1

Enabling Technologies for Sports (5XSF0) Module 2

Signals, sampling, Fourier series

Arash Pourtaherian

(a.pourtaherian@tue.nl)

2

Descriptive Statistics

* **M = mean(x)**
  – Calculates the average of a signal
* **V = var(x)**
  – Calculates the variance of a signal
* **M = median(x)**
  – Calculates the median of a signal
* **D = diff(x)**
  – Differences of two consecutive samples

Descriptive Statistics

* Moving average
  – Example of a 3 point moving average filter:

```
F = ones(3,1)/3;
M = filter(F,1,x);
```

* **`y = medfilt1(x,n)`**
  – Applies an order-n one-dimensional median filter to the input vector, x.

Descriptive Statistics

* Angle of rise/fall
  – Slope between two data points (X1,Y1) and (X2,Y2)

```
S = (Y2-Y1 / X2-X1);
```

  – Angle of the slope

```
angle = acosd(S);
```

  Calculates the Inverse cosine in degrees

Sine / cosine signals

* Sinusoidal signals are cornerstones of signal models and used in many real life applications…

Mathematical formula continuous-time cosine:

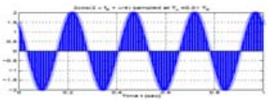
$$x(t) = A\cos(\omega_0 \cdot t + \phi) = A\cos(2\pi \cdot f_0 \cdot t + \phi)$$

Symbol	Name	Dimension
A	Amplitude	-
ω_0	Radian frequency	rad/sec
ϕ	phase	rad
f_0	(cyclic) frequency	$\sec^{-1}$= Hz

Sine / cosine signals – (2)

$x(t) = 2\cos(2\pi \cdot 4 \cdot t + \pi/4) \rightarrow$ One period? T_0=1/4 =0.25 [sec]

$T_s = 0.0025$ [sec] $\rightarrow$ 100 samples in T_0



* To create a sinusoid function in Matlab, we first need to define a time variable `t` and calculate `sin` and `cos` for frequency `f`. One period is T_0=`1/f`:
* `x1 = sin(2*pi*f*t);`
* `x2 = cos(2*pi*f*t);`

Sine / cosine signals – (3)

* **Plotting & sampling. An example:**

$$x = 2\cos(2\pi 4t + \pi/4)$$

$T_s = 0.0025$ [sec] $\rightarrow$ 100 samples in T_0

```
t = 0:0.0025:1;
x = 2*cos(2*pi*4*t + pi/4);
plot(t,x);
```


Read/write signals

* `load('filename')`
* `save('filename','variables')`
  - Option for the format –

    `save('filename', 'variables', fmt)`

Value of `fmt`	File Format
`'-mat'`	Binary MAT-file format.
`'-ascii'`	Text format with 8 digits of precision.
`'-ascii','-tabs'`	Tab-delimited text format with 8 digits of precision.
`'-ascii','-double'`	Text format with 16 digits of precision.
`'-ascii','-double','-tabs'`	Tab-delimited text format with 16 digits of precision.

Spectral Analysis

* **findpeaks**
  - Example: `[pks, locs] = findpeaks(X);`
  - Useful options –

```
[pks, locs] =
findpeaks(X,MinPeakHeight’,m)
```


1D Filtering

* **Y = filter(b,a,X)**
  - `b` and `a` are filter parameters, which can be designed by other functions.
  - Example: Low-pass Butterworth filtering

```
[b,a] = butter(6,0.6);
dataOut = filter(b,a,dataIn);
```

1D Fourier analysis

* **`F = fft(x)`**
  – returns the discrete Fourier transform (DFT) of vector `x`, computed with a fast Fourier transform (FFT) algorithm.
  – Fourier spectrum: `S=abs(F)`
  – Inverse Fourier transform: `f=ifft(F)`

1D Fourier analysis

* **`F = fft(x)`**
  – returns the discrete Fourier transform (DFT) of vector `x`, computed with a fast Fourier transform (FFT) algorithm.
  – Fourier spectrum: `S=abs(F)`
  – Example:

```
Y = abs(fft(x))/length(x);
F = Fs*(1:length(x))/length(x);
plot(F, Y);
```