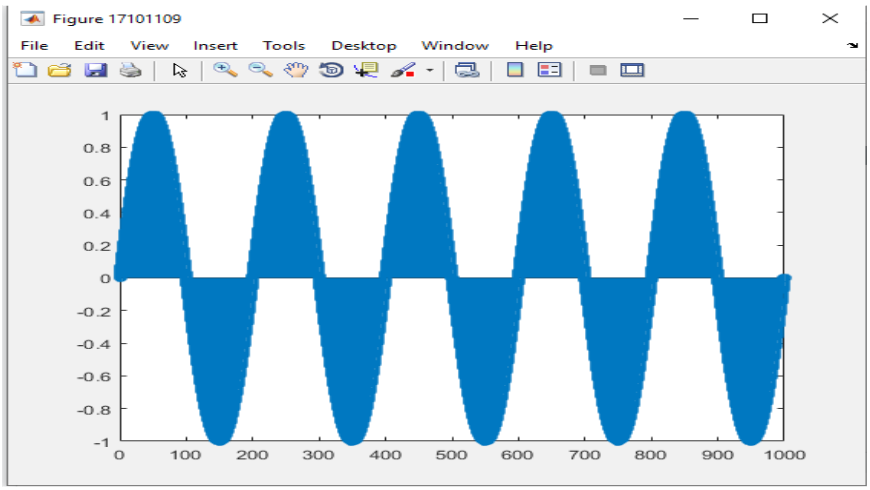


Gambar 2.4 Respon Frekuensi dari Sistem hpf

Percobaan 3 :

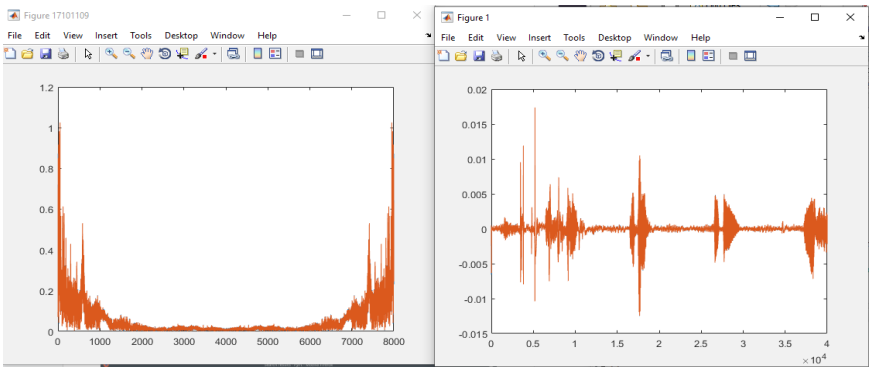
Langkah selanjutnya yaitu masuk ke transformasi *fourier*, dengan membuat program terlebih dahulu yang membangkitkan sinyal *sinusoidal* dengan frekuensi *sampling* yang berbeda. Masukkan *syntax fs*=8000 yang merupakan

inisialisasi frekuensi *sampling*, dengan menggunakan batas t=[0 :0 :01 :1000] ; yang merupakan batas t. Lalu masukan *syntax* selanjtunya jalankan program xt = sin (t*pi/100); yang merupakan membangkitkan sinyal *sinusoidal*. Kemudian masukan *syntax stem* (t,xt) ; yang merupakan *plot* sinyal dalam bentuk *dikrit* lalu masukan perintah *sound* (xt, fs) ; yang merupakan berguna untuk memainkan sinyal suara.



Gambar 2.5 Transformasi *fourier*.

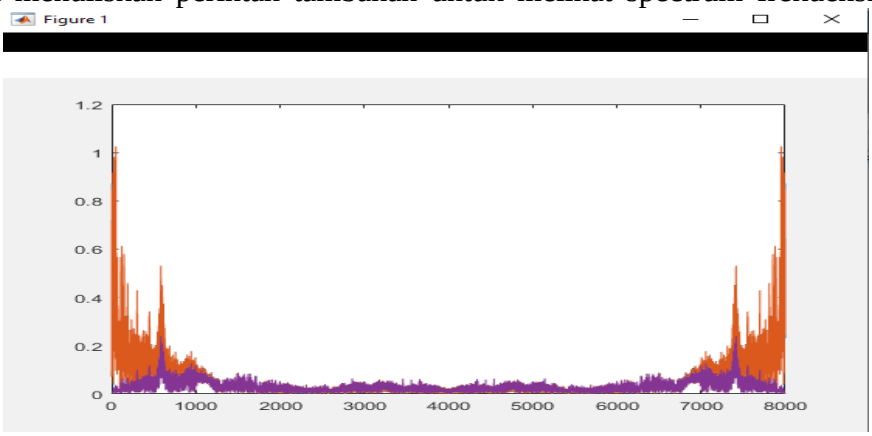
Percobaan 4 :



Gambar 2.6 Spektrum frekuensi.

Kemudian percobaan selanjutnya yaitu dapat dihasilkan rekaman suara yang dihasilkan pada *microphone* dengan frekuensi 8000Hz yang akan memainkan frekuensi tersebut, dengan menggunakan 16 bit dengan *2 channel stereo.* Dan terdapat *syntax plot* (abs(fft(x,fs)) yang digunakan untuk melihat spektrum frekuensi. Praktikan bisa menuliskan *syntax fs* = 8000; merupakan inisialisasi jumlah frekuensi, lalu praktikan menuliskan *syntax* bits = 16;

merupakan jumlah bit,ch=2; meruapakan *channel stereo,* lalu menuliskan rec=*audiorecorder* (fs,bits,ch); perintah ini berguna untuk merekan suara, disp (‘*start speaking’*); perintah ini berguna untuk menampilkan tulisan pada layar, *recordblocking* (rec,5); fungsi ini berguna untuk lamanya durasi pada rekaman, *play* (rec); fungsi ini berguna untuk memulai rekaman, *myRecording=getaudiodata* (rec); berguna untuk menyimpan hasil rekaman, *plot* (*myRecording*); berguna untuk memplot hasil rekaman, *audiowrite* (‘*recording.wav’,myRecording*,fs); berguna untuk nama rekaman *audio*, lalu praktikan menuliskan perintah tambahan untuk melihat spectrum frekuensi dari



suara. x=*audioread* (‘*recording*.wav’); berguna untuk membaca file sinyal suara dalam format, fs=8000; berguna untuk inisialisasi frekuensi *sampling*, *plot* (abs(fft(x,fs))); proses fft untuk melihat spektrum frekuensi, *sound* (x,fs); berguna untuk mendengarkan sinyal suara.

Gambar 2.7 filter respon *impuls* sebelum dan sesudah di filter

Lalu praktikan akan membuat program untuk menyaring suatu suara dengan menggunakan respon *impuls* sebagai filternya. Praktikan bisa menuliskan x=*audioread* (‘suaramodul2b.*wav’*); *syntax* ini berguna untuk membaca file sinyal suara yang telah praktikan buat, fs=8000; inisialisasi frekuensi *sampling*, hn=[1 -1]; inisialisasi respon *impuls*, *freqz* (hn); berguna untuk melihat respon frekuensi, *sound* (x,fs); mendengarkan sinyal suara, x=*audioread* (‘suaramodul12b.wav’); membaca file sinyal suara yang telah praktikan buat, fs= 8000; inisialisasi frekuensi *sampling*, *plot*(abs (fft(x,fs))); melihat spektrum frekuensi, *hold on*; berguna untuk menahan suatu *figure*, y = *filter* (hn,1,x); memfilter suara hasil rekaman, *plot* (abs(fft(y,fs))); berguna untuk melihat hasil *filter* melalui spectrum frekuensi, *sound* (y,fs); mendengarkan sinyal suara hasil

filter. Dari hasil yang praktikan dapat bahwa grafik pertama dan kedua berbeda karena yang kedua sudah di filter, grafik kedua hanya berisi frekuensi-frekuensi rendah sedangkan frekuensi tinggi sudah hilang akibat sudah di filter

BAB III
KESIMPULAN DAN SARAN

A. Kesimpulan

1. Dalam mencari nilai *pole* dan *zero* yang menentukan daerah kestabilan dari sebuah sistem yaitu menggunakan perintah *zplane*.
2. Dalam *transformasi fourier praktikan bisa menggunakan fungsi impuls sebagai filter untuk menyaring suatu suara.*
3. Transformasi Z merupakan suatu tekni untuk mempermudah analisa dan realisasi sinyal *diskrit*

B. Saran

1. Praktikan Sebaiknya praktikan aktif menanyakan sesuatu jika tidak dimengerti.
2. Praktikan diharapkan lebih fokus dalam menuliskan program dengann benar dan teliti agar program bisa berjalan
3. Sebaiknya terlebih dahulu mempelelajari modul supaya praktik lebih mengerti dan mudah.
4. Sebaiknya pada saat pelaksanaan praktikum, komputer yang ada di laboratorium sudah dalam keadaan siap pakai dan sudah terinstal aplikasi yang ingin digunakan padasaat praktikum.

LAMPIRAN

Tugas 1 :

1. Mengapa *transformasi* Z memerlukan RoC ?

Jawab : *Transformasi* Z memerlukan RoC untuk menghasilkan deret geometri yang *konvergen*. Untuk harga r = ׀z׀ tertentu, H(z) harus *kovergen*. Sebagai contoh untuk sinyal undak U(n), harga tidak *konvergen*, namun untuk r = > 1, adalah *konvergen*. Daerah harga z dimana X(z) *kovergen* dinamakan daerah *konvergen* (*Region of Convergence*; ROC). Dengan demikian *transformasi*-z dari sinyal u(n) mempunyai daerah *konvergen* ׀z׀> 1, termasuk z = ∞.

2. Bagaimana konsep dasar RoC pada *transformasi* Z ?

jawab : *Transformasi* Z adalah suatu deret tak hingga, sehingga mungkin *divergen* untuk beberapa nilai z. Suatu nilai daerah yang hasil transformasinya terhingga diberi nama RoC (*Region of Convengence*). Roc dari *transformasi* Z berbentuk : $R_1 < |Z| < R_2$, Dimana |Z| = r. Dengan batas R_1 dan R_2 tergantung pada sinyal yang ditransformasikan.

Tugas 3 :

1. Apa perbedaan antara deret *fourier* dengan *transformasi fourier*?

Jawab : **Deret *Fourier*** merupakan suatu fungsi periodik dapat dinyatakan sebagai deret tak hingga dari fungsi trigonometri sinus dengan *amplitudo* dan fase yang berbeda-beda. Suatu fungsi periodik $f(x)$ dapat dituliskan sebagai

$$f(x) = \sum_{n=0}^{\infty} A_n \sin(nx + \Phi_n)$$

Karena

$$\sin(nx + \Phi_n) = \sin(nx)\cos(\Phi_n) + \cos(nx)\sin(\Phi_n),$$

kita bisa mengekspresikan fungsi periodik sebagai penjumlahan dari fungsi *sinus* dan *cosinus*,

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty}\left(a_n \cos(nx) + b_n \sin(nx)\right)$$

Deret tersebut disebut **deret *Fourier***. Tiap suku dalam deret *Fourier* memiliki periode $\frac{2\pi}{n}$.

Sedangkan *Transformasi Fourier* merupakan suatu fungsi dengan periode tertentu dapat dinyatakan dalam deret *Fourier*. Tetapi, bagaimana dengan fungsi yang memiliki periode tak berhingga atau dengan kata lain tidak periodik? Kita dapat menganggap fungsi tersebut sebagai fungsi periodik dengan periode tak terhingga dan mengganti penjumlahan pada deret *Fourier* dengan *integral*. Metode ini disebut transformasi Fourier. Kemudian dapat melihat periodisitas sinyal tersebut setelah sinyal tersebut ditransformasi. Jika kita memiliki suatu fungsi $f(x)$, transformasi Fourier dari fungsi tersebut adalah : $g(k) = \int_{-\infty}^{\infty} f(x) e^{-ikx} dx$

$f(x) = \int_{-\infty}^{\infty} \frac{g(k)}{2\pi} e^{ikx} dx$

Sebagai contoh, terdapat suatu fungsi

$$f(x) = \begin{cases} 1 & (-a < x < a) \\ 0 & (\text{lainnya}) \end{cases}$$

Transformasi dari fungsi tersebut adalah

$$g(k) = \int_{-a}^{a} e^{-ikx} 1 dx = \frac{1}{-ik} \left[e^{-ika} - e^{ika} \right] = \frac{2 \sin ka}{k}$$

2. Apa perbedaan antara DTFT dan DFT?

   Jawab :

   a. *Discrete-Time Fourier Transform* (DTFT)

   DTFT adalah *transformasi Fourier* (konvensional) dari sinyal waktu diskrit. Hasilnya adalah spektrum frekuensi versi *kontinyu* dari sinyal tersebut. Misalnya: digunakan untuk mencari spektrum frekuensi dari sinyal hasil sampling x(kT) dari sinyal waktu *kontinyu* x(t).

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]\, e^{-i\omega n}$$

   Deskripsi: mencari spektrum frekuensi versi *kontinyu* X(ω) dari sinyal waktu diskrit x[n] dengan menggunakan deret kontinyu dari −∞ s.d. +∞.

   b. *Discrete Fourier Transform* (DFT)

DFT adalah versi diskrit dari hasil DTFT. DFT digunakan untuk menghitung spektrum frekuensi dari sinyal waktu *diskirt* dengan komputer, karena komputer hanya bisa melibatkan bilangan berbatas.

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-\frac{i2\pi kn}{N}}, \quad k \in \mathbb{Z}\,(\text{integers})$$

Deskripsi: mencari spektrum frekuensi versi diskrit X[k] dari sinyal waktu *diskrit* x[n] dengan menggunakan deret *diskrit* dari 0 s.d. N-1.