Digital Filter

HuStar

2020.07.

이성윤

평균 필터

- 측정한 전체 샘플 k개를 평균 내어($\bar{x}_k$) 정적인 데이터(x)의 참값을 추정($\hat{x}_k$)
- 배치식, Batch expression:

$$\hat{x}_k = \bar{x}_k = \frac{x_1 + x_2 + \cdots + x_k}{k}$$

- 재귀식, Recursive expression:

$$\hat{x}_k = \bar{x}_k = \frac{1}{k}(x_1 + x_2 + \cdots + x_k) = \frac{1}{k}\{(k-1)\bar{x}_{k-1} + x_k\}$$

$$\therefore \hat{x}_k = \frac{k-1}{k}\bar{x}_{k-1} + \frac{1}{k}x_k = \alpha\bar{x}_{k-1} + (1-\alpha)x_k$$

Example 1

- 10V 배터리의 전압을 측정하는데 잡음이 심해 평균 필터를 사용하여 표시하려고 한다. 재귀식을 구현해보자.
  - 0.1초 간격으로 20초 동안 측정, 잡음의 평균은 0.0V, 표준편차는 2.0V로 가정 (정규분포)
  - 배터리 전압값 생성식: data = 10.0 + 2*randn(1);
  - persistent 선언으로 재귀식 함수를 구현하며, raw data와 filtered data를 비교 (avg = AvgFilter(data)와 같이 사용할 수 있도록 구현)

이동평균 필터

- 측정한 전체 샘플 k개 중에 최근 n개만 평균 내어($\bar{x}_n$) 동적인 데이터(x)의 참값을 추정($\hat{x}_k$)
- 배치식, Batch expression:

$$\hat{x}_k = \bar{x}_n = \frac{x_{k-(n-1)} + x_{k-(n-2)} + \cdots + x_{k-1} + x_k}{n}$$

- 재귀식, Recursive expression:

$$\hat{x}_k = \bar{x}_n = \frac{1}{n}\left(x_{k-(n-1)} + x_{k-(n-2)} + \cdots + x_{k-1} + x_k\right) = \frac{1}{n}\left(x_{k-n} + x_{k-(n-1)} + x_{k-(n-2)} + \cdots + x_{k-1} + x_k - x_{k-n}\right)$$

$$= \frac{1}{n}\left(x_{k-n} + x_{k-(n-1)} + x_{k-(n-2)} + \cdots + x_{k-1}\right) + \frac{1}{n}(x_k - x_{k-n})$$

$$\therefore \hat{x}_k = \bar{x}_{k-1} + \frac{1}{n}(x_k - x_{k-n})$$

Example 2

- IR 센서로 거리를 측정하는데 잡음이 심해 Moving Average Filter를 사용하여 표시하려고 한다. 이동 평균 필터를 구현해보자.
  - Filter size는 10 sample
  - ‘IR_data.mat’에서 측정값을 읽어서 실행 (load(‘IR_data.mat’);)

Example 3

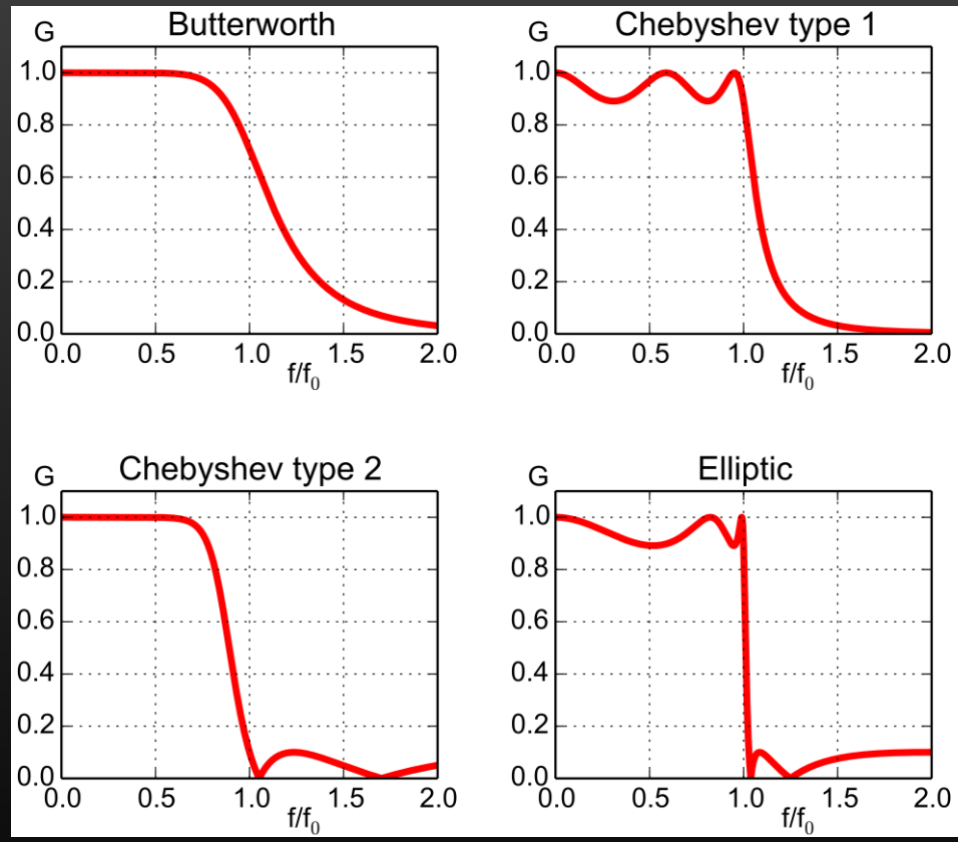
- Ex2에서 구현한 moving average filter에서 filter size를 변경하여 실행할 수 있도록 수정해보자.
  - Filter size를 2, 10, 40으로 설정한 세 가지 결과를 비교

평균 필터의 비교

Average Filter	Moving Average Filter
$\bar{x}_k = \frac{k-1}{k}\bar{x}_{k-1} + \frac{1}{k}x_k$	$\bar{x}_k = \bar{x}_{k-1} + \frac{x_k - x_{k-n}}{n}$
• Static input에 대한 실시간 LPF • Data에 k분의 1의 같은 weight 적용 • Noise의 평균은 0이라는 가정 • Calibration에 적용 가능	• Dynamic input에 대한 실시간 LPF • Data에 n분의 1의 같은 weight 적용 • n 증가: rough + 최근 값 민감 • n 감소: smooth + 최근 값 둔감 • 잡음 제거와 민감성을 동시에 달성 힘듦

아날로그 필터

- 필요한 passband(통과영역), cutoff slope(기울기), stopband(제거영역) 특징에 따라 필터들 중에서 골라서 사용
  - 통과영역: Butterworth > Chebyshev > Elliptic
  - 기울기: Elliptic > Chebyshev > Butterworth
  - 제거영역: Chebyshev> Butterworth > Elliptic



Butterworth Filter

$$G(s) = \frac{G_0}{\prod_{k=1}^{n} \frac{1}{\omega_c}(s - s_k)}$$

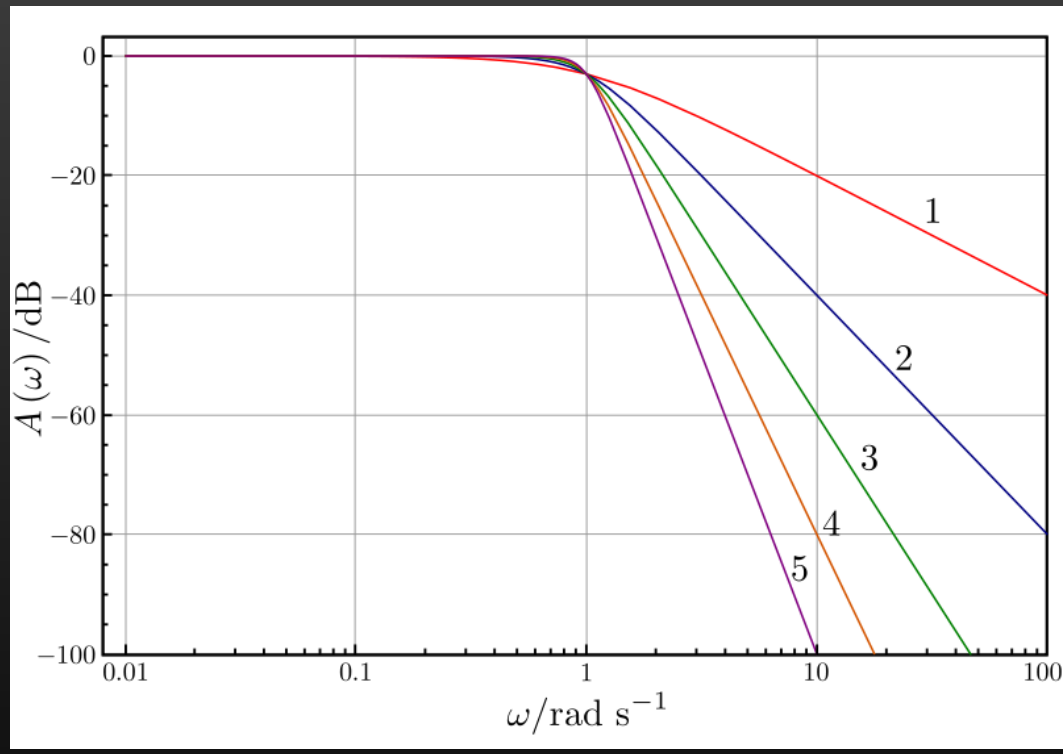
- K[th] Pole : $s_k = \omega_c e^{\frac{j(2k+n-1)\pi}{2n}}$

  Cutoff frequency : $\omega_c = 2\pi f_c$

  DC gain : G_0

  Filter order : n

- 주요 특징
  - 부드러운 이득 곡선, 작은 overshoot, 차수를 통한 필터링 강도 조절
  - 적절한 특성으로 가장 많이 사용되는 필터



Example 4

- Cut-off frequency가 4 rad/sec인 1차, 2차, 3차, 4차 butterworth filter를 구현하여 bode plot을 그리고, ‘IR_data.mat’의 데이터가 필터를 거친 후의 결과를 비교해보자.

Other Filters

- 일반적으로 low-pass filter의 전달함수를 1에서 빼면 동일한 cut-off frequency를 갖는 high-pass filter가 됨

$$G_{HPF}(s) = 1 - G_{LPF}(s)$$

- 이상적인 적분기를 전달함수로 $H_{int}(s) = \frac{1}{s}$와 같이 나타낼 수 있으나, 미분기에 해당하는 $H_{diff}(s) = s$는 improper로, 실제로는 구현 불가능
- 따라서, high-pass filter를 이용하여 미분과 유사한 전달함수를 만들어 사용
  - $H(s) = \frac{s}{\alpha s+1} = s \cdot \frac{1}{\alpha s+1}$
  - 미분에 low-pass filter를 연결한 형태

Example 5

- 1 rad/sec의 사인함수를 전달함수를 이용하여 적분하고 미분한 뒤 비교해보자.
  - 미분 전달함수에서 α=0.01

수치미분법

- 1차 버터워스 필터의 전달함수를 실제 사용하기위해 시간영역으로 변환하면 다음과 같다.

$$G(s) = \frac{1}{\frac{1}{\omega_c}s + 1} = \frac{Output(s)}{Input(s)} = \frac{Y(s)}{X(s)}$$

$$\left(\frac{1}{\omega_c}s + 1\right)Y(s) = X(s)$$

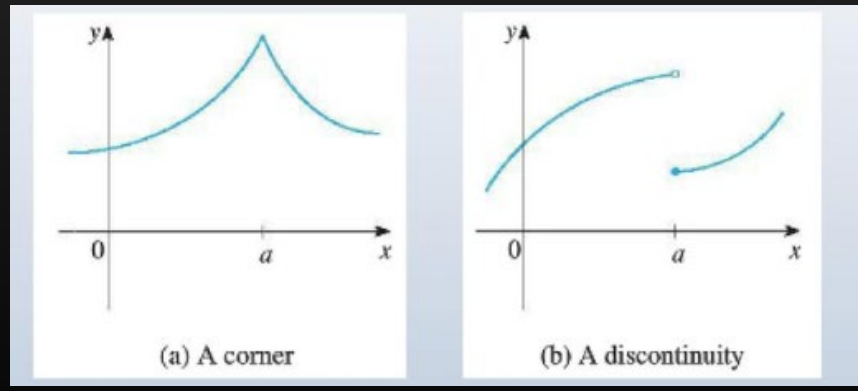
$$\therefore \frac{1}{\omega_c}\frac{dy(t)}{dt} + y(t) = x(t)$$

- 여기서 $\frac{dy(t)}{dt}$를 구현해야 하는데, 수학적으로 미분은 극한 값으로 정의되어 있다.

$$\frac{dy(t)}{dt} = \lim_{\Delta t \to 0} \frac{y(t + \Delta t) - y(y)}{\Delta t}$$

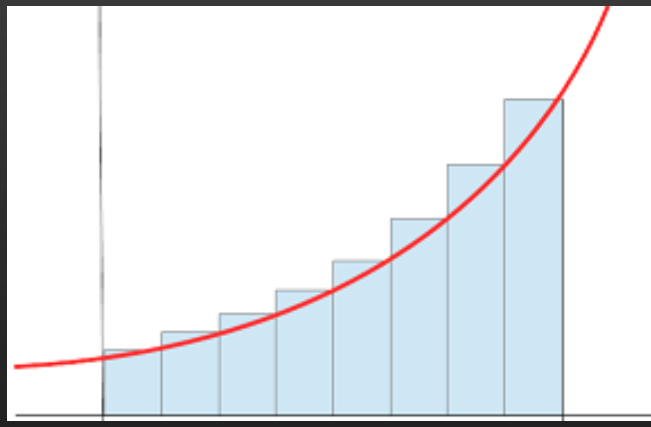
수치미분법

- 미분이 존재하는 경우는 함수가 좌극한과 우극한이 같고, smooth한 모양이어야 하는 조건이 있다.
- 따라서 이상적인 미분은 여러가지 이유로 real-life application에서 적용하기 힘들다.
  1) 우극한을 알기 위해서 미래의 값을 알아야하기 때문
  2) 신호 계측은 항상 discrete(이산) 영역에서 일어나기 때문
- 미분 값을 근사하는 몇가지 간단한 방법들을 소개한다.
  1) Euler's backward rectangle method
  2) Euler's forward rectangle method
  3) Trapezoid method (Tustin's method)



Euler’s Backward Rectangle Method

$$\frac{dy}{dt} = f(t) \;\leftrightarrow\; y = \int f(t)dt$$



- 도함수에서 높이의 차이는 원시함수의 area under the curved와 같다.
- 원시함수의 면적을 계산할 때, 시간의 뒤쪽으로 향하는 사각형을 그려 근사 값을 구한다.

$$y(t_k) - y(t_{k-1}) = \int_{t_{k-1}}^{t_k} f(t)dt \approx \Delta t f(t_k)$$

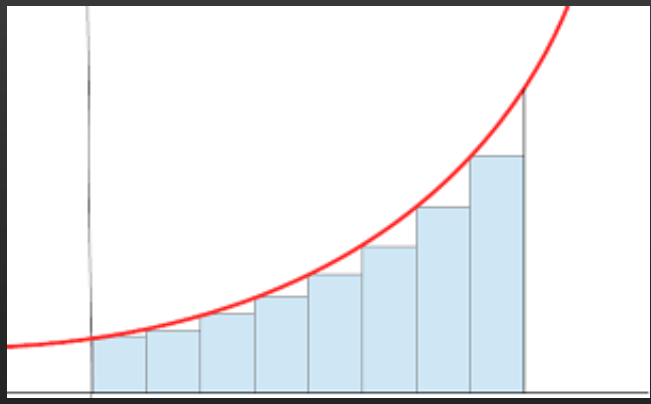
- 따라서 미분은 다음과 같이 근사하여 표현할 수 있다.

$$\therefore f(t_k) = \left.\frac{dy}{dt}\right|_{t=t_k} \approx \frac{y(t_k) - y(t_{k-1})}{\Delta t}$$

  - Euler's backward rectangle method으로 미분을 근사하는 방법은 현재 값과 과거 값의 차이를 두 값의 시간 차이로 나누는 것이다.
  - 방법이 매우 간단하기 때문에 미분을 근사하기 위해 많이 사용된다.
  - 미분을 구하기 위해 초기값을 설정해야 한다.

Euler’s Forward Rectangle Method

$$\frac{dy}{dt} = f(t) \;\leftrightarrow\; y = \int f(t)dt$$



- 도함수에서 높이의 차이는 원시함수의 area under the curved와 같다.
- 원시함수의 면적을 계산할 때, 시간의 앞쪽으로 향하는 사각형을 그려 근사 값을 구한다.

$$y(t_k) - y(t_{k-1}) = \int_{t_{k-1}}^{t_k} f(t)dt \approx \Delta t f(t_{k-1})$$

- 따라서 미분은 다음과 같이 근사하여 표현할 수 있다.

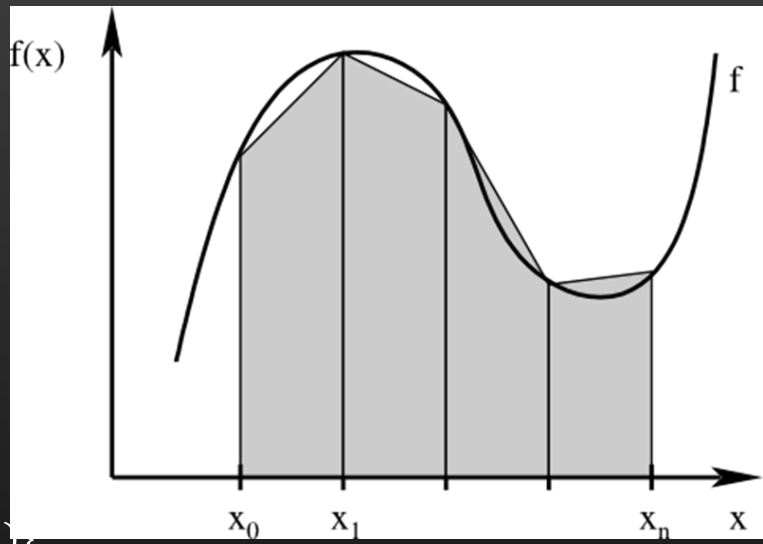
$$f(t_{k-1}) = \left.\frac{dy}{dt}\right|_{t=t_{k-1}} \approx \frac{y(t_k) - y(t_{k-1})}{\Delta t}$$

$$\therefore f(t_k) = \left.\frac{dy}{dt}\right|_{t=t_k} \approx \frac{y(t_{k+1}) - y(t_k)}{\Delta t}$$

  - Euler’s forward rectangle method으로 미분을 근사하는 방법은 미래 값과 현재 값의 차이를 두 값의 시간 차이로 나누는 것이다.
  - 비슷하게 간단한 방법이지만 미래의 값을 사용해야 하기 때문에 실시간 적용은 불가능하다.

Trapezoid Method

$$\frac{dy}{dt} = f(t) \;\leftrightarrow\; y = \int f(t)dt$$



- 도함수에서 높이의 차이는 원시함수의 area under the curved와 같다.
- 원시함수의 면적을 계산할 때, 사다리꼴을 그려 근사 값을 구한다.

$$y(t_k) - y(t_{k-1}) = \int_{t_{k-1}}^{t_k} f(t)dt \approx \Delta t f(t_{k-1}) y(t_k) - y(t_{k-1}) = \int_{t_{k-1}}^{t_k} f(t)dt \approx \frac{\Delta t}{2}\{f(t_k) + f(t_{k-1})\}$$

- 따라서 미분은 다음과 같이 근사하여 표현할 수 있다.

$$\therefore f(t_k) = \left.\frac{dy}{dt}\right|_{t=t_k} \approx \frac{2}{\Delta t}\{y(t_k) - y(t_{k-1})\} - \left.\frac{dy}{dt}\right|_{t=t_{k-1}}$$

  - 직사각형을 그리는 Euler's method보다 일반적으로 계산 정확도가 높다.
  - 재귀함수 형태로 초기값이 정확해야 함

Example 6

- 수치미분법 세 가지를 이용하여 주파수가 1rad/sec인 사인파의 미분을 수행하고 비교해보자.
  - Trapezoid method의 경우, 초기값을 0으로 둘 때와 1로 둘 때를 나누어서 실행해볼 것

Example 7

- 수치미분법 세 가지의 개념을 참고하여 수치적분법 세 가지를 만들고, 주파수가 1rad/sec인 사인파의 적분을 수행해보자.
  - 구분구적법을 수식으로 나타내는 것과 동일

디지털 필터

- 디지털 필터는 샘플링된 데이터를 활용하므로, 실질적인 구현에서 매우 중요한 위치를 차지한다.
- 지금까지 살펴본 디지털 필터는 입력과 출력의 계수를 통한 재귀식이라는 공통 구조를 가지며, 일반화할 수 있다.

$$\sum_{i=0}^{N} y[n-i]a_i = \sum_{j=0}^{M} x[n-j]b_j$$

$$y[n] = \sum_{j=0}^{M} x[n-j]\frac{b_j}{a_0} - \sum_{i=1}^{N} y[n-i]\frac{a_i}{a_0}$$

- 일반식에서 $a_i = 0,\ i > 0$인 경우를 FIR(finite impulse response)라 부르며, 반드시 수렴하는 특성을 갖는다.
- 일반식에서 $a_i \neq 0,\ i > 0$인 경우를 IIR(infinite impulse response)라 부르며, 계수에 따라 수렴하지 않는 경우가 발생한다.

Z-Transform과 전달함수

- 디지털 필터를 표현하는 방식 중 가장 유용한 방법으로 z-transform이 있다.

$$X(z) = \mathcal{Z}\{x[n]\} = \sum_{n=-\infty}^{\infty} x[n]z^{-n}, \qquad z = Ae^{j\phi}$$

- Z-transform은 DT에서 시점에 대한 차이를 z의 차수로 표현할 수 있어 유용하며, 전달함수의 형태로 표현할 수도 있다.

$$x[n-k] = z^{-k}X(z)$$

$$\sum_{i=0}^{N} y[n-i]a_i = \sum_{j=0}^{M} x[n-j]b_j$$

$$Y(z)\sum_{i=0}^{N} z^{-i}a_i = X(z)\sum_{j=0}^{M} z^{-j}b_j$$

$$\frac{Y(z)}{X(z)} = \frac{\sum_{j=0}^{M} z^{-j}b_j}{\sum_{i=0}^{N} z^{-i}a_i}$$

CT 전달함수의 DT 변환

- Bilinear transform을 통해 CT 전달함수(라플라스 변환)를 DT 전달함수(z-transform)로 변환할 수 있으며, 이를 통해 아날로그 시스템을 디지털로 근사할 수 있다.

$$z = e^{sT} = \frac{e^{\frac{sT}{2}}}{e^{-\frac{sT}{2}}} \approx \frac{1+\frac{sT}{2}}{1-\frac{sT}{2}}$$

$$s = \frac{2}{T} \cdot \frac{1-z^{-1}}{1+z^{-1}}$$

- DT 전달함수는 재귀식으로 변환하여 실시간 시뮬레이션에 활용할 수 있다.

Example 8

- 다음 회로의 전달함수를 구하고, 디지털로 변환하여 재귀식을 만들어보자.
  - R=100Ω, C=0.01F
- 1초부터 2초 사이에 크기 1을 갖는 펄스를 입력하여 시뮬레이션한다. 재귀식을 통한 결과와 Isim을 통한 결과를 비교해보자.