Universidade Federal de Santa Catarina - UFSC
Centro Araranguá - ARA
Departamento de Computação - DEC
Análise de Sinais e Sistemas
Projeto : DFT e FFT

Regras:

- Projeto Opcional.
- 1,0 ponto adicional na prova P_2
- O projeto deve ser apresentado até a data final de entrega.
- Em caso de plágio, cópia ou suspeita de plágio ou cópia **perde 1 ponto** na prova P_2.
- Data final de entrega e apresentação: 29 de novembro de 2016.

1 Definição do Problema

Uma das ferramentas mais usadas em processamento digital de sinais é a transformada de Fourier Discreta. Definida como:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}nk}, \text{ para } k = 0, 1, 2, \cdots, N-1.$$

No entanto, a implementação direta da equação acima é pouco eficiente. Para contornar este problema de pouca eficiência, usamos a transformada rápida de Fourier (FFT).

2 Parte 1: DFT

Implemente em linguagem C a DFT usando a definição acima.

3 Parte 2: FFT

Implemente em linguagem C a FFT conforme detalhamento em sala de aula.

4 Testes e Validação

- Use os includes teste1.h, teste2.h e teste3.h para validar os dois algoritmos.
- Use a biblioteca time.h, para medir o tempo de execução, dos laços principais, dos dois algoritmos (desconsiderando leitura de dados e inicializações de variáveis).
- Use o arquivo.dat como sinal de entrada para medir os tempos de execuções dos algoritmos.

5 Relatório:

- Código fonte dos algoritmos.
- Tabela com os tempos de execuções para os dois algoritmos nos quatro cenários de teste.

6 Revisão de números complexos:

Para cálculos com dois números complexos $z_1 = a_1 + jb_1$ e $z_2 = a_2 + jb_2$ que produz um terceiro número $z = a + jb$, temos:

- Soma:
$$z = (a_1 + a_2) + j(b_1 + b_2)$$
- Subtração
$$z = (a_1 - a_2) + j(b_1 - b_2)$$
- Multiplicação:
$$z = (a_1 a_2 - b_1 b_2) + j(a_1 b_2 + a_2 b_1)$$

- Divisão

$$z = \frac{(a_1a_2 + b_1b_2) + j(a_2b_1 - a_1b_2)}{a_2^2 + b_2^2}$$

Para representação polar temos:

- Módulo

$$|z| = \sqrt{a^2 + b^2}$$

- Fase

$$\theta = \arctan \frac{b}{a}$$

6.1 Funções Complexas

```
typedef struct{
 double a, b;
} complex;
```

e os protótipos

```
complex soma(complex z1, complex z2);
complex subtr(complex z1, complex z2);
complex mult(complex z1, complex z2);
complex div(complex z1, complex z2);
double modulo(complex z);
double angle(complex z);
void mostra(complex z);
complex le(void);
```