韓國數學教育學會誌 시리즈 E <數學教育 論文集>
https://doi.org/10.7468/jksmee.2023.37.2.277
제 37집 제 2호, 2023. 6. 277-297

J. Korea Soc. Math. Ed. Ser. E:
Communications of Mathematical Education
Vol. 37, No. 2, Jun. 2023. 277-297

인공지능에 활용되는 공학수학 합성곱(convolution) 교수・학습자료 연구

이 상 구 (성균관대학교, 교수)
남　　윤 (성균관대학교, 연구원)
이 재 화 (성균관대학교, 연구원)†
김 응 기 (성균관대학교, 초빙교수)

합성곱(convolution)은 인공지능(artificial intelligence)에서 컴퓨터 비전(computer vision), 심층학습(deep learning) 등의 분야를 이해하고 응용하려면 알아야 하는 중요한 수학적 연산이다. 그러나 현재의 공학수학 교과과정의 합성곱 내용은 독립적인 주제가 아니라 단편적으로 다루어지고 있어서 그 의미를 충분히 전달하지 못하고 있다. 이에 본 논문에서는 공학수학에서 인공지능 교육과 연계할 수 있도록 개발한 합성곱 교수·학습 자료를 제시한다. 먼저 기존 공학과 인공지능 기술의 통합적 관점에서 합성곱에 대한 배경지식과 응용 사례를 정리하고, 코딩을 이용한 교육이 가능하도록 파이썬(Python)/SageMath 코드를 개발하여 제공한다. 또한 합성곱 지식이 인공지능에서 어떻게 활용되는지 보여주는 구체적인 예시로, 이미지 분류에 사용되는 합성곱신경망(Convolutional Neural Network, CNN)을 개발된 코드와 함께 제공한다. 본 교수·학습자료는 합성곱 개념을 쉽고 효과적으로 교육할 수 있도록 공학수학의 보충 자료로 활용가능하며, 학습자는 코딩을 통해 합성곱을 배우고 본인의 전공과 관련된 인공지능 기술을 학습하는 데 이를 이용할 수 있다.

Ⅰ. 서론

공학수학(engineering mathematics)은 공학 분야에 필요한 수학적 지식을 제공하는 과목이다. 공학의 범위가 세분화되고 기술이 발전함에 따라, 공학의 각 전공마다 다뤄야 하는 수학의 내용 및 범위도 달라지고 있다. 특히 산업 전 분야에 영향을 미칠 수 있는 인공지능(artificial intelligence) 기술의 발전은 공학수학의 교수·학습 방법은 물론, 교과 과정에 대한 변화도 촉진하고 있으며, 과거 전통적인 공학도 이와 같은 기술적 맥락에서 새롭게 조명할 필요가 있다. 또한 인공지능은 소프트웨어 기술과의 결합이 필수적이므로, 공학수학 교수·학습의 관점에서 볼 때 컴퓨터와 코딩(coding)을 이용한 교육이 반드시 동반되어야 한다고 생각한다.

공학의 각 전공에 맞추어 공학수학 교육과정을 개편하려는 연구는 꾸준히 진행되어왔다. 서종진 외(2007)는 생명·나노 관련분야의 수학 내용을 조사하여 전공에 적합한 교수·학습자료를 개발하였고, 이승우(2008)는 전기전자공학 학부과정의 세부전공트랙을 작성하여 전공교과목에서 필요한 수학·통계의 내용을 분석하였다. 김성옥 외(2009)는 공학의 여러 전공 과정을 분석하고, 공학의 기초과정으로서의 수학 교과목의 체계를 구축하는 과정에서, 필요한 내용을 충실하게 포함하면서 독립성을 지닐 수 있도록 모듈화를 시도하였다. 정수연·송영무(2011)는 전자공학과 학생들을 대상으로 전공 교과 내용의 연관성과 수학 교과 내의 선수학습내용과의 연관성을 이용하

* 접수일(2023년 4월 17일), 심사(수정)일(2023년 5월 29일), 게재확정일(2023년 6월 7일)
* MSC2000분류 : 97U50, 97U70
* 주제어 : 공학수학, 합성곱, 라플라스 변환, 인공지능
† 교신저자 : jhlee2chn@skku.edu
* 이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No.2021R1F1A1046714).

여 공학수학 교수·학습자료를 개발하였다. 강주석·박찬일(2017)은 기계공학의 학문적 변화에 맞추어, 각 분야별 전문가를 대상으로 설문 조사를 실시하고, 수학 교과 강의계획서와 비교 분석하여 수학 교과 과정의 개편 방향을 제시하였다.

그러나 이러한 선행 연구들은 다음과 같은 한계를 지니고 있다. 첫째, 기존의 공학수학 교과과정의 분류 체계 내에서 연구를 진행하다 보니, 인공지능과 같은 기술의 발전에 따라 새롭게 중요성이 강조되는 개념이나 주제임에도 독립적으로 발전하지 못하고 여전히 비중이 작거나 생략되는 내용이 존재한다. 둘째, 전문화·특성화에 치중하다 보니 전공별로 수학 교과 과정을 지나치게 세분화하여 교과 내용이 통합적이지 않다. 따라서 에드워드 윌슨(2005)이 주장한 전공 간의 융합과 통섭의 능력을 기르는 데 장애가 될 수 있을 뿐만 아니라 자칫 수학에 대한 편협한 생각을 심어 줄 수도 있다.

그러나 전문화·특성화와 융합·통섭 사이에서 적절한 균형을 맞추기란 쉽지 않다. 전재복(2008)은 전통적인 수학교육 시스템으로는 전문화·특성화를 제대로 수용하기 어려우므로 이를 대체할 수 있는 새로운 수학교육 시스템이 필요하다고 주장하기도 하였다. 공학수학 교육과정을 성공적으로 개편하기 위해서는 공학수학에서 등장하는 각각의 주제나 개념에 대한 기존의 교과 체계를 재검토하고, 새롭게 구성한 내용을 전개해 나가야 할 것이다. 하지만 공학수학이 선형대수학, 미분방정식, 복소해석 등 수학의 광범위한 내용을 포함하고 있는 만큼 한꺼번에 모든 주제를 재검토하고 분석하는 것은 현실적으로 불가능하다.

이에 본 논문에서는 ① 공학수학에서 전통적으로 중요한 주제이면서도 단편적으로 제시되어왔고, ② 최근 인공지능에서의 응용으로 새롭게 중요성이 부각된 개념인 합성곱(convolution)을 예시로, 합성곱 관련 배경지식과 응용 사례를 정리하였다. 특히 기존의 공학과 인공지능이라는 새로운 기술에 대한 통합적 관점을 바탕으로, 공학수학 교과 체계 내에 합성곱의 중요성에 대한 인식이 부족함을 확인하고, 이를 보완하는 교수·학습자료를 연구개발하였다. 또한 소프트웨어 역량을 증진할 수 있도록 코딩을 이용해 합성곱을 교육할 수 있도록 SageMath와 파이썬(Python) 코드를 포함한 웹 콘텐츠[1]도 함께 개발하였다. 본문에서 자세히 소개한다.

Ⅱ. 연구의 배경

1. 합성곱의 이론적 배경

합성곱은 수학적으로 두 함수를 이용하여 새로운 함수를 생성하는 연산으로 정의된다. 전통적으로 미분방정식, 수치해석 등의 과목에서 중요하게 다루는 주제일 뿐만 아니라, 동역학(dynamics), 전자기학(electro-magnetism) 등에서도 많이 응용된다. 합성곱은 인공지능에서도 중요한 개념으로, 심층학습(deep learning), 컴퓨터 비전(computer vision), 신호 처리(signal processing) 등의 분야를 이해하고 활용하려면 합성곱을 반드시 학습해야 한다. 예를 들어, 행렬합성곱(matrix convolution)은 이미지 처리(image processing)에서 핵(kernel) 또는 필터(filter)를 이용하여 특징을 추출하고 이미지에서의 패턴을 인식하는데 사용된다. 게다가 행렬합성곱을 이용한 합성곱신경망(Convolutional Neural Network, CNN) 모형은 인공신경망의 심층학습에서 이미지나 비디오 같은 2차원 데이터를 처리할 수 있으며, 이를 통해 인식, 분류, 객체 검출 등 다양한 작업을 수행한다.

합성곱은 크게 연속함수(continuous function)의 합성곱과 이산함수(discrete function)의 합성곱으로 나눌 수 있다. 먼저 연속함수의 합성곱은 다음과 같이 정의된다.

---

1) http://matrix.skku.ac.kr/SKKU-EM-2020/conv4ai/

정의 1. $\mathbb{R}$에서 정의된 두 함수 f와 g에 대하여 다음의 적분이 존재할 때, 이를 $f * g$로 나타내고 f와 g의 합성곱(convolution)이라 부른다.

$$(f * g)(x) = \int_{-\infty}^{\infty} f(y)\, g(x-y)\, dy$$

연속함수의 합성곱은 라플라스(Laplace) 변환, 푸리에(Fourier) 변환과 밀접한 관계를 가지고 있다. 우선 합성곱에 라플라스 변환을 취하는 경우를 생각해 보자. 라플라스 변환은 $t \geq 0$인 정의역을 갖는 함수에 대해서 정의되는데, $f(t)$와 $g(t)$가 그러한 함수들이라고 하자. 두 함수의 정의역을 실수 전체로 확장하면, 함수 $f(t)$는 $t<0$일 때 $f(t)=0$으로 생각할 수 있고, 새로운 변수 τ에 대해 $g(t-\tau)$을 생각하면 $t-\tau<0$ 경우, 즉 $t<\tau$ 일 때 $g(t-\tau)=0$으로 생각할 수 있다. 따라서 다음 식

$$\int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau = \int_{-\infty}^{0} f(\tau)\, g(t-\tau)\, d\tau + \int_{0}^{t} f(\tau)\, g(t-\tau)\, d\tau + \int_{t}^{\infty} f(\tau)\, g(t-\tau)\, d\tau$$

에서 $\int_{-\infty}^{0} f(\tau)\, g(t-\tau)\, d\tau$와 $\int_{t}^{\infty} f(\tau)\, g(t-\tau)\, d\tau$은 각각 0이 된다. 즉 라플라스 변환과 관계된 함수들의 합성곱은 다음과 같이 정의할 수 있다.

정의 2. $t \geq 0$에서 정의된 두 함수 f와 g에 대해 다음의 적분이 존재할 때, 합성곱 $f * g$는 다음과 같다.

$$(f * g)(t) = \int_{0}^{t} f(\tau)\, g(t-\tau)\, d\tau$$

합성곱의 라플라스 변환에 관해서는 다음의 정리가 알려져 있다.

정리 1. $\mathbb{R}$에서 정의된 두 함수 f와 g가 각각 $F = \mathcal{L}(f)$, $G = \mathcal{L}(g)$라는 라플라스 변환을 가질 때 다음을 만족한다.

$$\mathcal{L}(f * g) = FG$$

그리고 함수 f의 푸리에 변환을 $\boldsymbol{F}(f) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x)\, e^{-i\omega x}\, dx$으로 정의[2]하면, 다음과 같이 푸리에 변환에서도 합성곱에 대한 라플라스 변환과 유사한 결과를 얻을 수 있다. 정리 1과 정리 2의 증명은 Kreyszig(2022)에서 확인할 수 있다.

정리 2. $\mathbb{R}$에서 정의된 두 함수 f와 g가 구분적 연속(piecewise continuous)이고, 유계이며, $|f|$와 $|g|$가 적분가능하면 다음을 만족한다.

$$\boldsymbol{F}(f * g) = \sqrt{2\pi}\, \boldsymbol{F}(f)\, \boldsymbol{F}(g)$$

여기서 $\boldsymbol{F}$는 푸리에 변환이다.

이산함수의 합성곱은 다음과 같이 정의된다.

---

[2] 간혹 푸리에 변환을 적분식 앞의 상수 $1/\sqrt{2\pi}$를 생략하고 정의하는 경우도 있다. 그러나 이런 경우 상수만큼 차이가 날 뿐 본질적인 내용은 다르지 않다.

정의 3. 두 이산함수 f와 g가 정수 $n = \cdots -3, -2, -1, 0, 1, 2, 3 \cdots$ 대해 다음의 합이 존재할 때, 이를 f와 g의 합성곱으로 정의한다.

$$(f * g)(n) = \sum_{k=-\infty}^{\infty} f(k)g(n-k)$$

특히 함숫값이 유한개이면, 위의 합은 유한합이 된다. 합성곱 연산에서 기억해야 할 사실은 연속함수의 합성곱이든 이산함수의 합성곱이든 모두 교환법칙, 분배법칙, 결합법칙을 만족한다는 점이다.

$$f * g = g * f,\ f * (g + h) = f * g + f * h,$$
$$(f * g) * h = f * (g * h),\ f * 0 = 0 * f = 0$$

그리고 어떠한 상수 c에 대해서도 $c(f * g) = (cf) * g = f * (cg)$를 만족한다. 이를 분배법칙과 결합하여 생각하면 합성곱 연산은 선형성(linearity)이 있다고 말할 수 있다.

2. 연구방법 및 절차

본 연구는 다음과 같은 방법으로 수행되었다.

① 기존 공학수학 교재 중 선택된 해외 교재 4권을 조사하여 합성곱 부분을 파악한다.

② 인공지능에 응용하기 위한 내용을 포함하여 일관되게 설명이 가능하도록 하나의 주제로 정리한다.

③ 합성곱을 응용할 수 있는 적절한 예제를 선정하여, 코딩으로 쉽게 학습할 수 있도록 콘텐츠를 개발한다.

이때 코딩 콘텐츠는 공학수학 교수·학습의 관점에서 쉽게 활용할 수 있도록 소프트웨어 구매나 설치가 필요 없는 SageMath와 파이썬(Python)을 사용하여 개발하였다. 본 연구진은 이미 SageMath를 이용하여 공학수학 전 범위에 걸쳐 실습할 수 있는 '공학수학 & 코딩' 콘텐츠를 개발한 경험이 있다(이상구 외, 2016;이재화 외, 2022). 물론 김성옥 외(2009)의 연구와 같이 MATLAB과 같은 상업용 수학 소프트웨어를 공학수학 교육에 접목한 시도가 있으나 비용, 프로그래밍 언어, 설치 등 여러 단점이 있다.

그리고 구체적인 교수·학습 자료는 다음 사항에 중점을 두고 개발하였다.

① 합성곱이 자연과학이나 공학에서 응용되는 이유를 쉽게 파악할 수 있도록 연속함수에 관한 합성곱에 대하여 직관적 의미를 소개하였다. 그리고 연속적 구간에서 정의된 함수와 비슷하게 정수를 정의역으로 하는 이산함수의 합성곱을 소개하였다. 특히 수학적 모델링의 관점에서 학습자가 이산함수의 합성곱과 연속함수의 합성곱의 긴밀한 관계를 이해하도록 하였다.

② 라플라스 변환과 관련하여 합성곱이 선형미분방정식과 선형연립미분방정식에 응용되는 과정을 자세히 설명하였다. 특히 선형연립미분방정식의 경우, 라플라스 변환을 이용한 풀이과정을 통해 해의 공식을 연속함수 합성곱으로 간단히 표현할 수 있음을 보였다. 이때 필요한 행렬지수(matrix exponential) 개념은 SageMath 코드를 제공하여, 수학적 엄밀성을 완화하고도 쉽게 접근할 수 있도록 구성하였다.

③ 이산함수 합성곱은 정의를 확장하여, 인공지능 알고리즘에서 이미지 데이터를 처리하는데 유용한 도구인 행렬합성곱을 소개하였다. 또한 두 행렬을 입력하여 행렬합성곱을 계산하는 코드와 MNIST 손글씨의 이미지 파일을 변환할 수 있는 파이썬(Python) 코드를 개발하여 제시하였다. 그리고 CNN 알고리즘을 간단하게 소개함으로써 행렬합성곱이 어떻게 인공지능에서 사용되는지 설명하였다.

Ⅲ. 연구 결과 및 논의

이 장에서는 합성곱의 직관적인 의미와 라플라스 변환과 선형미분방정식에서의 응용, 행렬합성곱과 인공지능에서의 응용에 관하여 설명한다.

1. 기존 교재의 합성곱 내용 조사

본 연구진은 먼저 국내에 번역되어 사용 중인 공학수학 교재의 원서 4권을 선택하여 합성곱 내용이 어떻게 서술되어 있는지를 조사하였다. 대상으로 삼은 교재는 Kreyszig(2022), Zill(2020), O'Neil(2017), Croft et. al.(2017)으로, 이들을 선택한 이유는 4권 모두 세계적으로 널리 통용되어 공학수학 교과 체계에 대한 국제적 표준을 알 수 있을 뿐만 아니라, 한국어로도 번역되어 국내에서 많이 사용되기 때문이다.

연속함수 합성곱과 이산함수 합성곱 두 가지 주제로 나누어 조사한 결과, Kreyszig(2022), Zill(2020), O'Neil(2017)의 교재에서는 이산함수의 합성곱을 전혀 다루지 않고, 오직 Croft et. al.(2017)에서만 24.15절에서 이산함수 합성곱을 교차상관(cross-correlation)과 함께 소개하고 있다. 이는 부제인 'A Foundation for Electronic, Electrical, Communications and Systems Engineers'에서 알 수 있듯이 해당 교재가 전자, 전기, 통신, 시스템 공학에 특화되었기 때문에 관련 내용을 포함한 것으로 여겨진다.

연속함수 합성곱의 경우, 모든 교재가 2개의 절에 걸쳐서 설명하였는데, 라플라스 변환, 푸리에 변환과 관련지어 소개하고 있다. 각 교재에서 합성곱이 등장하는 절(section)을 간단히 언급하면, 라플라스 변환 관련 내용은 Kreyszig(2022)은 6.5절, Zill(2020)은 4.4의 하위절로 4.4.2절, O'Neil(2017)은 3.4절, Croft et. al.(2017)은 21.9절에서 다루었다. 그리고 푸리에 변환 관련 내용은 Kreyszig(2022)에서 11.9절, O'Neil(2017)에서 14.3절, Croft et. al.(2017)에서 24.8절에서 등장하고, Zill(2020)에서는 15.4절의 연습문제 중 한 문제로 제시하였다.

그러나 모든 교재가 라플라스 변환과 푸리에 변환에서 합성곱을 각기 다른 방식으로 소개하였다. 즉 Ⅱ장에서 이미 살펴보았듯이, 라플라스 변환에서 합성곱을 정의할 때는 적분식의 구간이 $[0,t]$인 반면, 푸리에 변환에서는 실수 전체인 $(-\infty,\infty)$로 표시한 것이다. 하지만 그 어떤 교재도 합성곱을 정의하는 두 가지 적분식의 관계를 명확하게 설명하지는 않아, 초심자가 라플라스 변환의 합성곱 정의식과 푸리에 변환의 합성곱 정의식이 실제 같은 개념이라는 사실을 알기 어렵다.

게다가 Croft et. al.(2017)을 제외하고 합성곱을 계산할 때, 두 함수의 특성이 적분식 연산을 통해 어떻게 반영되는지에 관하여 자세히 설명하지 않았다. 그리고 그래프를 이용하여 합성곱 연산 과정을 설명하는 시각적 자료도 Croft et. al.(2017)만 제공하였다. 반면 Kreyszig(2022), Zill(2020), O'Neil(2017) 3권의 교재는 주로 합성곱의 계산적 기능만을 설명하는데 그치고 있다. 특히 Zill(2020)의 경우, 라플라스 변환의 여러 계산 방법을 소개하는 4.4절 Additional Operational Properties의 하위절인 4.4.2 Transforms of Integrals에서 합성곱을 처음 소개하고 있다. 이 때문에 해당 절의 제목만 봐서는 합성곱이 단순히 적분 계산만을 위한 도구에 불과하다고 인식하기 쉽다. 더군다나 푸리에 변환과 관련해서는 아예 해당 절의 본문이 아닌 Exercises에서 잠깐 소개하고 있다.

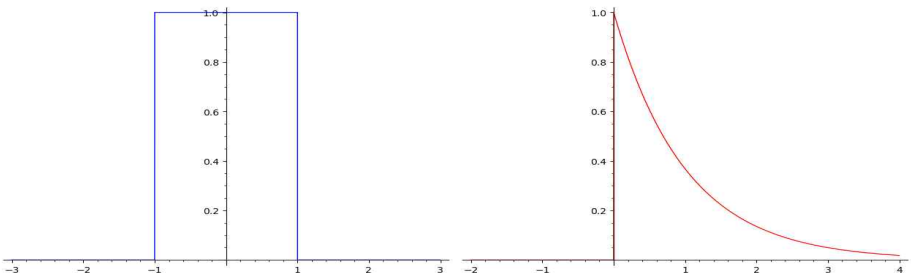
요약하자면 4권의 교재 중 합성곱을 하나의 통합된 개념으로 설명하는 교재는 없었으며, 한 권을 제외하면 이산함수 합성곱은 소개조차 되지 않았다. 게다가 합성곱을 라플라스 변환과 푸리에 변환의 하위 주제 정도로 다루면서 이를 연결하는 설명이 부실하므로 학습자 스스로 합성곱의 개념을 전반적으로 정리하기는 사실상 힘들다. 한편, 행렬합성곱과 CNN은 4권 중 어떠한 교재도 소개하지 않았다.

2. 합성곱의 직관적 의미

앞서 살펴본 바와 같이, 합성곱을 다루는 주요 교재들은 주로 계산에 치중하여 설명하고 있다. 그러나 합성곱은 계산의 유용성 이상으로 중요한 의미를 담고 있다. 이를 설명하기 위해 다음과 같이 함수 f와 g를 예로 들자.

$$f(x)=\begin{cases}1, & -1\le x\le 1\\ 0, & x<-1, x>1\end{cases},\quad g(x)=\begin{cases}e^{-x} & x\ge 0\\ 0 & x<0\end{cases}$$

즉, f는 $-1\le x\le 1$에서 $f(x)=1$, 그 외 영역에서는 $f(x)=0$인 사각형 형태의 함수이고, g는 $x\ge 0$일 때 $g(x)=e^{-x}$, 그 외 영역에서는 0인 함수이다([그림Ⅲ-1]). 또한 f는 $x=-1,1$에서 불연속이며, g는 $x=0$에서 불연속이다.



[그림 Ⅲ-1] 합성곱 피적분함수들의 그래프(왼쪽: 함수 f, 오른쪽: 함수 g)

$x<-1$의 경우, $f(y)$와 $g(x-y)$의 그래프의 0이 아닌 부분이 서로 겹치지 않으므로 $f(y)g(x-y)=0$이고, 따라서 이 범위에서 합성곱은 $f*g=0$이다. $-1\le x\le 1$의 경우, 구간 $-1\le y\le x$에서는 $f(y)=1$이고 $g(x-y)=e^{-(x-y)}$이므로, 이 범위에서 합성곱은 다음과 같다.

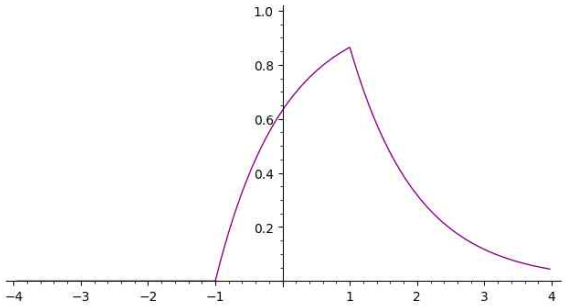
$$(f*g)(x)=\int_{-\infty}^{\infty}f(y)\,g(x-y)\,dy=\int_{-1}^{x}e^{-(x-y)}\,dy=1-e^{-(x+1)}$$

그리고 $x>1$의 경우, 구간 $-1\le y\le 1$에서는 $f(y)g(x-y)=e^{-(x-y)}$이므로 이 범위에서 합성곱의 적분식은 다음과 같다.

$$(f*g)(x)=\int_{-\infty}^{\infty}f(y)\,g(x-y)\,dy=\int_{-1}^{1}e^{-(x-y)}\,dy=e^{-(x-1)}-e^{-(x+1)}$$

이제 모든 결과를 하나로 정리하면, 함수 f와 g의 합성곱은 다음과 같다([그림Ⅲ-2]).

$$(f*g)(x)=\begin{cases}0, & x<-1\\ 1-e^{-(x+1)}, & -1\le x\le 1\\ e^{-(x-1)}-e^{-(x+1)}, & x>1\end{cases}$$



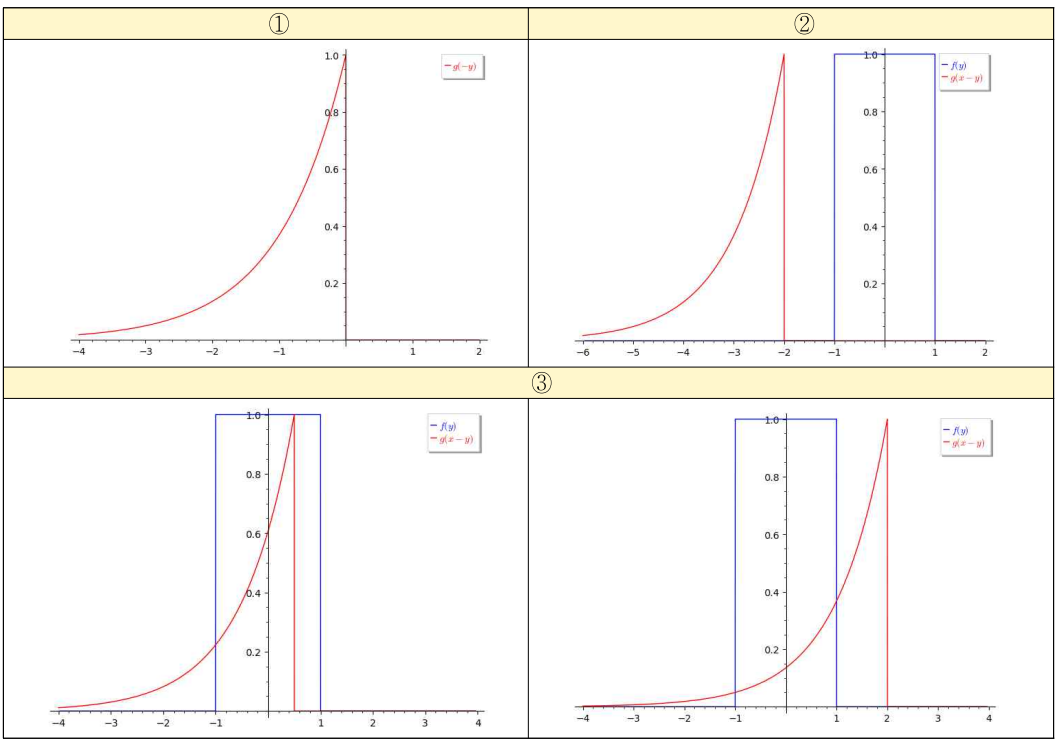
[그림 III-2] 합성곱 $f * g$의 그래프

위의 합성곱은 [그림Ⅲ-3]의 그래프를 통해 다음과 같이 이해할 수 있다.

① $g(y)$를 수직축에 대칭으로 그려서 $g(-y)$의 그래프를 얻는다.

② $g(-y)$를 x만큼 평행이동하여 $g(x-y)$를 얻는다.

③ $g(x-y)$를 $f(y)$와 곱한 뒤 모든 y에 대해 적분한다. 이제 x를 $-\infty$에서부터 ∞까지 움직여가면서 적분값을 살펴본다.



[그림 III-3] 합성곱 연산의 시각화

합성곱의 적분 연산을 통해 마치 함수 g를 함수 f에 대해 문지르는(smearing)듯한 효과가 발생한다. 앞의 예에서 f와 g는 각각 불연속인 점도 포함하고 있었으나 계산한 합성곱은 모든 점에서 연속인 함수가 되었다. 이처럼 합성곱은 두 함수의 중첩을 일으켜 매끄러운 함수를 생성한다.

이산함수는 연속함수를 샘플링(sampling)한 경우로 이해할 수 있다. 즉, 함수 $f(t)$를 구간 Δt 간격으로 샘플링한 수열 $f(0), f(\Delta t), f(2\Delta t), \cdots$로 이해하는 것이다. 예를 들어, 디지털화한 음성 데이터나 이미지 데이터는 원래 음성이 갖는 연속적인 주파수나, 원래 이미지가 나타내는 연속적인 색상을 이산화한 것으로 볼 수 있다. 이러한 접근 방법은 수학적 모델링을 해석하고 응용할 때 유용하다. 그리고 이산함수와 연속함수의 관계를 생각한다면, 이산함수의 합성곱이 연속함수의 합성곱과 거의 같은 성질을 갖는다는 점도 이해하기도 쉬울 것이다. 만약 어떠한 디지털 이미지의 윤곽선을 매끄러운 느낌이 생기도록 변형하고자 싶다면, 연속함수의 경우와 유사하게, 적절한 이산함수와의 합성곱을 통해 생성할 수 있다. 이 점은 뒤에서 행렬합성곱의 예제를 통해 구체적으로 살펴본다.

3. 라플라스 변환과 선형연립미분방정식에서의 응용

가. 선형미분방정식

상수계수를 갖는 1계 비동차 선형미분방정식의 초깃값 문제를 생각해 보자.

$$\frac{dy}{dt} = ay + f(t),\ y(0) = y_0 \qquad \text{- (1)}$$

여기서 $t \geq 0$이다. 그러면 미분방정식 (1)의 해는 다음과 같음이 알려져 있다.

$$y(t) = e^{at}\left[y_0 + \int_0^t e^{-a\tau} f(\tau) d\tau \right] = e^{at} y_0 + \int_0^t e^{a(t-\tau)} f(\tau) d\tau$$

이때 우변의 마지막 항을 합성곱 형태로 쓰면, 해의 공식[3]은 다음과 같이 간결해진다.

$$y(t) = e^{at} y_0 + f(t) * e^{at} = e^{at} y_0 + e^{at} * f(t)$$

나. 선형연립미분방정식

선형연립미분방정식의 경우도 이와 비슷한 해공식을 얻을 수 있는지 살펴보자. 먼저 다음과 같이 초깃값을 갖는 비동차 선형연립미분방정식을 생각해 보자.

$$\frac{d\mathbf{y}}{dt} = A\mathbf{y} + \mathbf{f}(t),\ \mathbf{y}(0) = \mathbf{y}_0 \qquad \text{- (2)}$$

여기서 벡터함수 $\mathbf{y}(t) = \big(y_1(t), \cdots, y_n(t)\big) \in \mathbb{R}^n$와 $\mathbf{f}(t) = \big(f_1(t), \cdots, f_n(t)\big) \in \mathbb{R}^n$는 $t \geq 0$일 때 정의되며, $A = [a_{ij}]$는 실수를 성분으로 하는 n차 정사각행렬이다.

비동차 미분방정식 (2)는 공학의 많은 수학적 모델링에서 나타나는데, 이때 $\mathbf{f}(t)$와 $\mathbf{y}(t)$는 각각 시스템의 입력과 출력으로 인식된다. 예를 들어, 자율주행 자동차를 생각해 보자. 자율주행의 핵심기술에는 차량의 위치를

---

[3] 여기서 합성곱은 정의 2에서 사용한 형식 $(f * g)(t)$와는 다른 $f(t) * g(t)$을 사용하였다. 공학수학에서는 $f(t) * g(t)$와 같은 형식으로도 많이 사용한다(Kreyszig, 2022).

파악하는 측위(localization) 기술, 차량 주변 환경을 인식하는 인지(perception) 기술, 차량의 진행 방향을 결정하는 계획(planning) 기술, 그리고 차량의 운동을 통제하는 제어(control) 기술이 있다. 여기서 조향(steering)을 자동으로 제어하는 시스템, 즉 자동제어시스템(automatic control system)을 설계할 때, 앞의 두 바퀴의 방향은 제어변수 또는 출력으로, 운전대의 회전방향을 구동신호 또는 입력으로 생각할 수 있다. 만약 자동차의 속도를 제어하는 것이 목적이라면 가속장치(accelerator)에 가해지는 압력을 구동신호로, 자동차의 속도를 제어변수로 볼 수 있다. 즉 전체적으로 차량의 제어시스템은 조향각과 가속도라는 두 개의 입력과 진행방위각과 속도라는 두 개의 출력을 갖는 시스템으로 단순화하여 생각할 수 있다. 이 경우 제어시스템은 조향장치와 자동차의 운동을 기술하는 동역학(dynamics)으로 구성된다. 그리고 이를 구체적으로 서술하는 상태방정식(state equation)은 대부분 위와 같은 비동차 선형연립미분방정식 (2)로 표현된다(Golnaraghi & Kuo, 2017).

이러한 비동차 선형연립미분방정식에 대한 다양한 풀이와 이론적 설명이 있으나 여기서는 라플라스 변환과 행렬지수(matrix exponential)를 이용해서 합성곱 형태의 해를 표현한다. 먼저 행렬지수는 n차 정사각행렬 $A=[a_{ij}]$에 대하여 다음과 같이 정의되는 행렬이다.

$$e^{At}=\sum_{m=0}^{\infty}\frac{A^m t^m}{m!}$$

여기서 $A^0=I$인 n차 단위행렬이다. 따라서 e^{At}도 n차 정사각행렬이다. 행렬 급수는 다소 생소할 수 있으나 미분적분학에서 e^{at}의 테일러 급수를 알고 있는 사람이라면 e^{At} 급수식의 형태가 익숙할 것이다. 단지 상수 a가 행렬 A로 바뀐 것밖에는 없다. 수렴조건도 매우 유사하여 위의 행렬급수는 모든 실수 t에 대해 수렴함을 증명할 수 있다(Horn & Johnson, 2013). 만약 행렬 A가 대각화 가능하면 $A=PDP^{-1}$로 나타낼 수 있으므로, e^{At}는 다음과 같이 계산된다.

$$e^{At}=Pe^{Dt}P^{-1},\quad e^{Dt}=\begin{bmatrix}e^{\lambda_1 t} & & \\ & \ddots & \\ & & e^{\lambda_n t}\end{bmatrix}$$

여기서 λ_k는 A의 고윳값들이다. 만약 행렬 A가 대각화 가능하지 않을 때는 조르당 표준형(Jordan Canonical Form)으로 나타낼 수 있다. 이때 $A=QJQ^{-1}$이므로, e^{At}는 다음과 같이 계산된다.

$$e^{At}=Qe^{Jt}Q^{-1},\quad e^{Jt}=\begin{bmatrix}e^{J_1 t} & & \\ & \ddots & \\ & & e^{J_k t}\end{bmatrix}$$

여기서 $J_i,\ i=1,\cdots,k$는 크기가 $m_i\times m_i$인 조르당 블록 (행렬)이고 k는 행렬 A의 서로 다른 고윳값의 갯수이다. 그리고 $e^{J_i t}$도 크기가 $m_i\times m_i$인 행렬로 다음과 같이 계산된다.

$$e^{J_i t}=e^{\lambda_i t}\begin{bmatrix}1 & t & \frac{t^2}{2!} & \cdots & \frac{t^{m_i-1}}{(m_i-1)!}\\ 0 & 1 & t & \cdots & \frac{t^{m_i-2}}{(m_i-2)!}\\ \vdots & & \ddots & & \vdots\\ 0 & & & & t\\ 0 & & & & 1\end{bmatrix}$$

즉 A가 어떤 행렬이든 e^{At}형태를 정해진 공식에 의해 찾을 수 있다. 이제 선형연립미분방정식 (2)를 풀기 위해 e^{At}의 라플라스 변환을 알아보자.

정리 3. n차 정사각행렬 e^{At}의 라플라스 변환은 다음과 같다.

$$\mathcal{L}(e^{At}) = (sI - A)^{-1}$$

여기서 I는 n차 단위행렬이고 s는 A의 고윳값이 아니어야 한다.

따라서 $(sI-A)^{-1}$의 역라플라스 변환(inverse Laplace Transform)은 $\mathcal{L}^{-1}\{(sI-A)^{-1}\} = e^{At}$이다. 이는 일변수 함수 e^{at}의 라플라스 변환인 $\mathcal{L}(e^{at}) = \frac{1}{s-a}$과 그 역라플라스 변환 $\mathcal{L}^{-1}\left\{\frac{1}{s-a}\right\} = e^{at}$ 과 유사하다. 참고로 $(sI-A)^{-1}$는 다음과 같이 표현할 수 있다.

$$(sI-A)^{-1} = \frac{1}{s}\left(I - \frac{A}{s}\right) = \frac{1}{s}\sum_{m=0}^{\infty}\left(\frac{A}{s}\right)^m$$

그리고 적당한 행렬 노름(norm)에 대해 $|s| > \|A\|$ 이면 위의 행렬급수가 $(sI-A)^{-1}$로 수렴함을 보일 수 있다. 이 역시 다음과 같이 $\frac{1}{s-a}$를 무한급수로 나타낸 식과 비슷한 형태라는 점을 알 수 있다.

$$\frac{1}{s-a} = \frac{\frac{1}{s}}{1-\frac{a}{s}} = \frac{1}{s}\sum_{m=0}^{\infty}\left(\frac{a}{s}\right)^m,\ |s| > |a|$$

이제 비동차 선형연립미분방정식 (2)를 풀기 위해 양변에 라플라스 변환을 취한다.

$$\mathcal{L}\left(\frac{d\mathbf{y}}{dt}\right) = \mathcal{L}(A\mathbf{y} + \mathbf{f})$$

$\mathcal{L}(\mathbf{y}) = \mathbf{Y}(s)$, $\mathcal{L}(\mathbf{f}) = \mathbf{F}(s)$라고 하면 부분적분과 라플라스 변환의 선형성을 통해 다음과 같은 사실을 알 수 있다.

$$\mathcal{L}\left(\frac{d\mathbf{y}}{dt}\right) = s\mathcal{L}(\mathbf{y}) - \mathbf{y}(0) = s\mathbf{Y}(s) - \mathbf{y}_0, \qquad \mathcal{L}(A\mathbf{y}+\mathbf{f}) = A\mathbf{Y}(s) + \mathbf{F}(s)$$

따라서 식은 다음처럼 정리된다.

$$\mathbf{Y}(s) = (sI - A)^{-1}(\mathbf{F}(s) + \mathbf{y}_0)$$

그리고 양변에 역라플라스 변환을 취하면 해를 찾을 수 있다. (Cox, 2022)

$$\mathbf{y} = e^{At}\mathbf{y}_0 + \mathcal{L}^{-1}\{(sI-A)^{-1}\mathbf{F}(s)\}$$

우변의 마지막 항을 처리하기 위해 $\mathcal{L}^{-1}\{(sI-A)^{-1}\} = e^{At}$라는 사실을 이용하면

$$\mathcal{L}^{-1}\{(sI-A)^{-1}\mathbf{F}(s)\} = \int_0^t e^{A(t-\tau)}\mathbf{f}(\tau)d\tau \in \mathbb{R}^n$$

이다. 따라서 해의 공식은 다음과 같다.

$$\mathbf{y} = e^{At}\mathbf{y}_0 + \int_0^t e^{A(t-\tau)}\mathbf{f}(\tau)d\tau$$

여기서 우변의 $\int_0^t e^{A(t-\tau)}\mathbf{f}(\tau)d\tau$은 행렬과 벡터의 성분별 계산 과정에서 합성곱 연산이 필요하다. 예를 들어, 행렬 A는 많은 공학적 응용에서 대칭행렬인 경우가 많은데, 이때는 항상 대각화가 가능하므로 간단히 행렬지수를 찾을 수 있다. 그러면 적분식은 다음과 같이 나타낼 수 있다.

$$\int_0^t e^{A(t-\tau)}\mathbf{f}(\tau)d\tau = \int_0^t Pe^{D(t-\tau)}P^{-1}\mathbf{f}(\tau)d\tau = P\begin{bmatrix} e^{\lambda_1 t} * g_1(t) \\ \vdots \\ e^{\lambda_n t} * g_n(t) \end{bmatrix}$$

여기서 g_k는 벡터 $P^{-1}\mathbf{f}$의 k번째 성분이다. 이는 미분방정식 (1)의 해 공식 $y(t) = e^{at}y_0 + f(t) * e^{at}$와 매우 유사함을 알 수 있다. 마찬가지로 행렬 A가 대각화가 되지 않더라도 조르당 표준형을 통해 행렬지수를 쉽게 찾을 수 있다. SageMath에서는 조르당 표준형을 찾는 jordan_form이라는 명령어가 있으므로, 이를 이용해 e^{At}를 계산할 수 있고, 같은 방법으로 각 성분별 계산 과정에서 합성곱을 이용해 $\int_0^t e^{A(t-\tau)}\mathbf{f}(\tau)d\tau$를 찾을 수 있다. 다음 예제를 통해 살펴보자.

예제 1. 선형연립미분방정식 $\mathbf{y}'(t) = A\mathbf{y} + \mathbf{f}(t)$, $\mathbf{y}(0) = \mathbf{y}_0$의 해를 구하라($t \geq 0$). 여기서 A, $\mathbf{f}(t)$, $\mathbf{y}_0$는 다음과 같다.

$$A = \begin{bmatrix} 2 & 1 & 1 \\ 2 & 1 & -2 \\ -1 & 0 & -2 \end{bmatrix},\ \mathbf{f}(t) = \begin{bmatrix} t \\ \cos t \\ \sin t \end{bmatrix},\ \mathbf{y}_0 = \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix}$$

전체 코드가 길고 복잡하기 때문에 단계별로 나누어 설명한다. 아래에 소개하는 코드를 모두 복사해서 하나의 실행창에 수행하면 원하는 결과를 얻을 수 있다.[4] 먼저 다음 코드를 활용하여 행렬 A의 조르당 표준형을 찾는다.

```
A = matrix(QQ, [[2, 1, 1], [2, 1, -2], [-1, 0, -2]])  # 행을 기준으로 행렬 입력
[J, Q] = A.jordan_form(transformation = true)          # 조르당 표준형  A*Q = Q*J
print(J)
print(Q)
```

이어서 행렬 A가 갖는 조르당 표준형의 행렬지수 $E = e^{Jt}$를 다음과 같이 구한다.

```
var('t')
E = exp(J*t)
```

[4] 웹 실습실 http://matrix.skku.ac.kr/SKKU-EM-2020/conv4ai/desys.htm에서도 확인해 볼 수 있다.

위의 계산 결과를 별도로 출력하면 $E=\begin{bmatrix} e^{3t} & 0 & 0 \\ 0 & e^{-t} & te^{-t} \\ 0 & 0 & e^{-t} \end{bmatrix}$ 이다. 그리고 해 공식 $\mathbf{y}=e^{At}\mathbf{y}_0+\int_0^t e^{A(t-\tau)}\mathbf{f}(\tau)d\tau$ 을 이용하려면 $\mathbf{y}_h=e^{At}\mathbf{y}_0$와 $\mathbf{y}_p=\int_0^t e^{A(t-\tau)}\mathbf{f}(\tau)d\tau$를 구해야 하므로 먼저 $\mathbf{y}_h$를 계산한다.

```
y0 = vector([-1, -1, 1])
yh = Q*E*Q.inverse()*y0
```

여기서 E는 앞서 계산한 조르당 표준형 행렬지수이다. 다음 $\mathbf{y}_p$를 계산하기 위해 일단 $\mathbf{g}(t)=Q^{-1}\mathbf{f}(t)$를 계산한다.

```
f = vector([t, cos(t), sin(t)])
g = Q.inverse()*f
```

이를 이용하여 $\mathbf{y}_p=\int_0^t e^{A(t-\tau)}\mathbf{f}(\tau)d\tau=Q\int_0^t e^{J(t-\tau)}\mathbf{g}(\tau)\,d\tau$를 계산하고, 최종적으로 앞에서 계산한 $\mathbf{y}_h$와 $\mathbf{y}_p$를 더하면 구하려는 해를 얻을 수 있다. 그런데 $\int_0^t e^{J(t-\tau)}\mathbf{g}(\tau)\,d\tau$은 성분별로 계산하는 과정에서 합성곱 연산을 해야 한다. SageMath에는 합성곱 적분 연산에 대한 별도의 명령어가 없으므로 이를 수행하는 함수를 다음과 같이 정의한다.

```
def convolution_integral(f, g):              # 합성곱 적분 연산 함수 정의
    t, u = var('t, u')
    assume(t > 0)
    return integral(f(u)*g(t - u), u, 0, t)
```

위의 convolution_integral은 두 함수가 $t<0$일 때 함숫값이 0이 되는 경우에만 합성곱 적분을 계산하는 프로그램 함수이다. 주어진 행렬 A의 조르당 표준형 행렬지수 $E=e^{Jt}$의 0이 아닌 성분들은 대각선 성분들과 2행 3열 성분밖에 없으므로, 아래 코드를 활용하여 해당 성분과 관련된 부분만 계산하면 된다.

```
g0(t) = g[0]; g1(t) = g[1]; g2(t) = g[2]
e00(t) = E[0,0]; e11(t) = E[1,1]; e12(t) = E[1,2]; e22(t) = E[2,2]
p0 = convolution_integral(e00, g0)
p1 = convolution_integral(e11, g1) + convolution_integral(e12, g2)
p2 = convolution_integral(e22, g2)
p = vector([p0, p1, p2])
yp = Q*p
sol = yh + yp
show('y1 = ', sol[0])
show('y2 = ', sol[1])
show('y3 = ', sol[2])
```

실제 구한 해 $\mathbf{y} = (y_1, y_2, y_3)$는 상당히 복잡하다. 이를 각 성분함수별로 표현하면 다음과 같다.

$$y_1 = \frac{3}{8}(3t+4)e^{-t} + \frac{3}{4}te^{-t} + \frac{2}{3}t - \frac{3}{4}\cos t - \frac{223}{288}e^{3t} + \frac{15}{32}e^{-t} - \frac{1}{4}sint - \frac{13}{9}$$

$$y_2 = -\frac{3}{4}(3t+4)e^{-t} - \frac{3}{2}te^{-t} - 2t + \frac{6}{5}\cos t - \frac{223}{240}e^{3t} - \frac{15}{16}e^{-t} + \frac{3}{5}\sin t + \frac{8}{3}$$

$$y_3 = -\frac{3}{8}(3t+4)e^{-t} - \frac{3}{4}te^{-t} - \frac{1}{3}t + \frac{1}{20}\cos t + \frac{223}{1440}e^{3t} + \frac{45}{32}e^{-t} + \frac{13}{20}\sin t + \frac{8}{9}$$

그러나 합성곱과 행렬지수를 프로그램에 응용한 덕분에 완전히 코드만으로 문제 해결이 가능하였다.

앞서 소개했듯이, 자율주행 자동차의 구동과 관련된 조향, 감가속 등을 자동으로 제어하기 위해서는 적절히 입력을 통제해 출력을 제어하는 시스템을 설계해야 한다. 이러한 제어시스템을 이해하기 위해서는 상태방정식의 변수, 즉 상태변수(state variable)를 해석할 수 있는 수학 지식이 필요하고, 그 기초로써 선형연립미분방정식에 대한 이해가 필수적이다. 제어시스템의 모델링에서 또 하나 주목할 부분은 비동차 연립방정식이 중요한 역할을 한다는 점이다. 비동차 연립방정식을 통해 입력과 출력의 관계를 명확하게 이해해야 시스템을 자동으로 제어할 수 있도록 설계할 수 있다. 합성곱와 행렬지수의 개념을 이용하면 비동차 선형연립방정식 해의 구조를 쉽고 간결하게 파악할 수 있을 뿐만 아니라 코딩을 이용해서 계산할 수 있었다. 참고로 라플라스 변환을 통해 비동차 선형연립방정식의 풀이를 설명한 이유는 제어시스템을 해석하는 수학적 도구로 라플라스 변환이 사용되기 때문이다. 게다가 자율주행 차량의 차체를 제어하기 위해서는 최종 판단을 하는 소프트웨어와 연동하여 차량 운행을 제어해야 한다(장승주, 2016). 최근에는 자동차의 핵심 부품이 전자화되고 있어서 기존의 제어시스템에 비해 원하는 목표 제어량에 빠르게 반응할 수 있도록 개선되고 있다(김현규·허건수, 2018). 그러므로 관련 기술들을 연구하고 개발하는 직종으로 진출하려는 학생들은 코딩을 통해 미분방정식을 다룰 수 있는 능력을 길러야 할 것이다.

4. 행렬합성곱과 인공지능

이산함수가 유한개의 정수 집합을 정의역으로 할 때 그 합성곱은 유한합이다. 예를 들어, 이산함수 f와 g가 $n = -N, -N+1, \cdots, -2, 1, 0, 1, 2, \cdots, N-1, N$에서 정의된다면, 그 합성곱은 다음과 같다.

$$(f * g)(n) = \sum_{k=-N}^{k=N} f(k)\, g(n-k)$$

시작항을 $-N$이 아닌 1로 두면 유한 합성곱은 실제로는 벡터의 합성곱이 된다.

정의 4. 두 벡터 $\mathbf{a} = [a_1, a_2, \cdots, a_m]$와 $\mathbf{b} = [b_1, b_2, \cdots, b_n]$의 합성곱 $\mathbf{a} * \mathbf{b} = \mathbf{c} = [c_1, c_2, \cdots, c_{m+n-1}]$은 다음과 같이 정의된다.

$$c_i = \sum_p a_p b_{i-p+1}$$

여기서 p는 가능한 인덱스만 계산한다. 특히 $p = \max(1, i-n+1), \cdots, \min(i, m)$이다.

SageMath에서는 convolution이란 명령어를 사용하여 이를 계산할 수 있다. 예를 들어, 다음과 같이 명령어를 활용하여 벡터 $\mathbf{a} = [1,2,3,4,5]$와 $\mathbf{b} = [6,7]$의 합성곱을 계산하면 $\mathbf{a} * \mathbf{b} = [6, 19, 32, 45, 58, 35]$을 얻는다.

```
sage: convolution([1, 2, 3, 4, 5], [6, 7])  # 벡터의 합성곱
[6, 19, 32, 45, 58, 35]
```

벡터의 합성곱을 조금 변형하면 다음과 같이 행렬의 합성곱을 정의할 수 있다. 단지 행렬 성분의 인덱스가 두 개인 것만 주의하면 된다.

정의 5. $m\times n$ 행렬 $A=[a_{ij}]$와 $k\times l$ 행렬 $B=[b_{ij}]$의 합성곱은 다음과 같은 성분을 갖는 $(m+k-1)\times(n+l-1)$ 행렬 $A*B=C=[c_{ij}]$로 정의된다.

$$c_{ij}=\sum_{p}\sum_{q}a_{p,q}b_{i-p+1,j-q+1}$$

여기서 p와 q는 가능한 인덱스만 계산한다.

행렬의 합성곱과 관련하여 다음 예제를 살펴보자.

예제 2. 행렬 A와 B가 각각 다음과 같을 때 $A*B$를 계산하여라.

$$A=\begin{bmatrix}1&2\\3&4\end{bmatrix},\ B=\begin{bmatrix}5&6\\7&8\end{bmatrix}$$

행렬 $C=A*B$이라고 하면, A와 B의 크기가 2×2이므로 C는 3×3여야 한다. 그리고 $C=[c_{ij}]$의 각 성분을 계산하면 첫번째 행은 $c_{11}=1\cdot 5=5$, $c_{12}=2\cdot 5+1\cdot 6=16$, $c_{13}=2\cdot 6=12$, 두번째 행은 $c_{21}=3\cdot 5+1\cdot 7=22$, $c_{22}=4\cdot 5+3\cdot 6+2\cdot 7+1\cdot 8=60$, $c_{23}=4\cdot 6+2\cdot 8=40$, 세번째 행은 $c_{31}=3\cdot 7=21$, $c_{32}=4\cdot 7+3\cdot 8=52$, $c_{33}=4\cdot 8=32$이 된다. 그러므로 계산 결과는 다음과 같다.

$$A*B=\begin{bmatrix}5&16&12\\22&60&40\\21&52&32\end{bmatrix}$$

이러한 합성곱에서 행렬 A는 핵(kernel) 또는 필터(filter)라고도 하며, 이미지 데이터를 필터링할 때 사용한다. 한편, 주어진 필터 A에 대해 합성곱의 기본 성질 중 분배법칙과 상수에 관한 법칙을 통해 행렬합성곱은 선형변환(linear transformation)이 됨을 알 수 있다. 즉, $A*(B+C)=A*B+A*C$이고, 임의의 상수 r에 대해 $A*(rB)=r(A*B)$이다.

SageMath에서는 행렬합성곱에 대한 별도의 명령어는 없다. <표 Ⅲ-1>은 행렬합성곱을 계산하는 함수를 정의하는 코드로, matrix_convolution 함수는 MATLAB의 conv2 명령어의 실행 방식과 기본적으로 동일하다.[5]

[5] https://www.mathworks.com/help/matlab/ref/conv2.html

<표 Ⅲ-1> 행렬합성곱 계산 코드

```
def matrix_convolution(A,B):
    m = A.nrows()
    n = A.ncols()
    k = B.nrows()
    l = B.ncols()
    c = []
    for i in range(m+k-1):
        for j in range(n+l-1):
            p1 = max(0, i-k+1)
            p2 = min(i, m-1)+1
            q1 = max(0, j-l+1)
            q2 = min(j, n-1)+1
            d = []
            for p in range(p1,p2):
                for q in range(q1,q2):
                    d.append(A[p,q]*B[i-p,j-q])
            c.append(sum(d))
    return matrix(RDF, m+k-1, n+l-1, c)
```

다음 예제는 MNIST 손글씨 이미지[6]를 행렬합성곱을 이용해서 변환한 경우이다.

예제 3. 주어진 이미지 파일에 다음의 필터를 사용하여 합성곱을 계산하라.

$$A = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

행렬 A는 mean filter로, 입력한 이미지 데이터를 이웃한 픽셀들의 평균값으로 대체한다. <표 Ⅲ-2>는 변환하려는 이미지 데이터를 B라는 행렬로 나타내어 A와의 행렬합성곱을 계산하는 파이썬 코드이다. 원래 이미지는 foo.png에, 변환한 이미지는 foo_conv.png에 저장한다. 입력한 이미지와 프로그램을 실행한 결과가 [그림 Ⅲ-4]이다. 원래 이미지보다 변환된 이미지가 더 흐릿하고 살짝 작아진 것을 알 수 있다. 예제 3의 mean filter 이외에도 이미지의 윤곽을 찾기 위해 음영의 변화량을 조절하는 edge detection filter나 이미지를 더 선명하게 하는 sharpening filter 등 무수히 많은 종류의 필터가 존재한다.

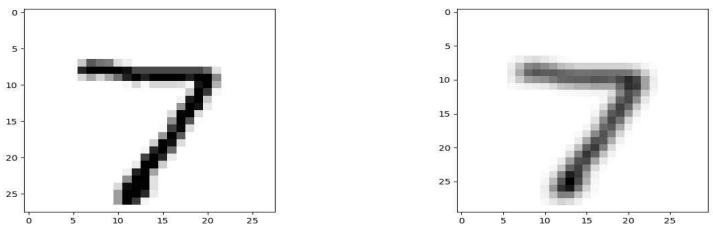
6) https://media.githubusercontent.com/media/freebz/Make-Your-Own-Neural-Network/master/mnist_dataset/mnist_test_10.csv

<표 III-2> 행렬합성곱을 이용한 이미지 변환 프로그램

```
import numpy
import matplotlib.pyplot
import csv
import urllib.request
import codecs
url = 'https://media.githubusercontent.com/media/freebz/Make-Your-Own-Neural-Network/master/mnist_dataset/mnist_test_10.csv'
response = urllib.request.urlopen(url)
training_data_file = csv.reader(codecs.iterdecode(response, 'utf-8'))
training_data_list = list(training_data_file)
all_values = training_data_list[0]
image_array = numpy.asfarray(all_values[1:]).reshape((28,28))
matplotlib.pyplot.imshow(image_array, cmap = 'Greys', interpolation = 'None')
matplotlib.pyplot.savefig('foo.png')
response.close()

def matrix_convolution(A,B):
    m, n = A.shape
    k, l = B.shape
    c = []
    for i in range(m+k-1):
        for j in range(n+l-1):
            p1 = max(0, i-k+1)
            p2 = min(i, m-1)+1
            q1 = max(0, j-l+1)
            q2 = min(j, n-1)+1
            d = []
            for p in range(p1,p2):
                for q in range(q1,q2):
                    d.append(A[p,q]*B[i-p,j-q])
            c.append(sum(d))
    return numpy.asfarray(c).reshape(m+k-1, n+l-1)

A = numpy.array([[1, 1, 1], [1, 1, 1], [1, 1, 1]]); A = 1/9*A
B = image_array
C = matrix_convolution(A,B)
matplotlib.pyplot.imshow(C, cmap = 'Greys', interpolation = 'None')
matplotlib.pyplot.savefig('foo_conv.png')
```



[그림 III-4] 행렬합성곱을 통한 이미지 변환(왼쪽: 원래 이미지, 오른쪽: 변환한 이미지)

실제 이미지 데이터에 행렬합성곱을 적용할 때는 공학적 필요에 따라 다음과 같이 수학적 정의를 변형시킨 알고리즘을 사용한다.

정의 6. $m\times n$ 행렬 $W=[w_{ij}]$와 $k\times l$ 행렬 $B=[b_{ij}]$의 합성곱은 다음과 같은 성분을 갖는 $(m+k-1)\times(n+l-1)$ 행렬 $W*B=C=[c_{ij}]$로 정의한다.

$$c_{ij}=\sum_p\sum_q w_{p,q}b_{i+p-1,j+q-1}$$

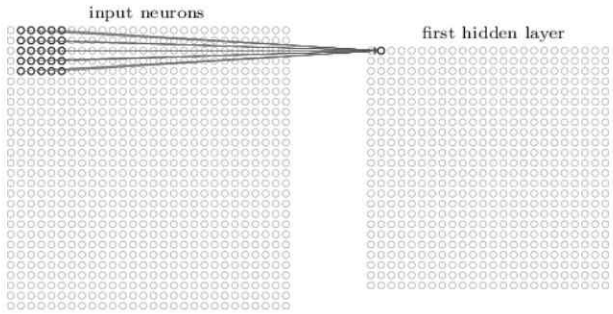
여기서 p와 q는 가능한 인덱스만 계산한다.

여기서 $W=[w_{ij}]$는 정의 5의 행렬 A가 가진 성분의 인덱스들을 $p\mapsto m-p+1$와 $q\mapsto n-q+1$로 치환(permutation)한 행렬이다. 사실 정의 6은 합성곱과 매우 유사한 연산을 하는 교차상관(cross-correlation)으로, 합성곱과 교차상관은 계산할 때 한 함수의 변수나 인덱스를 음수 부호로 반전하는 것만 빼고는 사실상 동일한 함수를 얻는다. CNN에서는 실제로 합성곱이 아닌 교차상관을 사용하는데, 그 이유는 합성곱 연산을 하려면 행렬 B의 인덱스들을 모두 뒤집어야 하는 불편함이 따르기 때문이다. 행렬 A와 B의 인덱스와 행렬 성분의 개수는 유한하므로 적절히 치환하면 행렬합성곱을 적용하든 행렬에 교차상관을 적용하든 결과에 차이가 없다. 참고로 정의 6을 padding과 stride 개념을 이용해 더욱 일반화할 수 있다. padding은 입력하는 데이터 행렬 주변을 특정한 값으로 채워 늘리는 것을 말한다. 그리고 stride는 필터가 이동하는 간격을 일컫는다. 정의 5는 stride가 1이고 padding이 행은 $m-1$, 열은 $n-1$인 경우로 행렬 $B=[b_{ij}]$를 $(m+k-1)\times(n+l-1)$로 늘린 후 확장한 부분을 0인 성분으로 채운 것을 계산하는 것이다.

완전연결계층(fully connected layer)을 사용하는 단순 인공신경망과 비교해서 이제 행렬합성곱을 이용한 CNN의 구조를 간략히 살펴보자. 완전연결계층 인공신경망 모형에서는 이미지 데이터를 벡터로 입력한다. 그러나 [그림 Ⅲ-5]처럼 CNN에서는 이미지 데이터를 행렬로 표현한다. 이를 합성곱층(convolution layer)이라고 부르고, 행렬의 각 성분을 인공 뉴런으로 간주한다. 입력 신호의 총합을 출력 신호로 변환하는 함수를 활성화 함수(activation function) σ라고 하면, 은닉층(hidden layer)을 가지는 인공신경망에서 입력 데이터를 받은 은닉층의 (i,j)번째 인공 뉴런의 출력값은 다음과 같다.

$$\sigma\left(d+\sum_p\sum_q w_{p,q}b_{i+p-1,j+q-1}\right)$$

여기서 d는 편향(bias)를 의미한다. 그리고 w_{ij}는 오차역전파법(back propagation)에 의해 학습해야 할 매개변수(parameter)이다. 활성화 함수로는 시그모이드 함수(Sigmoid function), 쌍곡탄젠트 함수(hyperbolic tangent function) 등이 사용된다. 오차역전파법을 계산할 때 행렬-벡터 식으로 표현해야 할 때가 있다. 앞서 합성곱 연산은 선형성을 가지고 있다고 했으므로 입력 데이터인 행렬 B를 벡터로 표현한다면 행렬합성곱을 선형변환으로 계산할 수 있는 행렬을 찾을 수 있다. 따라서 오차역전파법 적용이 가능하다.



[그림 III-5] CNN의 뉴런 연결 구조 예시[7]

완전연결계층 인공신경망 알고리즘에 비해 행렬합성곱을 이용한 CNN 모델이 심층학습에서 갖는 큰 장점은 매개변수의 수를 많이 줄일 수 있다는 것이다. 예를 들어 MNIST 문제의 모형 중 편향이 없으면서 데이터 행렬 B가 28×28인 경우를 생각해 보자. CNN의 경우 필터로 사용되는 행렬 W가 5×5이면 학습해야 할 매개변수는 25개이고 은닉층에 연결할 수 있는 뉴런의 개수는 최대 $(28+5-1) \times (28+5-1) = 1024$개이다. 그리고 padding과 stride를 통해 뉴런의 개수를 조정할 수 있다. 그러나 완전연결계층의 경우는 $28 \times 28 = 784$의 입력 뉴런에 100개 은닉층 뉴런만 연결해도 학습시켜야 할 매개변수(parameter)가 $784 \times 100 = 78400$개가 된다. CNN 모형을 더욱 정교하게 하도록 W의 채널(channel)을 늘린 경우에도 완전연결계층보다 CNN의 매개변수 개수가 훨씬 적다. 예를 들어 채널이 10개라고 한다면 CNN에서 학습해야 할 매개변수는 $25 \times 10 = 250$개밖에 되지 않는다. 두 알고리즘의 결과에 큰 차이가 나지 않는다면 CNN이 훨씬 효율적이다.

완전연결계층 모형의 또 다른 문제점은 이미지 데이터가 갖는 공간적 구조(spatial structure)를 무시한다는 것이다. 예를 들어 공간적으로 가까운 픽셀은 값이 비슷하거나 서로 밀접하게 관련되어 있지만, 거리가 먼 픽셀끼리는 관련이 적거나 없을 것이다. 이미지 데이터에는 공간적으로 이러한 정보들이 내포되어있지만 완전연결계층에서는 행렬형 데이터를 1차원 벡터형 데이터로 펼치게 되면서 이러한 정보들이 사라지게 된다. 이렇게 행렬의 특성과 행렬합성곱의 개념을 활용한다면 데이터 구조의 특징을 더 깊이 파악할 수 있다.

자율주행에서 CNN은 일반적으로 카메라와 센서의 시각적 데이터를 이용하여 물체를 감지하고 추적하며, 물체의 크기나 속도와 같은 특성을 식별하는 데 사용된다. 예를 들어, 입력 데이터는 물체의 경계, 형태 등과 같은 특징을 감지하도록 설계된 일련의 합성곱 층을 통해 처리되어 차선을 감지하거나 자동차, 보행자, 교통 표지판 등을 분류한다. 그리고 입력 데이터의 처리에 따라 차량이 주변 환경에 대한 판단을 내릴 수 있도록 하면 소프트웨어와 연동하여 앞에서 소개한 제어시스템을 통해 자율주행 차량을 제어할 수 있다(황광복·박진현, 2020).

공학수학이 수학의 광범위한 내용을 포함하고 있는 만큼 모든 주제를 한 번에 재검토하고 분석하는 것은 현실적으로 불가능하지만, 상황이나 필요에 따라 기존의 다룬 주제도 새롭게 구성하여 교수·학습해야 함을 합성곱을 예로 들어 확인하였다. 본 연구는 공학수학을 지도하는 교수자들에게 도움이 될 기초자료가 될 것이며, 특히 제공된 코드는 앞으로 만날 다양한 공학 문제를 이해하고 해결하는데 바로 적용이 가능할 것이다.

---

7) https://eng.libretexts.org/Bookshelves/Computer_Science/Applied_Programming/Book%3A_Neural_Networks_and_Deep_Learning_(Nielsen) 6.1절의 이미지

Ⅳ. 결론 및 제언

본 논문에서는 인공지능의 원리를 이해하고 기술을 응용하는 능력을 기르는데 적합한 합성곱 교수·학습자료를 연구하였다. 연속함수 합성곱과 이산함수 합성곱을 일관되게 정리하고, 코딩으로 교수·학습이 이루어질 수 있도록 관련 코드를 개발하여 온라인 실습실 콘텐츠를 구축하였다. 먼저 전통적인 연속함수의 합성곱에서 출발하여 미분방정식에서 라플라스 변환과 함께 어떻게 합성곱이 응용될 수 있는지 설명하고, 해를 컴퓨터로 계산할 수 있도록 코드를 제공하였다. 이렇게 코딩을 통해 습득한 합성곱과 미분방정식의 지식이 자율주행 연구의 어떤 부분에서 활용될 수 있는지도 간략히 소개하였다. 이산함수는 연속함수를 샘플링한 함수로 이해하였으며 이에 대한 합성곱을 행렬합성곱까지 연계해서 서술하였다. 동시에 각 하위 주제별 내용에서 역시 코딩으로 교수·학습이 이루어질 수 있도록 관련 프로그램 코드와 실습실을 개발하였다.

이렇게 한 이유는 인공지능과 추후 공학의 타분야 기술이 결합할 수 있기 때문에, 교육적 차원에서 전체적인 흐름을 살펴볼 수 있도록 구성한 것이다. 고전적인 내용이라 하더라도 코딩을 활용하여 합성곱을 쉽고 효과적으로 교육하면 나중에 본인의 전공지식 중 인공지능 기술과 결합할 수 있는 부분을 탐색하고 실행하는 데 큰 도움이 될 것이다. 또한 학생들의 수학 지식이 인공지능의 원리, 예를 들어 심층신경망 등을 이해하는 데 쓰일 수 있을 뿐만 아니라, 인공지능에 대한 깊은 전공지식이 없이도 실제 활용하는 데 도움이 되게 하려는 것이다.

본 논문에서 소개한 합성곱의 응용 사례 이외에도, 푸리에 변환을 이용하여 무경계 확산방정식의 해공식을 유도하거나(Stanley, 1993), 독립적인 확률 변수들의 합으로 이루어지는 새로운 확률 변수의 질량함수 또는 밀도함수를 나타낼 때(Grinstead & Snell, 1997) 합성곱을 활용할 수 있다. 특히 인공신경망에 관심이 많은 교수자나 학습자라면, 심층신경망을 효율적으로 학습시키는데 중요한 방법인 오차역전파법과 최적화 알고리즘을 본 연구 자료와 연계해서 살펴볼 수 있다. 또한 합성곱신경망 연구에서 합성곱의 역연산인 역합성곱(deconvolution)도 함께 다루는 것을 고려해볼 수 있다. 역합성곱이란 예를 들어, 이미지 데이터와 관련한 행렬합성곱 $A * B = C$에서 행렬 A를 필터라고 할 때, 출력 이미지 C에 대해 입력 이미지 B를 거꾸로 찾는 과정을 말한다. 역합성곱을 통해 출력값이 계산되기까지 거쳐왔던 과정을 역추적하여 처음의 이미지까지 돌아간다면, 입력값의 어떤 부분이 출력값을 증폭시켰는지를 알 수 있다. 그러나 이러한 것들까지 모두 포함하면 내용이 지나치게 방대해지므로 본 논문에서는 인공지능 기술과 가까운 응용 사례 중심으로 선별하여 전개하였다.

본 연구에서 제시한 내용들이 앞으로 공학수학의 교과 과정 개편이나 새로운 교재를 구상하고 있는 사람에게, 합성곱과 관련하여 하나의 방향을 제시할 수 있기를 기대한다. 새로운 공학수학 커리큘럼에 행렬합성곱을 포함하는 방향도 진지하게 고민해 보아야 할 것이다.

참 고 문 헌

강주석・박찬일 (2017). 기계공학수학의 현황 분석을 통한 개편안 제시. 공학교육연구, **20(2)**, 50-56.

Kang, J., & Park, C. (2017). Suggestions for revision of mathematics curriculum by analysis of current mechanical engineering mathematics. *Journal of Engineering Education Research*, **20(2)**, 50-56.

김성옥・안경모・이종원 (2009). 공학전공자를 위한 대학수학교육과정 및 교과목 개발 연구. 수학교육논문집, **23(4)**, 961-976.

Kim, S.-O., Ahn, K., & Lee, J. (2009). A study for the development of curriculum and courses for mathematics for engineering majors. *Communications of Mathematical Education*, **23(4)**, 961-976.

김현규・허건수 (2018). 자율주행 기술 연구 동향 및 전망. 정보와 통신, **35(5)**, 3-12.

Kim, H.-G., & Huh, K. (2018). Autonomous vehicle technology research trends and prospects. *Information & Communications Magazine*, **35(5)**, 3-12.

서종진 · 유천성 · 최은미 (2007). 대학 교양수학의 교육 내용 구성에 관한 고찰: 생명·나노 관련 분야를 중심으로. 수학교육논문집, **21(3)**, 559-573.

Seo, J., Ryoo, C., & Choi, E. (2007). A study on construction of educational contents in college general mathematics. *Communications of Mathematical Education*, **21(3)**, 559-573.

에드워드 윌슨 (2005). 통섭: 지식의 대통합(최재천, 장대익 옮김). 사이언스북스.

Wilson, E. O. (1999). *Consilience: The unity of knowledge.* Vintage.

이상구 · 이재화 · 박준현 · 김응기 (2016). SageMath를 활용한 '대화형 공학수학 실습실'의 개발과 활용. 수학교육논문집, **30(3)**, 281-294.

Lee, S.-G., Lee, J.H., Park J.H. & Kim, E.-K. (2016). Interactive engineering mathematics laboratory. *Communications of Mathematical Education,* **30(3)**, 281-294.

이승우 (2008). 전기전자공학분야에서 수학/통계 내용분석. 한국수학사학회지, **21(4)**, 127-140.

Lee, S. (2008). A content analysis of math/stat in electrical & electronic engineering fields. *Journal for History of Mathematics*, **21(4)**, 127-140.

이재화 · 이상구 · 함윤미 (2022). 직업계 고등학교 졸업생 대상 'Math & 코딩'을 활용한 대학 미분적분학 교육 사례 연구. 수학교육논문집, **36(4)**, 611-626.

Lee, J.H., Lee, S.-G. & Ham, Y. (2022). Case study on college calculus education for vocational high school graduates with coding. *Communications of Mathematical Education*, **36(4)**, 611-626.

장승주 (2016). 자율 주행 자동차 관련 SW기술 동향. 정보와 통신, **33(4)**, 27-33.

Jang, S.-J. (2016). Software technology trends related to self-driving cars. *Information & Communications Magazine*, **33(4)**, 27-33.

전재복 (2008). 바람직한 대학기초수학 교육과정 운영방안: 공학기초수학을 중심으로. 수학교육논문집, **22(4)**, 399-416.

Jun, J.-B. (2008). Desirable management of basic mathematics curriculum in college. *Communications of Mathematical Education*, **22(4)**, 399-416.

정수연 · 송영무 (2011). 대학 공업수학 학습자료 개발 및 효과. 수학교육논문집, **25(2)**, 361-379.

Jeong, S., & Song, Y. (2011) Investigation of the effect of a learning program for university engineering mathematics. *Communications of Mathematical Education*, **25(2)**, 361-379.

황광복 · 박진현 (2020). CNN을 이용한 자율주행차 조향 제어. 한국정보통신학회논문지, **24(7)**, 834-841.

Hwang, K.-B. & Park, J.-H. (2020). Steering control of an autonomous vehicle using CNN. *Journal of the Korea Institute of Information and Communication Engineering*, **24(7)**, 834-841.

Cox, S. (2022). *Matrix analysis.* LibreTexts.
https://math.libretexts.org/Bookshelves/Linear_Algebra/Book%3A_Matrix_Analysis_(Cox)

Croft, A., Davison, R., Hargreaves, M., & Flint, J. (2017). *Engineering mathematics (5th Ed.).* Pearson Education.

Golnaraghi, F., & Kuo, B. (2017) *Automatic control systems (10th Ed.).* McGraw Hill.

Grinstead, C. M., & Snell, J. L. (1997) *Introduction to probability (2nd Ed.).* American Mathematical Society.

Horn, R. A., & Johnson, C. R. (2013). *Matrix analysis (2nd ed.).* Cambridge University Press.

Kreyszig, E. (2022). *Advanced engineering mathematics (10th Ed.).* Wiley.

O'Neil, P. V. (2017). *Advanced engineering mathematics (8th Ed.).* Cengage Learning.

Stanley J. F. (1993) *Partial differential equations for scientists and engineers*, Dover Publications.

Zill, D. G. (2020). *Advanced engineering mathematics (7th Ed.).* Jones & Bartlett Learning.

A Study on Teaching of Convolution in Engineering Mathematics and Artificial Intelligence

Lee, Sang-Gu
Department of Mathematics, Sungkyunkwan University
E-mail : sglee@skku.edu

Nam, Yun
Institute of Basic Science, Sungkyunkwan University
E-mail : yunnam@skku.edu

Lee, Jae Hwa†
Research Institute of Basic Sciences, Sungkyunkwan University
E-mail : jhlee2chn@skku.edu

Kim, Eung-Ki
Department of Mathematics, Sungkyunkwan University
E-mail : upusup@skku.edu

In mathematics, the concept of convolution is widely used. The convolution operation is required for understanding computer vision and deep learning in artificial intelligence. Therefore, it is vital for this concept to be explained in college mathematics education.

In this paper, we present our new teaching and learning materials on convolution available for engineering mathematics. We provide the knowledge and applications on convolution with Python-based code, and introduce Convolutional Neural Network (CNN) used for image classification as an example. These materials can be utilized in class for the teaching of convolution and help students have a good understanding of the related knowledge in artificial intelligence.

* 2000 Mathematics Subject Classification : 97U50, 97U70
* Key words : engineering mathematics, convolution, Laplace transform, artificial intelligence
* This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No.2021R1F1A1046714).
† Corresponding author