

신호 처리를 위한 R활용서[†]

박노진¹

¹단국대학교 응용통계학과

접수 2017년 7월 13일, 수정 2017년 9월 5일, 게재확정 2017년 9월 7일

요약

신호 처리는 전자 공학의 한 분야로서 시계열 분석과 매우 깊은 연관성을 갖고 있다. 신호 처리를 위해 현재는 상용 프로그램이 널리 사용되고 있는데 적지 않은 비용을 지불해야 한다. 반면 통계 분석을 위해서는 R이라는 무상 프로그램이 널리 사용되고 있다. 본 연구에서는 신호 처리를 위한 R-프로그램 활용서를 만들어 보았다. 본 연구의 구성은 신호 처리에 대한 이론을 간단히 정리하고 적당한 예제들을 택하여 R을 활용하여 구현하는 방식으로 되어있다. 각 단원에 서술된 이론을 읽고 해당하는 그림과 R 코드를 제8장에서 찾아서 구현해보는 방법으로 본 활용서를 사용할 수 있겠다. 연구의 순서는 (1). 푸리에 변환과 역변환 (2). 스펙트럼분석 (3). 비모수적 주기도와 모수적 주기도 (4). 필터의 구성과 소음처리 순서로 되어있다. 본 활용서는 신호 처리를 완벽하게 이해하기에는 분명 부족하지만 처음 시작하는 단계로서 도움이 됐으면 한다.

주요용어: R-패키지, 시계열, 푸리에 변환, 필터링.

1. 서론

현재까지 신호 처리를 위해 MATLAB이라는 가장 선호되는 소프트웨어가 사용되고 있다. 문제는 MATLAB을 사용하는데 많은 비용을 지불해야 한다는 것이다. 그런데 무료인 R을 사용하여 신호 처리가 가능하게 되었다. 이미 많은 연구자들이 R을 사용하여 신호 처리를 수행하고 있으나 잘 정리된 문서를 찾기 어렵다. R을 사용하여 신호 처리를 처음 수행하는데 도움이 되기를 원하는 마음에서 짧은 활용서를 만들었다.

신호 (signal)이라 함은 심전도, 강우량, 주식가격 등 시간의 흐름에 따라 나타나는 물리적 양을 의미한다. 우선 시간이 연속적인가 혹은 이산적인가에 따라 연속 시간 신호 혹은 이산 시간 신호로 나눈다. 통계학 연구자에게 익숙한 시계열은 이산 시간 신호에 해당한다고 하겠다. 본 활용서에서도 이산시간 신호를 다루는 방법만 소개하려고 한다.

신호는 '주파수 (frequency)'와 '주기 (period)'라는 특성치를 갖는데 주파수는 단위시간당 신호의 변화를 의미하고 주기는 신호가 반복되는 데 걸리는 시간이라고 간단히 정의할 수 있다. 통계학에서의 시계열 분석이 주기에 집중한다면 공학에서는 주파수에 관심이 더 많다고 하겠다. 그 이유 중 하나는 주기의 정의역이 무한대인 반면 주파수의 정의역은 0에서 1 혹은 -1/2에서 1/2로 유한하여 공학적으로 다루기 가능하다는 것이다. 그래서 시간 영역에서 신호가 주어지면 주파수 영역으로 변환을 하여 분석 및 공학적 작용을 가한 후 다시 시간 영역으로 역변환 하는 과정을 수행하게 된다. 이 과정에서 시계열 분석

[†] 이 연구는 2017학년도 단국대학교 대학연구비 지원으로 연구되었음.

¹ (16890) 경기도 용인시 기흥구 죽전동 단국대학교, 응용통계학과, 교수. E-mail: rjpak@dankook.ac.kr

과 공유되는 내용이 많이 있다. 국내에서도 Ha (1999), Kim 등 (2015)에 의해 공간상에서의 분석 그리고 기상에 관한 분석들이 각각 시도되었다.

본 활용서의 서술 방식은 신호를 처리하는 방법에 대한 간단한 기술을 하고 관련된 예제를 택하여 R을 이용하여 풀어내는 방법을 제시하였다. 각 절에 서술 된 이론은 읽고 해당하는 그림과 R 코드를 제8장에서 찾아서 구현해보는 방법으로 본 활용서를 사용할 수 있겠다. 논문의 구성은 (1). 푸리에 변환과 역변환 (2). 스펙트럼분석 (3). 비모수적 주기도와 모수적 주기도 (4). 필터의 구성과 소음처리 순서로 되어있다.

신호 처리에 대한 일반적인 내용은 Hayes (1996), Lee (2013), Oppenheim과 Schaffer (1999) 그리고 Proakis와 Manolakis (1996)를 참조하였다. 신호에 대한 시계열적 설명은 Bloomfield (1976)와 Priestley (1981)을 참조하였다. 웹사이트 <http://www.wikipedia.com>, <http://www.dsprelated.com> 그리고 <http://www.gwinst.com>의 내용도 참조하였다.

2. 이산 푸리에 변환과 역변환

이산 시간 신호 $x[n]$ 이 $n = 0, \dots, N - 1$ 과 같은 유한 시간에 대하여 주어지면,

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-j2\pi kn/N}$$

를 DFT (discrete fourier transform; 이산 푸리에 변환)라 정의하고 IDFT (inverse DFT; 이산 푸리에 역변환)를 통해 본래 신호

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] \cdot e^{j2\pi kn/N}$$

가 얻어진다.

오일러의 공식에 의해

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot \{\cos(-2\pi kn/N) + j \sin(-2\pi kn/N)\},$$

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] \cdot \{\cos(2\pi kn/N) + j \sin(2\pi kn/N)\}$$

로 표시할 수도 있다. 통계학에서 다루는 시계열은 DFT의 대상으로 이해될 수 있다. $X[k]$ 는 복소수 형태를 띠고 있어서 실수부분과 허수부분으로 나누어 그림을 그릴 수 있고 $|X[k]|$ 를 이용하여 한 번에 그리기도 한다. 특별히 크기 (magnitude)의 제곱 ($|X[k]|^2$)을 평균 전력 (average power) 혹은 전력이 라고 하고 모든 k 에 대하여 전개한 것을 전력 스펙트럼 (power spectrum)이라고 한다.

예제로서 이산 시간 신호 $x[n] = 5 \sin(2\pi n/10) + 5 \sin(2\pi n/25)$, $n = 0, \dots, 49$ 에 대하여 전력 스펙트럼을 구하면 $k=2$ 와 5에서 강한 막대기를 볼 수 있고 이는 신호의 주파수 $1/25$ ($=2/50$)와 $1/10$ ($=5/50$)에 해당한다 (Figure 8.1).

이제 $x[n]$ 에 백색잡음 ($\epsilon[n]$)을 더한 $y[n] = x[n] + \epsilon[n]$ 의 DFT인 $Y[k]$ 를 fft 함수를 이용하여 구하고 길이가 3인 이동평균의 DFT인 $H[k]$ 를 $Y[k]$ 에 곱한 후 역변환 (IDFT)을 ifft 함수를 이용하여 백색잡음을 감소시킨 신호를 구하는 과정을 Figure 8.2에 그려보았다. 이동평균의 DFT는 k 가 15에서 35 근방의 $Y[k]$ 의 값을 제한하는 효과를 내며 역변환을 취하면 위아래로 크게 존재하는 신호값 들의 안쪽으로

형성되는 이동평균값들을 얻게 되는데 어느 정도 잡음이 해소되어 Figure 8.2의 원신호와 비슷한 신호를 얻을 수 있다.

3. Z-변환

주어진 순열 $\{\dots, x[-1], x[0], x[1], \dots\}$ 을 간단한 함수 형식으로 바꾸어주는 변환으로 z -변환 (transformation)이 아래와 같이 정의된다.

$$X(z) = \sum_{n=-\infty}^{\infty} x[n]z^{-n}.$$

예를 들어, 순열 $\{x[n] = a^n; n = 0, 1, 2, \dots\}$ 혹은 $x[n] = a^n u(n)$, $u(n) = 1, n = 0, 1, 2, \dots$ 에 대한 z -변환은

$$X(z) = \sum_{n=0}^{\infty} a^n u(n) z^{-n} = \sum_{n=0}^{\infty} (az^{-1})^n$$

가 되고 만일 $|az^{-1}| < 1$ 이라면

$$X(z) = \frac{1}{1 - az^{-1}} = \frac{z}{z - a}, \quad |z| > |a|$$

로 표현된다. 즉, 시간 축에서 표현되던 순열이 이차원 (혹은, 복소수 차원)에서 하나의 함수로 표현된다. 이때, $|z| > |a|$ 를 수렴구역 (region of convergence)이라고 한다. 때때로 $z = e^{j\Omega}$ 로 놓고 z -변환을 다시 쓰면

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n}$$

가 되고 이 급수가 수렴하면 이를 푸리에 변환 (fourier transform)이라고 한다. 푸리에 변환은 z -변환에서 z 가 반지름이 1인 원이라고 할 때와 동일하다. 위의 예제에서 만일 $|a| < 1$ 이면 (즉, 수렴구역이 반지름이 1인 원이 포함한다면) 푸리에 변환이 존재한다. 또한, 오일러의 공식을 이용하면

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x[n](\cos(\Omega n) - j \sin(\Omega n))$$

라고 표현할 수 있고 적분을 이용하여

$$x[n] = \frac{1}{2\pi} \int_{2\pi} X(\Omega) e^{j\Omega n} d\Omega$$

을 역으로 구할 수 있다. 위의 과정을 이산 시간 푸리에 변환 (discrete time fourier transform; DTFT)과 역 이산 시간 푸리에 변환 (inverse discrete time fourier transform; IDTF)이라고 한다.

4. 전력 스펙트럼과 피리오도그램

신호처리에서는 신호 $x[n]$ 와 켈레복소수 신호 $x^*[n]$ 를 이용하여 자기상관함수 (autocorrelation function)를 (i) 결정적 (deterministic) 신호에 대하여는

$$r_x(k) = \sum_{n=-\infty}^{\infty} x(n+k)x^*(n) = \sum_{n=-\infty}^{\infty} x[n]x^*(n-k)$$

와 같이 (ii) 랜덤 (random) 신호에 대하여는

$$r_x(k) = E[x[n]x^*(n-k)] = E[x(n+k)x^*(n)]$$

로 정의한다. 시계열 분석에서의 자기상관함수는 표준화된 버전이라고 하겠다. 그리고 자기상관함수의 z-변환 혹은 푸리에변환은

$$P_x(z) = \sum_{k=-\infty}^{\infty} r_x(k)z^{-k}$$

또는

$$P_x(e^{j\omega}) = \sum_{k=-\infty}^{\infty} r_x(k)e^{-jk\omega}$$

로 정의된다.

현실적으로 $\{x[n]; n = 0, \dots, N-1\}$ 와 같이 유한개의 관측치가 주어지는 경우가 대부분이고 전력 스펙트럼이 자기상관순열의 푸리에변환으로 정의됨으로 스펙트럼의 추정량을 구하는 자연스런 방법은 자기상관순열 $\{r_x(k); k = 0, \dots, N-1\}$ 를

$$\hat{r}_x(k) = \frac{1}{N} \sum_{i=1}^{N-1} x(n+k)x^*(n); k = 0, \dots, N-1$$

와 같이 정의되는 표본자기상관순열 $\{\hat{r}_x(k); k = 0, \dots, N-1\}$ 로 대체하는 것이다. 전력 스펙트럼 추정량 또는 피리오도그램 (periodogram)은

$$\hat{P}_x(e^{j\omega}) = \sum_{k=-N+1}^{N-1} \hat{r}_x(k)e^{-jk\omega}$$

로 정의된다. k 가 음수인 경우는 켈레복소수의 특징에 따라 $\hat{r}_x(-k) = \hat{r}_x^*(k)$ 로 계산한다.

이제 $x[n]$ 이 오직 $0 \leq n < N$ 에서만 정의된다는 의미에서 $x[n]$ 을 $x_N(n)$ 으로 표기하면 합성곱 (convolution)을 이용하여

$$\hat{r}_x(k) = \frac{1}{N} \sum_{n=-\infty}^{\infty} x_N(n+k)x_N^*(n) = \frac{1}{N} x_N(k) \star x_N^*(k)$$

가 되고 이를 푸리에변환하면

$$\hat{P}_x(e^{j\omega}) = \frac{1}{N} X_N(e^{j\omega})X_N^*(e^{j\omega}) = \frac{1}{N} |X_N(e^{j\omega})|^2$$

가 된다. 여기서,

$$X_N(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x_N(n)e^{-jn\omega} = \sum_{n=0}^{N-1} x[n]e^{-jn\omega}$$

이 됨으로 위의 피리오도그램은 계산상 편리성을 위해 ω 를 $2\pi k/N$ 으로 대체하여

$$\hat{P}_x(e^k) = \frac{1}{N} |X_N(e^{jk})|^2$$

와 같이 사용하곤 한다. 전력스펙트럼과 피리오도그램의 예제를 Figure 8.3에 그려보았다.

5. 자기회귀모형

입력 $x[n]$ 와 출력 $y[n]$ 사이의 관계가

$$\sum_{k=0}^N a_k y[n-k] = \sum_{k=0}^M b_k x[n-k]$$

와 같을 때, 이를 선형 상수 계수 차분 방정식 (linear constant-coefficient difference equation; LCCD equation)이라고 한다. 만일 입력이 $\epsilon[n]$, 출력이 $x[n]$ 이 라면

$$\sum_{k=0}^N a_k x[n-k] = \sum_{k=0}^M b_k \epsilon[n-k]$$

가 되고 이를 시계열 분석에서는 ARMA(N,M)모형이라 부른다. LCCD 방정식에 z -변환을 수행하면

$$\sum_{k=0}^N a_k z^{-k} Y(z) = \left(\sum_{k=0}^N a_k z^{-k} \right) Y(z) = \left(\sum_{k=0}^M b_k z^{-k} \right) X(z) = \sum_{k=0}^M b_k z^{-k} X(z)$$

가 되고 이를 정리하여 $H(z)$ 를

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^M b_k z^{-k}}{\sum_{k=0}^N a_k z^{-k}}$$

와 같이 정의하고 입력-출력 시스템의 변이함수 (transfer function)라고 한다. 실제로는 주파수 (f)에 대하여 $\omega = 2\pi f$ 의 관계가 성립하여 f 따른 변화를 보고 싶을 때가 있기에 z 대신 $e^{j\omega}$ 를 대체하여

$$H(f) = H(\omega) = H(z)|_{z=e^{j\omega}}$$

라 하고 이를 주파수 반응 함수 (frequency response function)라고 부른다. $H(\omega)$ 는 R의 signal 패키지의 `freqz` 함수를 사용하여 구할 수 있다.

입력(x)과 출력(y) 사이에 합성곱 (convolution) $y[n] = h[n] * x[n]$ 의 관계가 성립한다면,

$$P_y(z) = P_x(z)H(z)H^*(1/z^*)$$

또는

$$P_y(e^{j\omega}) = P_x(e^{j\omega})|H(e^{j\omega})|^2$$

가 된다.

예를 들어, AR(1), MA(1)과 ARMA(1,1)의 전력 스펙트럼은

$$P_x(e^{j\omega}) = \sigma^2 \left| \frac{b_0}{1 - a_1 e^{-j\omega}} \right|^2, \quad P_x(e^{j\omega}) = \sigma^2 |b_0 - b_1 e^{-j\omega}|^2,$$

$$P_x(e^{j\omega}) = \sigma^2 \left| \frac{b_0 - b_1 e^{-j\omega}}{1 - a_1 e^{-j\omega}} \right|^2$$

가 되고 시계열 분석에서는 일반적으로 b_0 를 1로 놓는다. R에서 제공하는 `arma.spec(ar=a,ma=b)`은 $H(j\omega) = (1 + be^{-j\omega})/(1 - ae^{-j\omega})$ 를 의미하고 사용 시 b 의 부호에 주의를 해야 한다. `spectrum` 명령문은 기본적으로 AR모형을 기반으로 피리오도그램을 출력해 주는데 신경 써야 할 추가 옵션으로 `spans`이 있는데 초기의 평활 기법인 Daniell의 방법을 사용하기 위한 구간을 의미한다. 디폴트는 Bloomfield (1976)의 공식을 사용한다고 나와 있다. 몇 가지 모델의 스펙트럼을 Figure 8.4와 8.5에 그려 보았다.

6. FIR 필터와 IIR 필터

FIR는 finite impulse response의 약자이고 IIR은 infinite impulse response의 약자이다. 먼저 FIR은

$$y[n] = \sum_{k=0}^M a_k x[n-k]$$

와 같이 유한개의 입력 항에 의해 출력이 결정되는 관계를 FIR 시스템 혹은 FIR 필터라고 한다. 예를 들어

$$y[n] = \frac{1}{3}x[n] + \frac{1}{3}x[n-1] + \frac{1}{3}x[n-2]$$

와 같이 표현되는 이동평균이 대표적인 예라고 하겠다. 한편,

$$y[n] = \sum_{k=0}^{\infty} a_k x[n-k] \text{ or } \sum_{k=-\infty}^{\infty} a_k x[n-k]$$

와 같이 무한개의 입력 항으로 이루어진 출력과의 관계를 IIR시스템 혹은 IIR 필터라고 한다. 엄밀하게 말하면 a_k 를 $h[k]$ 로 대체하여 표현한 것을 FIR 또는 IIR 필터라고 한다.

6.1. FIR 필터

아래와 같이 정의된 신호를 단위 자극 (unit impulse) 혹은 (Kronecker) delta function이라고 한다.

$$\delta[n] = \begin{cases} 1 & n = 0, \\ 0 & n \neq 0. \end{cases}$$

FIR 필터에서 $x[n] = \delta[n]$ 로 하면 $y[n] = \sum_{k=0}^M h[k]\delta[n-k] = h[n]$ 가 된다. 이때 $h[n]$ 를 단위 자극 함수 (unit impulse response) 혹은 자극 함수 (impulse response)라고 부른다. FIR 필터는 사실상 $h[n]$ 와 $x[n]$ 의 (유한) 합성곱 (convolution)

$$y[n] = \sum_{k=0}^M h[k]x[n-k] = h[n] * x[n]$$

으로 표시할 수 있고 이것을 z -변환을 시키면 $h[n]$ 와 $x[n]$ 의 z -변환의 곱

$$Y(z) = H(z)X(z)$$

이 된다. (자세한 과정은 신호처리 전문서를 참고) 예를 들어 이동평균 $y[n] = (x[n] + x[n-1] + x[n-2])/3$ 의 경우 $h[0] = h[1] = h[2] = 1/3$ 이고 달리 쓰면

$$h[n] = \frac{1}{3}\delta[n] + \frac{1}{3}\delta[n-1] + \frac{1}{3}\delta[n-2]$$

이다. 따라서

$$H(z) = \frac{1}{3}z^0 + \frac{1}{3}z^{-1} + \frac{1}{3}z^{-2} = \frac{z^2 + z^1 + 1}{3z^2}$$

가 된다.

6.2. IIR 필터

예를 들어,

$$h[n] = \delta[n] + \frac{1}{3}\delta[n-1] + \left(\frac{1}{3}\right)^2 \delta[n-2] \dots = \sum_{k=0}^{\infty} \left(\frac{1}{3}\right)^k \delta[n-k]$$

처럼 항이 무한 개인 자극함수로 이루어진 입력/출력 시스템을 IIR 시스템 혹은 IIR 필터라고 한다. 역시 또한 $h[n]$ 은 z -변환을 통해 간단하게

$$H(z) = \sum_{k=0}^{\infty} \left(\frac{1}{3}\right)^k z^{-k} = \frac{1}{1 - \frac{1}{3}z^{-1}}$$

로 표기가 가능하다.

6.3. 유리함수 (rational function)

FIR과 IIR을 표현하는 분수형태의 반응함수를 차수가 (p, q) 인 유리함수라고 한다.

$$H(z) = \frac{b_0 + b_1z^{-1} + \dots + b_qz^{-q}}{1 + a_1z^{-1} + \dots + a_pz^{-p}} = \frac{\sum_{k=0}^q b_kz^{-k}}{1 + \sum_{k=1}^p a_kz^{-k}}$$

앞서 본 FIR예제는 $p = 0$ 이고 $q = 3$, IIR예제는 $p = 1$ 이고 $q = 0$ 인 경우가 되겠다.

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^q b_kz^{-k}}{1 + \sum_{k=1}^p a_kz^{-k}}$$

를 역변환하면

$$y[n] + a_1y[n-1] + \dots + a_py[n-p] = b_0x[n] + \dots + b_qx[n-q]$$

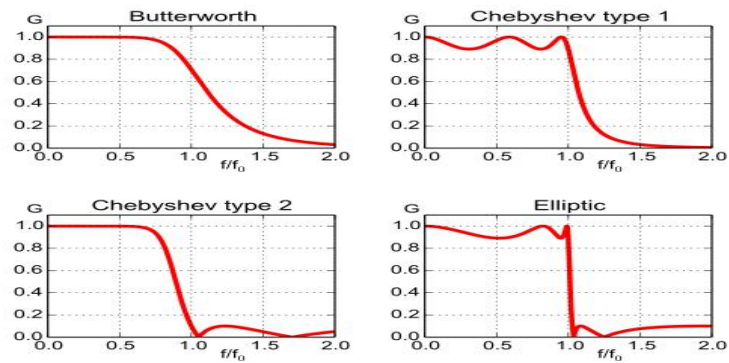
가 된다. $y[]$ 대신 $x[]$ 혹은 $z[]$ 를 넣고 $x[]$ 대신 $\epsilon[]$ 을 넣으면 통계의 시계열의 ARMA(p,q)모형이 된다. 이 점이 신호처리와 시계열의 연결점이 된다고 하겠다.

Figure 8.6과 8.7에 FIR: $H(z) = (z^0 + z^{-1} + z^{-2} + z^{-3} + z^{-4})/5$ 와 IIR: $H(z) = 1/(1 - z^{-1}/3)$ 와 관련된 함수를 그려 놓았다. 왼편에 있는 그림은 freqz함수의 결과물인데 데시벨 (dB)은 $\text{dB} = 20 \log_{10} |H(e^{j\omega})|$ 를 의미하고 수직축의 단위로 사용한다. 이름이 'Phase'라고 달려 있는 그림은 복소수 평면에 그려진 $H(\omega)$ 의 ω 에 따른 각도의 변화를 그려 넣은 것이다. FIR의 예제에서 $H(\omega)$ 가 내부에 꼬임 (spiral)이 존재하고 IIR의 예제에서는 꼬임이 없는데 그 현상이 Phase그림에서 불연속과 연속으로 나타난다. Phase 그림에서 선이 단조증가, 감소의 모습을 보이는데 이것은 공학에서 시간 축에서의 지연 (delay)과 주파수축에서의 지연에 왜곡이 없음을 의미한다. 자세한 공학적 설명은 유보하겠다.

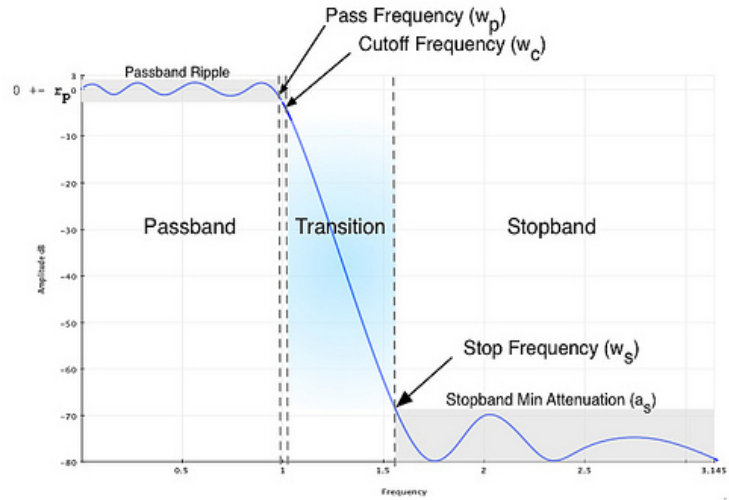
7. 필터 디자인

이제 마지막으로 원하는 필터를 만드는 프로그래밍을 해보겠다. 필터 디자인은 간단하게 몇 줄로 쓸 수 있는 내용이 아니며 본 절에서는 R함수를 소개하는 것으로 같음하려한다. 필터는 크게 저역 통과 필터 (low pass filter), 고역 통과 필터 (high pass filter) 그리고 대역 통과 필터 (band pass filter) 등이 있다. 예를 들어 이동평균같이 낮은 주파수 구간을 살리고 변화가 심한 높은 주파수 구간을 걸러내는 필터를 저역 통과 필터라고 한다. 따라서 주어진 시계열에 이동평균을 적용하면 급격한 변화를 감쇄시킨 전체적 흐름을 파악할 수 있는 시계열로 변형을 시킬 수 있다.

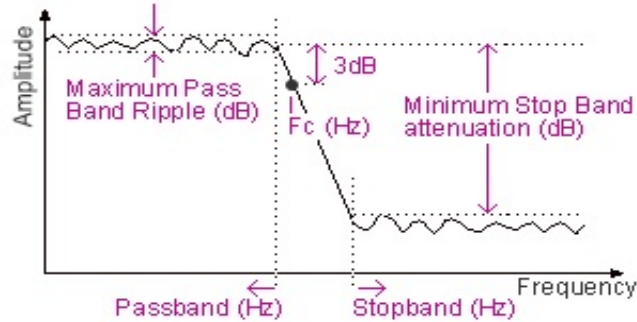
많은 공학자들과 수학자들이 대표적인 필터들을 미리 만들어 놓았고 필요에 따라 데이터에 맞게 변형하여 사용할 수 있다. 예를 들어 Figure 7.1에 대표적인 저역 통과 필터들의 그림을 첨부하였고 Figure



(a) representative low pass filters (source: wikipedia)



(b) low pass filter specification (source: www.dsprelated.com)



(c) low pass filter specification (source: www.gwinst.com)

Figure 7.1 filter examples and specification

8.8에 그들 중 Elliptic 필터를 디자인해보았다. 먼저 Figure 7.1의 (b)와 (c)를 보면 필터의 모양새를 규정하는 파라미터가 (W_p, W_s, R_p, A_s)로 네 개가 주어지는데 W_p 는 통과 대역의 끝 점, W_s 는 저지 대역의 시작 점, R_p 는 통과 대역의 최대폭 (maximum pass band ripple) 그리고 A_s 혹은 R_s 는 저지 대역의 최소 감쇠폭 (minimum stop band attenuation)을 의미한다. 예컨대 (0.5, 0.6, 0.5, 29)가 되도록 하여 `ellipord`를 수행하면 유리함수의 차수로서 $k = 5$ 가 적당하고 $W_c = 0.5$ 라고 출력해 준다. 이들을 이용하여 `ellip`를 수행하면 영점과 극점 그림 (Figure 8.7)을 기반으로 $H(z)$ 의 $Y(z)$ 와 $X(z)$ 의 다항식 계수 (b_k, a_k)를 출력해준다. 이 경우 5개의 a 계수와 4개의 b 계수를 출력해 주고 이를 이용하여

$$y[n] = .4925y[n - 1] + \dots + 0.1449y[n - 5] + 0.1440x[n] + \dots + 0.1440x[n - 5]$$

라는 필터/시스템을 확정할 수 있다. 이를 이용하여 앞서 Figure 8.2와 동일한 오차가 포함된 데이터를 필터링 한 결과를 본래의 신호와 함께 Figure 8.8에 그려 넣었다.

8. R-코드와 그림들

사용하게 될 R-패키지는 `signal` (signal developers, 2013)과 `TSA` (Chan과 Ripley, 2012)이다. `signal`은 신호처리에 특화된 패키지로서 `fft` (푸리에 변환), `ifft` (푸리에 역변환), `freqz` (ARMA 모형의 주파수 반응함수), `fftfilt` (이동평균 필터), `zplane` (극점-영점 그림) 등을 포함하고 있다. `TSA`는 시계열 분석을 위한 패키지로서 `ARIMA` (ARIMA 모형추정) 그리고 `spec`, `periodogram` (스펙트럼, 피리오도그램 추정) 등을 포함하고 있다.

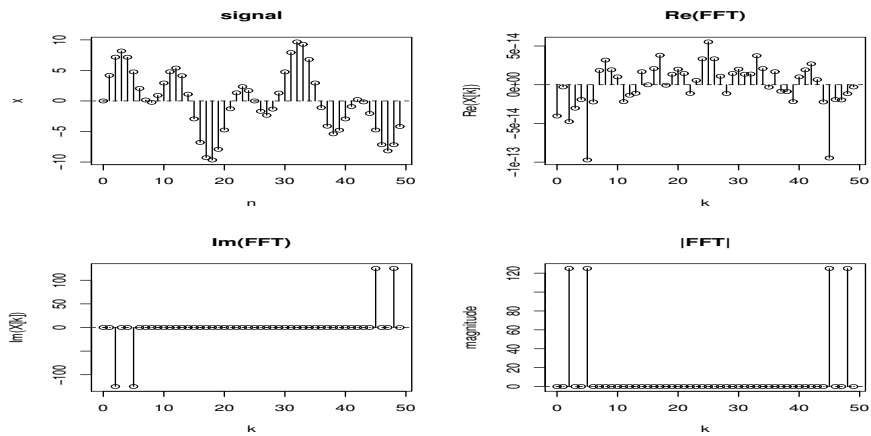


Figure 8.1 DFT example

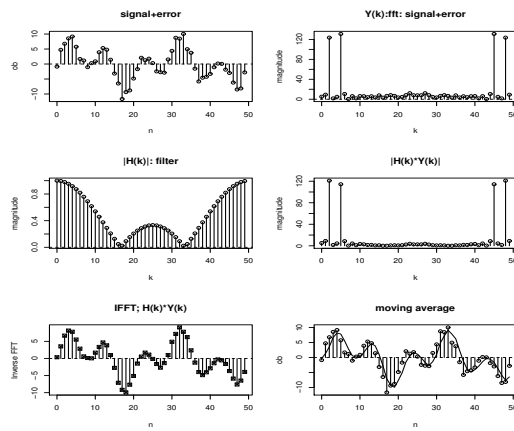


Figure 8.2 DFT filtering example

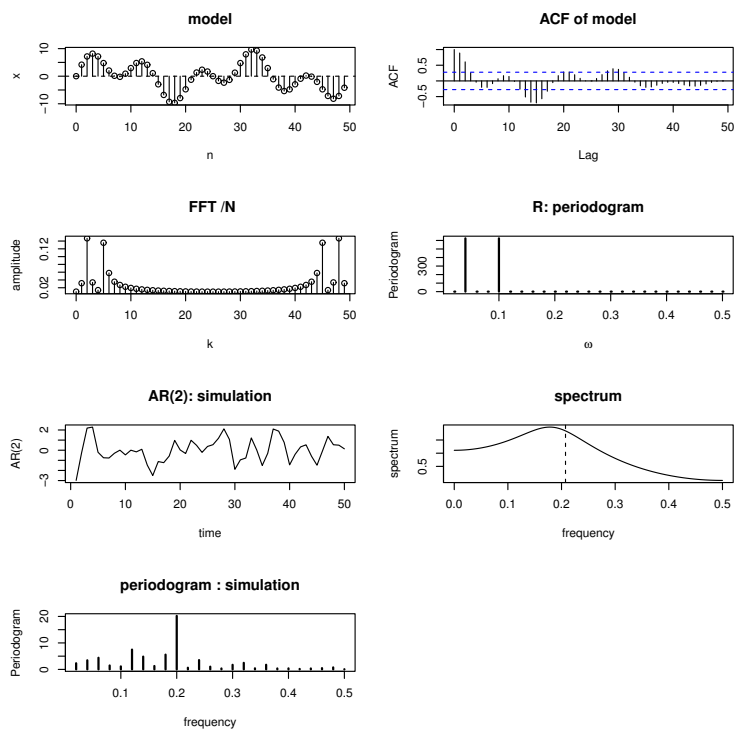


Figure 8.3 Periodogram example

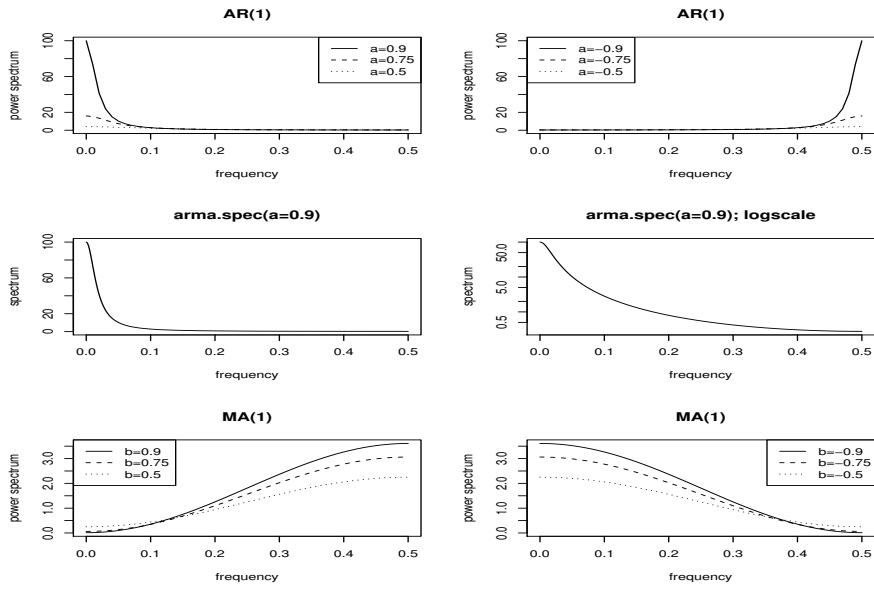


Figure 8.4 Parametric modeling example

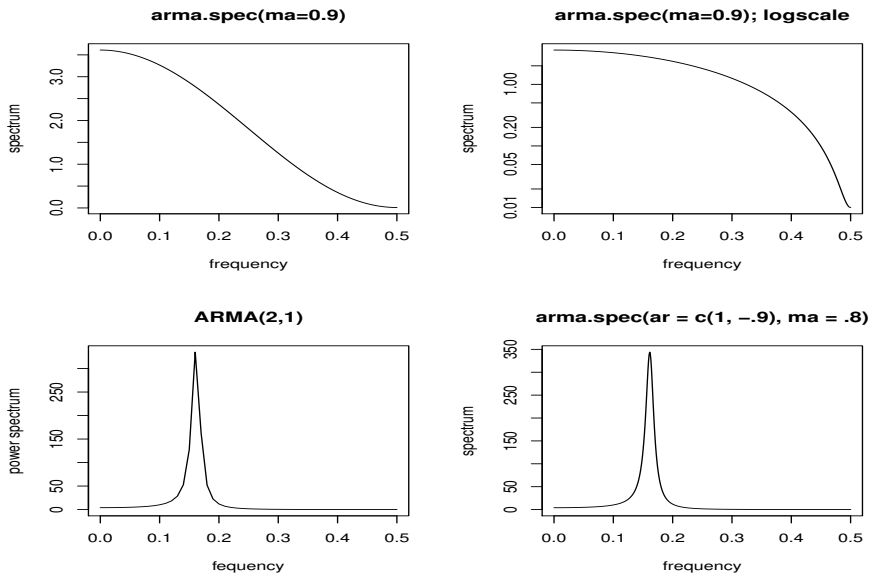


Figure 8.5 Parametric modeling example (continued)

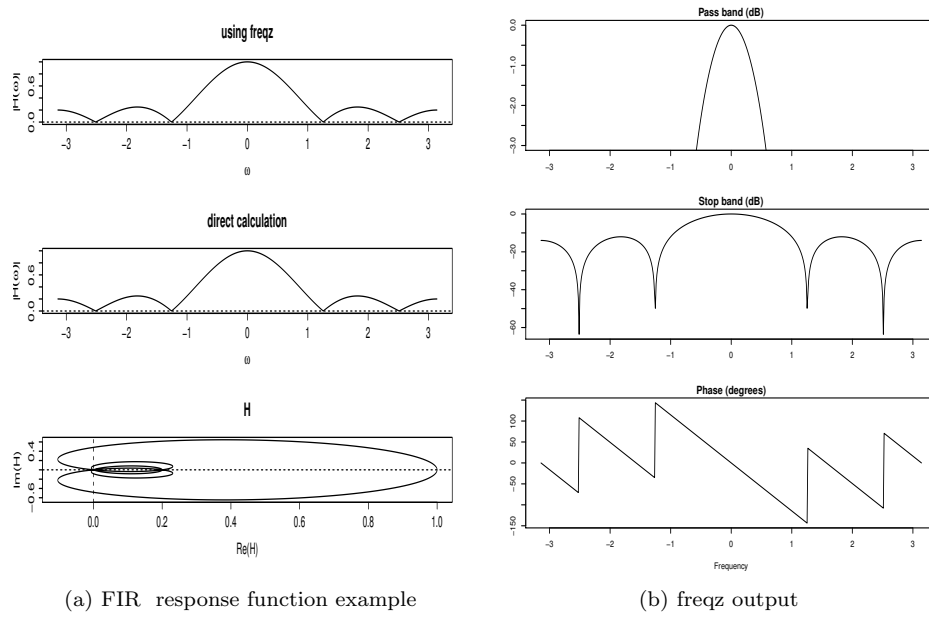


Figure 8.6 FIR example

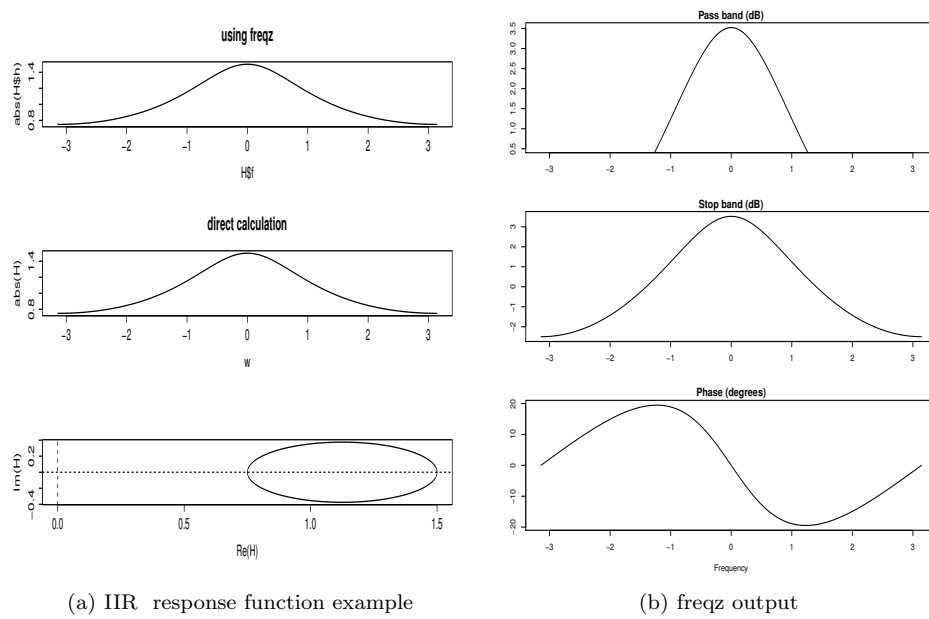


Figure 8.7 IIR example

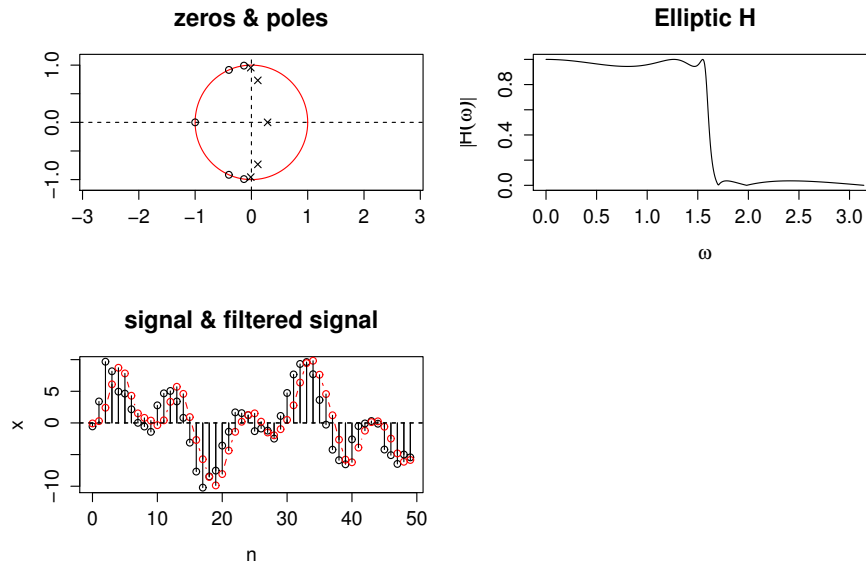


Figure 8.8 Filter design example

Figure 8.1: 제2장 푸리에 변환

```

library(signal)
n <- seq(0, 49, by=1)
x <- 5*sin(2*pi*n/10)+5*sin(2*pi*n/25)
par(mfrow=c(2,2))
plot(n,x,main="signal") #original signal
segments(0:49, 0, 0:49, x)
abline(h=0,lty=2)
fx <- fft(x) # fast fourier transform
k <- n plot(k,Re(fx), xlab="k",ylab="Re(X[k])",main="Re(FFT)")
segments(0:49, 0, 0:49, Re(fx))
abline(h=0,lty=2)
plot(k,Im(fx), xlab="k",ylab="Im(X[k])",main="Im(FFT)")
segments(0:49, 0, 0:49, Im(fx))
abline(h=0,lty=2)
mag <- Mod(fx) # abs(fx)
plot(k,mag,xlab="k", ylab="magnitude",main="| FFT |")
segments(0:49, 0, 0:49, mag)
abline(h=0,lty=2)

```

Figure 8.2: 제2장 잡음 제거

```

par(mfrow=c(3,2))
e ← rnorm(50)
ob ← x+e # noisy signal
plot(n,ob,main="signal+error")
segments(0:49, 0, 0:49, ob)
abline(h=0,lty=2)
fob ← fft(ob)
mag ←abs(fob)
plot(k,mag, xlab="k",ylab="magnitude",main="Y(k):fft: signal+error")
segments(0:49, 0, 0:49, mag)
abline(h=0,lty=2)
N ← 50;M ←3
mah ← sin(pi*(k/N)*M)/(M*sin(pi*(k/N))) # FFt of moving average filter
mah[1] ← 1
plot(k,abs(mah),ylab="magnitude",main="| H(k)|: moving average filter")
segments(0:49, 0, 0:49, abs(mah))
abline(h=0,lty=2)
plot(k,abs(mah*fob),ylab="magnitude",main="| H(k)*Y(k) |")
segments(0:49, 0, 0:49, abs(mah*fob))
ifmf ← ifft(mah*fob)
plot(n,Re(ifmf),pch=7,ylab="Inverse FFT", main="IFFT; H(k)*Y(k)")
segments(0:49, 0, 0:49,Re(ifmf))
abline(h=0,lty=2)
z ← fftfilt(rep(1,3)/3, ob)
plot(n,ob,main="moving average") # smoothed signal
lines(n,z)
segments(0:49, 0, 0:49, ob)
abline(h=0,lty=2)

```

Figure 8.3: 제4장 스펙트럼, 피리오도그램

```

library(TSA)
n ← seq(0, 49, by=1)
x ← 5*sin(2*pi*n/10)+5*sin(2*pi*n/25)
par(mfrow=c(4,2))
plot(n,x, main="model")
segments(0:49, 0, 0:49, x)
abline(h=0,lty=2)
xacf ←seq(1:50)
xacf ← acf(x, lag.max=50,drop.lag.0 = FALSE,main="ACF of model")
fft(xacf$acf) #fast fourier of ACF
amp ← abs(fft(xacf$acf))/50

```

```

plot(n,amp,ylab="amplitude",xlab="k",main=" FFT /N") #k=2, 5
segments(0:49, 0, 0:49, amp)
abline(h=0,lty=2)
periodogram(x,xlab=expression(omega),main="R: periodogram")
#AR(2) simulation
ar.ts ← arima.sim(n = 50, list(ar = c(0.5, -0.5)))
plot(ar.ts,ylab="AR(2)",xlab="time",main="AR(2): simulation")
spec.ar(ar.ts,main="",xlab="frequency")
# AR spectrum
abline(v=0.207414830,lty=2,main="spectrum : simulation") # maximum at 0.2074
periodogram(ar.ts,xlab="frequency",main="periodogram : simulation") #highest spike at 0.2

```

Figure 8.4-5: 제4, 5장 스펙트럼, AR, MA, ARMA

```

par(mfrow=c(3,2))
#AR
f ← function(x,a){1/(1-(a)*exp(-(0+1i)*x))} # AR(1) H(f)
x ← seq(0,0.5,by=0.01)
plot(x,Mod(f(x*2*pi,0.9))^2,type="l",lty=1,xlab="frequency",ylab="power spectrum",
main="AR(1)")
lines(x,Mod(f(x*2*pi,0.75))^2,lty=2)
lines(x,Mod(f(x*2*pi,0.5))^2,lty=3)
legend("topright",lty=1:3, c("a=0.9", "a=0.75", "a=0.5"))
plot(x,Mod(f(x*2*pi,-0.9))^2,type="l",lty=1,xlab="frequency",ylab="power spectrum",
main="AR(1)")
lines(x,Mod(f(x*2*pi,-0.75))^2,lty=2)
lines(x,Mod(f(x*2*pi,-0.5))^2,lty=3)
legend("topleft",lty=1:3, c("a=-0.9", "a=-0.75", "a=-0.5"))
arma.spec(ar=0.9,main="arma.spec(a=0.9)",log="no") # AR(1) spectrum
arma.spec(ar=0.9, main="arma.spec(a=0.9); logscale")
#MA
f ← function(x,b){(1-b*exp(-(0+1i)*x))} # MA(1) H(f)
x← seq(0,0.5,by=0.01)
plot(x,Mod(f(x*2*pi,0.9))^2,type="l",lty=1,xlab="frequency",ylab="power spectrum",
main="MA(1)")
lines(x,Mod(f(x*2*pi,0.75))^2,lty=2)
lines(x,Mod(f(x*2*pi,0.5))^2,lty=3)
legend("topleft",lty=1:3, c("b=0.9", "b=0.75", "b=0.5"))
plot(x,Mod(f(x*2*pi,-0.9))^2,type="l",lty=1,xlab="frequency",ylab="power spectrum",
main="MA(1)")
lines(x,Mod(f(x*2*pi,-0.75))^2,lty=2)
lines(x,Mod(f(x*2*pi,-0.5))^2,lty=3)
legend("topright",lty=1:3, c("b=-0.9", "b=-0.75", "b=-0.5"))

```

```

par(mfrow=c(2,2))
arma.spec(ma=0.9,main="arma.spec(ma=0.9)", log="no") # MA(1) sepctrum
arma.spec(ma=0.9, main="arma.spec(ma=0.9); logscale")
# ARMA(1,2)
f ← function(x,a,b,c){(1-c*exp(-(0+1i)*x))/(1-a*exp(-(0+1i)*x)-b*exp(-(0+2i)*x))} # ARMA(1,1)
spectrum
x ← seq(0,0.5,by=0.01)
plot(x,Mod(f(x*2*pi,1,-0.9,-0.8))^2,type="l",lty=1,xlab="frequency",ylab="power spectrum".
main="ARMA(2,1)") arma.spec(ar = c(1, -.9), ma = .8, main="arma.spec(ar = c(1, -.9), ma =
.8)", log="no") # ARMA(1,1) spectrum

```

Figure 8.6-(a): 제6장 FIR

```

w ← seq(-pi,pi,length=500)
b ← c(1/5,1/5,1/5,1/5,1/5) #coefficients of numerator
a ← 1 # coefficients of denominator
par(mfrow=c(3,1))
H ← freqz(b,a,w)
# z-transform plot(H$f,abs(H$h),main="using freqz", xlab=expression(omega),
ylab= expression(abs(H(omega))),type="l")
abline(h=0,lty=2)
H ← (1+exp(-1i*w)+exp(-1i*2*w)+exp(-1i*3*w)+exp(-1i*4*w))/5 # Z-transform
plot(w,abs(H),main="direct calculation", xlab=expression(omega),
ylab= expression(abs(H(omega))),type="l")
abline(h=0,lty=2)
plot(H, ,main="H",type="l")
H ← freqz(b,a,w)
plot(H,type="l")
par(mfrow=c(3,1))
b ← c(1)
a ← c(1,-1/3)
H ← freqz(b,a,w)
plot(H$f,abs(H$h),,main="using freqz", xlab=expression(omega),
y lab= expression(abs(H(omega))),type="l")
H ← 1/(1-(1/3)*exp(-1i*w))
plot(w,abs(H),main="direct calculation", xlab=expression(omega),
ylab= expression(abs(H(omega))),type="l")
abline(h=0,lty=2)
plot(H,type="l",,main="H",xlim=c(-.2,1.5))
abline(h=0, v=0, lty=2)

```


Figure 8.6-(b): 제6장 FIR

```
H ← freqz(b,a,w) # z-transform of FIR
plot(H,type="l")
```

Figure 8.7-(a): 제6장 IIR

```
par(mfrow=c(3,1))
b ← c(1) # coefficient of numerator
a ← c(1,-1/3) #coefficient denominator
H ← freqz(b,a,w)
plot(H$f,abs(H$h),type="l",main="using freqz")
H ← 1/(1-(1/3)*exp(-1i*w))
plot(w,abs(H),type="l",main="direct calculation")
abline(h=0,lty=2)
H ← 1/(1-(1/3)*exp(-1i*w))
plot(H,type="l",xlim=c(0,1.5))
abline(h=0,v=0,lty=2)
```

Figure 8.7-(b): 제6장 IIR

```
H ← freqz(b,a,w) # z-transform of IIR
plot(H,type="l")
```

Figure 8.8: 제7장 필터 디자인

```
par(mfrow=c(2,2))
ellipord(0.5, 0.6, 0.5, 29) # Wp, Ws, Rp, Rs
coeff ← ellip(5,0.5,29,0.5) # n, Rp, Rs, Wc
b ← coeffb
a ← coeffa
zplane(b,a,main="zeros & poles")
abline(h=0,v=0,lty=2)
w ← seq(0,pi,length=500)
H ← freqz(b,a,w)
plot(H$f, abs(H$h), main="Elliptic H",
xlab=expression(omega),
ylab=expression(abs(H(omega))), type="l")
n ← seq(0, 49, by=1)
e ← rnorm(50)
x ← 5*sin(2*pi*n/10)+5*sin(2*pi*n/25)+e
plot(n,x,main="signal & filtered signal")
segments(0:49, 0, 0:49, x)
abline(h=0,lty=2)
ellip_filter ← ellip(5,0.5,29,0.5) # ellip filter construction
z ← filter(ellip_filter,x) # ellip filter implementation
```

```
lines(n,z, type="b", col="red",lty=1)
segments(0:49, 0, 0:49, z)
abline(h=0,lty=2)
```

9. 결론

현재 상용프로그램에 의해 구현되고 있는 신호처리 문제를 통계 전문가들에게 익숙한 R을 이용하여 구현할 수 있는 활용서를 써보았다. 이미 많은 연구자들에 의해 R을 활용되고 있으나 흩어져 있는 내용들을 정리하여 보았다. 본 활용서가 보다 심도 깊은 연구를 위한 디딤돌이 되기를 바란다.

References

- Bloomfield, P. (1976). *Fourier analysis of time series: an introduction*, Wiley, New York.
- Ha, E. H. (1999). Simultaneous confidence regions for spatial autoregressive spectral densities. *Journal of the Korean Data and Information Science Society*, **10**, 397-404.
- Hayes, M. H. (1996). *Statistical digital signal processing and modeling*, John Wiley & Sons, Inc., New York.
- Kim, H. J., Kwak, H. R., Kim, Y. N. and Choi, Y. J. (2015). A spectrum based evaluation algorithm for micro scale weather analysis module with application to time series cluster analysis. *Journal of the Korean Data and Information Science Society*, **26**, 41-53.
- Chan, K.-S. and Ripley, B. (2012). TSA: Time Series Analysis, R package version 1.01, <https://CRAN.R-project.org/package=TSA>.
- Lee, C. H. (2013). *Digital signal processing*, Hanbit media, Seoul.
- Oppenheim, A. V. and Schaffer, R. W. (1999). *Discrete-time signal processing*, 2nd Ed., Prentice Hall, New York.
- Priestley, M. B. (1981). *Spectral analysis and time series*, Academic Press, San Diego.
- Proakis, J. G. and Manolakis, D. G (1996). *Introduction to digital signal processing*, 3rd Ed., Prentice Hall, New York.
- Signal developers (2013). signal: signal processing, R package version 07-6, <http://r-forge.r-project.org/projects/signal/>.
- <http://www.dsprelated.com>.
- <http://www.gwinst.com>.
- <http://www.wikipedia.com>.

The practical guide for using the R-package in the digital signal processing [†]

Ro Jin Pak ¹

¹Department of Applied Statistics, Dankook University

Received 13 July 2017, revised 5 September 2017, accepted 7 September 2017

Abstract

The signal processing is a field of the electrical engineering but it is very much related with the time series analysis. These days the commercial softwares are widely used by the researchers. We have attempted to make a guide for using the R-package in the digital signal processing. It would be good to read the materials in each section first and to follow the plots in the section 8 and to run the attached R-codes. The article consists of (1) Fourier transform and Fourier inverse transform, (2) spectral analysis (3) parametric and non-parametric estimation for the period (4) filter design. Simple theoretical explanations are provided and R implementations are added.

Keywords: Digital signal processing, filtering, Fourier transform, R-package, time series.

[†] The present research was conducted by the research fund of Dankook University in 2017.

¹ Professor, Department of Applied Statistics, Dankook University, Jukjun-Dong, Suji-Gu, Yongin 16890, Korea. E-mail: rjpak@dankook.ac.kr