

인공지능을 위한 기초수학

(Basic Mathematics for Artificial Intelligence)

이상구 with 이재화

* 첫 강의 [Dummy를 위한 인공지능] <https://youtu.be/VZbv6BG-xIY> (12:50)

Math for AI 흄 폐이지 <http://matrix.skku.ac.kr/math4ai/>

 BigBook

현대차
정몽구 재단
Chung Mong-Koo Foundation

온드림빅북총서 009

인공지능을 위한 기초수학

Basic Mathematics for Artificial Intelligence

이상구 with 이재화



빅북이라 명명된 이 책은 지식공유의 세계적인 흐름에 동참하고
지적인 업적들이 세상과 인류의 지식이 되도록 하며, 누구나 쉽
게 접근하고 활용할 수 있는 환경을 만들고자 한다.

이 책의 저작권은 빅북(www.bigbook.or.kr)있으며 모든 용도로 활
용할 수 있다.

다만 상업용 출판을 하고자 하는 경우에는 사전에 문서로 된 허
락을 받아야 한다.

공유와 협력의 교과서 만들기 운동본부

인공지능을 위한 기초수학

Basic Mathematics for Artificial Intelligence

이상구 with 이재화

 BigBook

함께 만들고 함께 나누는 공유의 지식!

| 공유와 협력의 교과서만들기 운동본부 |

인간은 교육으로 성숙되고 사회는 지식으로 발전한다.

교육의 기회는 만인에게 평등하게 제공되어져야 하며 지식은 사회발전을 위하여 공유되어야 한다. 교육과 지식은 인류의 문화적 유산이며 나눔의 대상이다. 우리는 이러한 신념이 우리 사회의 교육평등과 보다 나은 미래를 위한 디딤돌임을 확신하며, 함께 만들고 함께 나누는 지식 창조와 공유의 새로운 지평을 열고자 한다.

빅북이라 명명된 이 책은 지식공유의 세계적인 흐름에 동참하고 지적인 업적들이 세상과 인류의 지식이 되도록 하며, 누구나 쉽게 접근하고 활용 할 수 있는 환경을 만들고자 한다.

이 책의 저작권은 빅북 (www.bigbook.or.kr)에 있으며 모든 용도로 활용할 수 있다. 다만 상업용 출판을 하고자 하는 경우에는 사전에 문서로 된 허락을 받아야 한다.

공유와 협력의 교과서만들기 운동본부

인류의 지식은 개인의 것이기에 앞서 문화의 유산입니다. 우리는 물려받은 지식의 토대 위에 지식을 창조한 것이며 이는 다음 세대도 그려할 것입니다. 우리의 삶을 풍요롭게 하는 지식은 공기와 같이 공유되어야 하며 이를 통해 더 나은 지식창조가 가능하다고 믿습니다.

이제 지식은 상아탑을 넘어 시민사회의 참여가 필요합니다. 이는 다양한 지식을 많은 전문가들이 가지고 있으며 그 변화속도는 상상하기 어렵기 때문입니다. 고등교육기관과 시민들이 협력한다면 다양한 견해를 담은 새롭고 혁신적인 지식이 창조될 수 있을 것이며, 함께 나누고 공유한다면 지식은 인류의 삶에 더 큰 기여를 할 수 있을 것입니다.

교육을 위한 지식들은 우선적으로 공유되어져야 하며 이는 모두에게 평등하게 제공되어야 한다고 생각합니다. 인종과 성별 그리고 지위의 부의 차이에 의하여 지식의 제공이 제한되는 것은 인간의 기본권이 침해되는 것입니다. 우리의 문화적인 유산인 지식이 그들을 필요로 하는 사람들에게 다가가 그들의 삶을 개선시킬 수 있도록 여건과 제도를 만들어 가는 것은 우리 지식인의 책무라고 생각합니다.

대학의 지식창조 활동의 결과물들도 이를 배워야 할 학생들에게 효과적으로 공유될 필요가 있으며, 이를 위해 지적재산권의 문제를 비롯한 많은 결집들을 시급히 개선되어야 합니다. 이제 대학의 지식을 갈망하는 우리 이웃들의 목마음을 채우기 위하여 작지만 먼 걸음을 시작합니다. 많은 뜻있는 분들의 도움으로 먼 길이 와롭지 않기를 바랍니다.

공유와 협력의 교과서만들기 운동본부

.. Contents

서 문	3
Python/Sage/R 언어 사용법	7
수학과 코딩!	18

PART 0 인공지능 개론

1. 거듭제곱법(Power Method)	21
2. MNIST 데이터셋을 활용한 손 글씨 숫자 인식 (패턴인식)	29
[읽을거리1] 인공지능, 알파고	31

PART I 행렬과 데이터분석

0. Big Picture	28
1. 벡터	34
1.1 벡터 (Vector)	37
1.2 내적	39
1.3 정사영	39
2. 선형연립방정식	43
2.1 선형연립방정식 (Linear System of Equations)	43
2.2 Gauss 소거법과 Gauss-Jordan 소거법	47
3. 행렬과 행렬식	51
3.1 행렬 연산	51
3.2 역행렬	57
3.3 기본행렬	59
3.4 선형연립방정식의 해집합, 특수행렬	62
3.5 행렬식	65
4. 일차독립과 기저(basis) 및 차원(Dimension)	70
4.1 일차독립과 부분공간	70
4.2 기저와 차원	73
4.3 정사영과 최소제곱해(least square solution), QR 분해	78

5. 선형변환 (Linear Transformations)	94
5.1 선형변환	94
5.2 핵(kernel)과 치역(range)	98
6. 고윳값, 고유벡터, 대각화(Diagonalization)	102
6.1 고윳값과 고유벡터	102
6.2 닮음 행렬(similar matrix)과 행렬의 대각화(Matrix Diagonalization)	103
6.3 직교대각화 (orthogonally diagonalizing)	105
7. SVD (특이값 분해, 특잇값 분해, singular value decomposition)	111
7.1 특이값 분해	111
7.2 일반화된 역행렬 (pseudo-inverse, Moore-Penrose Generalized Inverse)	114
8. 이차형식(quadratic form)	118
8.1 이차형식	118

PART II 다면수 미적분학과 최적화

1. 일변수함수와 미적분	131
1.1 함수	131
1.2 극한 (limit)	133
1.3 도함수(derivative)와 미분(differentiation)	134
1.4 미분의 응용 (Applications)	138
1.5 적분 (Integral)	141
2. 다변수함수와 미적분	144
2.1 벡터와 공간기하	144
2.2 벡터 함수	146
2.3 편도함수(Partial Derivative)와 그래디언트(gradients)	149
2.4 함수의 극대(Local Maximum), 극소	157
2.5 Gradient Descent Algorithm(경사-기울기 하강법, 傾斜下降法)	165
2.6 중적분 (double integral, multiple integral)	173
[부록 1] 미적분학의 상호연관성(Calculus Map)	185
[부록 2] 미적분학용 Sage/Python 코드	187
[부록 3] 미적분학 공식과 표(Table)	189

PART III 확률통계와 빅데이터

0. 통계학과 R	203
0.1 통계학	204
0.2 R 명령어 예시	204
0.3 학습할 내용	209
1. 순열, 조합, 확률	210
1.1 순열과 조합 (Counting Methods, 경우의 수를 구하는 법)	210
1.2 Multiset(중복집합) 의 순열	211
1.3 확률	213
1.4 베이즈 정리 (Bayes' theorem)	220
2. 확률변수 (random variable)	222
2.1 확률변수, 기댓값, 분산 및 표준편차	222
3. 확률분포	228
3.1 이산확률분포	229
3.2 연속확률분포	237
3.3 결합 확률분포, 공분산(Covariance)과 상관계수(Correlation)	259
4. 데이터 활용의 실제	267
[부록 4] 기초통계 개념	272

PART IV 빅데이터와 인공지능

1. 주성분분석	300
1.1 주성분 분석(Principal Component Analysis, PCA)	300
1.2 Principal Component의 유도	302
1.3 PC score	308
1.4 PCA 예제	310
2. 신경망	318
2.1 신경망(Neural Network)	318
3. 학생 프로젝트 사례	336
[읽을거리1] PageRank 알고리즘	354
[읽을거리2] 안면인식기술	363

... 머리말

『인공지능을 위한 기초수학』은 인공지능 분야의 진로를 탐색하는 다양한 전공의 대학생과 대학원생을 대상으로 인공지능에 꼭 필요한 수학의 주제들을 선별하여 한 학기용으로 만든 교재이다.

이 책은 처음부터 끝까지 인공지능(머신러닝, 딥러닝)을 이해하는데 필요한 기본적인 수학적 개념과 계산능력을 갖추도록 하는데 집중하고 있다. 먼저, 인공지능의 알고리즘을 이해하는 데 필요한 최소한의 수학 개념을 고교, 대학 수학 과정의 수준으로 설명하고, 이 개념들이 실제로 인공지능을 개발할 때 어떻게 쓰이는지, 잘 알려진 예와 알고리즘을 이용하여 쉽게 설명한다.

이 책은 인공지능 이해에 필수적인 수학 콘텐츠인 선형대수학, 다변수 미적분학, 기초통계와 확률, SVD, 주성분 분석(PCA) 및 그래디언트 알고리즘과 파이썬 및 R 코드를 포함하는 내용으로 구성된다. 특히 수학적인 이론, 풀이와 함께 관련 파이썬(Python) 및 R 코드를 교재에 제공하고, 클라우드 컴퓨팅 실습실을 활용하여, 어렵지 않게 복잡한 계산과 데이터 처리 및 시각화를 직접 수행하면서 학습할 수 있도록 한다.

(강의계획서 설명: <https://youtu.be/BLOMeZZqq7I>)



2019년 5월 수학체험한마당 (진주)

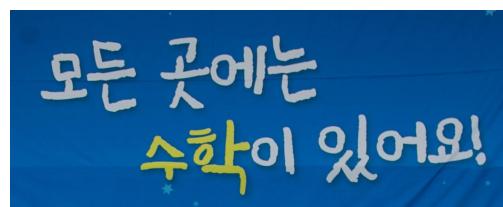
[저자 소개] 이상구 교수 (<http://matrix.skku.ac.kr/sglee/vita/LeeSG.htm>)



(현) 성균관대학교 수학과 교수이다. 성균관대를 졸업하고 미국 유타주립대학교에서 석사(1987년)와 박사학위(1991년)를 취득 했다. 2014년 8월 서울에서 개최한 국제수학자대회(ICM 2014) 조직위원회 문화위원, 국제선형대수학회 (ILAS 2014) 공동조직 위원장, 2012년 서울에서 열린 제12차 국제수학교육대회(ICME-12) 조직위원회 기획위원, 대한수학회 전산수학분과 위원장, 한국교육개발원 영재고교 교육과정위원회 위원, 대입 수능시험 출제위원, BK21 수학적 모델링 HRD 사업단 단장, 국제 선형대수학회(ILAS) 교육위원회 위원, 대한수학회 수학교육분과 위원장, 한국수학 사학회 정보이사, 한국수학교육학회 편집이사, 대한수학회 부회장 등을 역임했으며, 현재 한국수학교육학회 회장과 성균관대 자연대 학장으로 활동하고 있다. 제22회 스승의 날에는 부총리 겸 교육인적자원부장관 표창을 받았다. 행렬론 분야 150여 편의 학술논문과 『Calculus with Sage』, 『현대선형대수학 with Sage』, 『(온드 립 빅북-영문) Linear Algebra』, 『수학과 문화』, 『한국 근대수학의 개척자들』 등 10여권의 저서와 역서 『사회와 수학 – 400년의 파란만장』, 『수학, 형식과 기능』, 『이산수학』를 저술하였다. 한 학기 동안 직접 강의를 하며 본 저서를 저술하고, 모든 코드를 완성하여 책과 함께 무료로 공유하였다.

[도움] 이재화 박사

2002년 한림대학교 수학과 졸업, 2009년 중국과학원에서 최적화이론과 계산수학으로 이학 박사 학위를 받았다. 성균관대학교 BK21 수학적 모델링 HRD 사업단 박사 후 연구원을 거치며 첨단과 전통 수학을 연결하는데 관심을 갖고, 계산수학과 사회수학 분야의 연구를 수행하고 있다. 최근 대학수학교육과 동아시아 수학사에 관한 다수의 논문을 발표하고, 본 저서의 코드를 개발하였다.



... 서 문

인공지능(Artificial Intelligence, AI)은 우리의 곁에 아주 가까이 다가와 있다. 우리는 우리 삶의 거의 모든 곳에서 인공지능이 우리와 대화하고(Siri), 우리가 무엇을 보고 무엇을 구매할지를 추천 해주며(Netflix, Amazon), 가계와 금융에 관하여 조언을 해주고(Schwab의 Intelligent Portfolio), 퀴즈를 풀며, 환자의 병명을 전문가 보다 더 빠르고 정확한 답을 구해주고(IBM의 Watson), 법원의 판례를 찾아주고, 바둑의 고수를 이기는 것을(AlphaLaw, AlphaGo) 보았다. 또한 Google과 Facebook이 딥러닝을 기반으로 음성인식, 얼굴인식 및 무인 자율주행 자동차(Autonomous Vehicle)를 소개하고 있으며, 그 외에도 기계가 우리의 언어로 우리와 의사소통 할 수 있도록, 자연어 이해 및 생성(Quill)을 향상시키는 것을 목표로 하는 일들이 진행 중이다. 인공지능은 추구하고자 하는 목표와 기능에 따라 “인지 컴퓨팅, 스마트 머신, 머신러닝, 자연어 생성, 음성인식, 퓨링 테스트, 로봇비서, 예측 분석, 추천 시스템, 딥러닝, 자율주행 차, 질문–답변 시스템, 자연어 생성 플랫폼 등등 여러 가지 화려한 이름으로 불리고 있다.

우리는 인공지능 로봇이 앞으로 다양하고 힘든 일을 우리 대신 실제로 해줄 것을 이미 잘 알고 있다. 이미 공장, 도로, 심지어 집에서도 로봇이 우리를 위해 일하고 있으며, 미래에 인공지능 로봇이 직장에서의 우리 업무까지 대신하게 되면, 우리는 출근할 이유도 없어 질지 모른다. 따라서 인공지능은 우리에게 죽음, 노예화 또는 영구적인 실업을 줄 가능성이 있다. 그러나 우리는 인공지능에 대해 무작정 두려워하기보다 인공지능이 도대체 무엇인지 그리고 어떻게 작동되는지 기본 원리를 이해 할 필요가 있다. 우리는 이 책에서 학부 및 대학원생에게 인공지능이 작동하도록 만드는 기본 원리에 필요한 수학적 기초 지식을 제공하려고 한다.



2016년 3월 알파고와 이세돌의 바둑 대전에서 이세돌의 승리를 예측하는 사람들이 많았지만 놀랍게도 4대1로 패하고 말았다. 이후 알파고를 비롯한 인공지능 컴퓨터에 대하여 모두가 큰 관심을 갖게 되었다. 아마추어의 기보를 학습하여 아마추어의 수준에 머무는 것이 아니라 이를 출발점으로 시작해서 프로 9단과 인간계 바둑의 한계를 돌파한 것이다. “알파고 안에 어떤 알고리즘이 숨어 있을까?” 인공지능을 처음 연구한 사람은 영국의 수학자 앤런 튜링이라고 알려져 있다. 그는 컴퓨터를 상상하고 '튜링 테스트'라는 가설을 만들었다. 튜링 테스트는 제3자(사람)가 질문을 하고 기계와 사람이 동시에 대답을 할 때, 제3자 입장에서 어느 대답이 기계의 것이고 어느 대답이 사람의 것인지 구분하기 어렵다면, 그 기계는 인공지능을 가지고 있다고 판단해도 된다는 것이다.

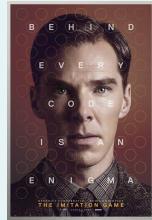
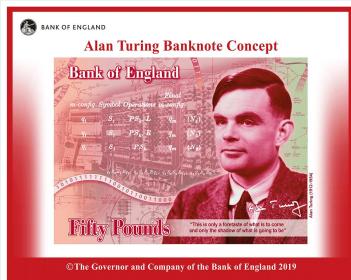
[인공지능과 수학은 무슨 상관이 있을까?] 4차 산업혁명의 핵심 기술요소로 인공지능을 지목하고 있다. 우리는 인공지능이 우리를 위하여 대신 일할 수 있도록 인공지능을 이해하고 활용하기 위한 수학적 이론들에 대해서 생각한다. 이를테면 '선형회귀이론(linear regression)'은 일반적으로 최소제곱법(least square method)을 사용해 선형 회귀 모델을 세운다. 선형회귀이론은 다음과 같은 상황에 적용된다. “한 반에 있는 학생들의 앉은키는 이 학생들의 키와 비례한다. 따라서 직선의 방정식 $y = ax + b$ 꼴로 앉은키와 그냥 키의 관계식을 만들어낼 수 있다.” 물론 더 복잡한 경우의 예로, 주거나 날씨를 시간에 따라 기록하면 그 값이 들쑥날쑥하다. 이런 데이터들을 조금 더 잘 예측할 수 있도록 딥러닝(Deep Learning)이란 기술도 쓰이고 있다. 딥러닝은 인간의 뇌를 흉내 낸 인공신경망을 여러 층으로 분리하여 학습하는 인공지능 기법 중 하나인데, 선형 함수보다 복잡한 비선형 함수를 포함하여 복잡한 데이터를 설명한다. 그 동안 딥러닝이 가장 큰 성과를 거둔 문제로는 음성 인식과 영상 인식(Video-to-video Synthesis)을 들 수 있고, 최근에는 자연어 처리와 이해에서 많은 발전이 이루어지고 있다.

딥러닝, 기계학습, 인공지능은 응용수학 분야라고 할 만큼 수학이 매우 광범위하게 활용되고 있다. 딥러닝에서의 학습은 훈련 데이터가 제시하는 연습 문제를 잘 해결하는 방향으로 신호를 전달하는 연결 지점에서 연결 강도를 변경하면서 이루어지는데, 이때 연결 강도를 어떻게 변경해야 할지를 선형대수학과 다변수 함수의 편미분을 사용하여 구한다. 이 과정에서 다양한 종류의 최적해를 찾는 문제가 등장한다. 딥러닝과 기계학습 그리고 인공지능의 발전을 이해하고 활용하려는 모든 전공의 학생들은 이에 필요한 수학적 기초 지식을 배워야 할 필요가 생겼다.



읽을거리

2019년 7월 영국 50파운드(약 7만4000원) 지폐 뒷면 초상인물로 천재 수학자 앤런 투링(1912~1954)이 선정되었다.



영화 '이미테이션
게임' 포스터

[그림 출처] <https://www.bankofengland.co.uk/banknotes/50-pound-note-nominations>

수학자 투링(Turing)은 인공지능(AI)의 아버지로, 또 전쟁 영웅으로 광범위하면서 선구자적인 기여를 했다. 그는 아카데미 각색상을 받은 영화 ‘이미테이션 게임’의 실제 모델로 화제가 되기도 했다. 투링은 1936년 ‘보편적 기계’ 개념을 창안해 AI 기틀을 마련했으며, 기계가 인간과 얼마나 비슷하게 대화할 수 있는지를 기준으로 ‘기계의 사고 능력’을 판별하는 ‘튜링 테스트’를 만들었다. 영국에서 대학을 졸업한 투링은 1936년부터 1938년까지 미국 프린스턴대 수학과 대학원에서 <서수에 근거한 논리시스템>를 연구하고 수학 박사학위를 받았다. 이때 투링은 체스게임 같이 복잡한 계산 능력을 필요로 하는 연산을 포함하는 많은 계산을 수행할 수 있는 기계는, 숫자와 부호의 이진법으로 작동해야만 한다는 아주 친숙한 개념을 처음으로 발견하였다. 오늘날 우리는 그것을 ‘튜링 머신(Machine)’이라고 부른다. 투링이 수학적이고 논리적인 발견을 바탕으로 착안한 것은 전자두뇌와 인공지능기계였고, 그는 그것을 현실화시켰다. 그는 아울러 제2차 세계대전 당시 독일 잠수함 암호기 ‘에니그마(Enigma)’를 해독했다.

튜링은 2차 세계대전에서 연합군 승리에 기여했지만, 1952년 당시 영국에서 범죄로 취급하던 동성애 혐의로 기소됐다. 수감되지는 않았지만 화학적 거세를 당한 튜링은 1954년 41세에 청산가리를 주입한 사과를 먹고 스스로 목숨을 끊은 것으로 추정된다. 튜링이 베어 먹은 사과가 미국 애플사 로고의 모티브를 제공했다는 설이 있으며, 미국컴퓨터학회(ACM)는 그의 기여를 기념해 1966년 튜링상(Turing Award)을 제정하여, 컴퓨터 과학 분야에 기여한 인사에게 시상하고 있다.

[참고 : <https://www.mk.co.kr/news/society/view/2019/07/528841/> 김덕식 기자, 매일경제]

튜링(Alan Turing, 1912~1954) – 컴퓨터의 아버지, 최초의 해커

<https://www.nature.com/news/turing-at-100-legacy-of-a-universal-mind-1.10065>

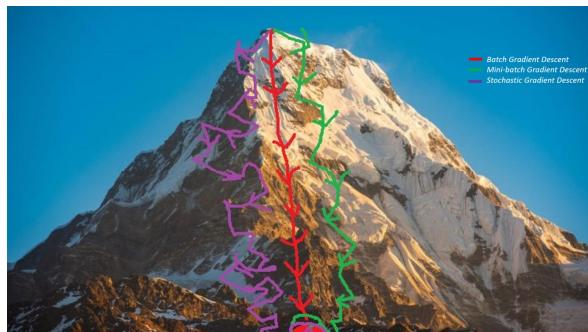
<http://dongascience.donga.com/news.php?idx=-5426053>

[칼럼] 알파고, 인공지능, 그리고 수학 (카이스트 전기 및 전자공학부 김준모 교수)

<https://mathsci.kaist.ac.kr/newsletter/article%ED%8A%B9%EB%83%84%EA%8B%BD%EA%B3%A0-%EC%95%8C%ED%8C%8C%EA%B3%A0-%EC%9D%8B%EA%8B%85%EC%A7%80%EB%8A%A5-%EA%B7%EB%8B%A9%AC%EA%B3%A0-%EC%88%98%ED%95%99/>

[칼럼] 인공지능과 수학은 무슨 상관이 있을까? (서강대 수학과 김종락 교수)

<https://m.post.naver.com/viewer/postView.nhn?volumeNo=11808498&memberNo=36498508>



경사하강법(gradiant descent method) (Batch, Mini-batch, Stochastic)

Imad Dabbura(2017)

<https://towardsdatascience.com/gradient-descent-algorithm-and-its-variants-10f652806a3>

0-1. Python/Sage/R 언어 사용법

1강 동영상, 강의운영 소개 <https://youtu.be/IkQxe1izELM> (32:24)

우리는 이 책에 수록된 연산 문제(CAS)를 해결하기 위해 Python/Sage/R 도구를 사용한다. Python/Sage/R 을 사용하는 SageMath는 Matlab이나 Maple, Mathematica와 같은 고가의 소프트웨어를 따로 설치하지 않아도 (인터넷 접속만 된다면) 언제 어디서든 (크롬 브라우저 등을 이용하여) 사용할 수 있는 수학 연산용 도구이다.

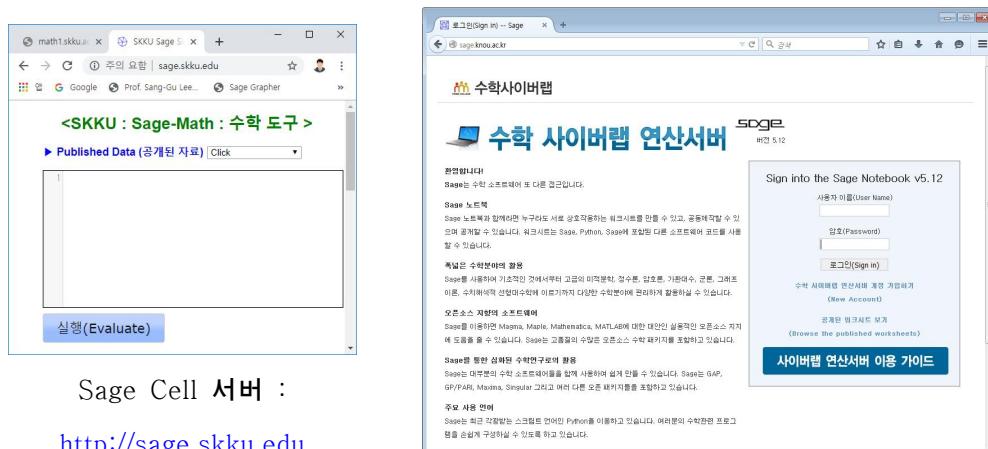
■ Sage Cell 서버

<http://sage.skku.edu> (바로 연결하여 로그인 없이 모바일로 사용가능)

■ Sage 노트북 서버

<https://youtu.be/cx6cx788aHc>

[그림 1]은 본 연구진이 한글화하여 설치한 [성균관대학교 방송통신대 프라임클러스](http://sage.knou.ac.kr)에서 운영 중인 Sage 노트북 서버 (<http://sage.knou.ac.kr>)와 계산용 셀 (<http://mathlab.knou.ac.kr:8080>)이다. 여기에서는 Sage 노트북(Notebook) 서비스를 이용하는 방법을 설명할 것이다. 먼저 이 서버를 사용하기 위해서는 계정이 필요하므로 먼저 계정을 만들고, 메인 화면의 메뉴 및 워크시트 사용법을 살펴보기로 한다. 그런 다음 Sage 명령을 입력하여 결과를 확인하고, 마지막으로 워크시트를 저장하고 종료하는 방법을 살펴볼 것이다.



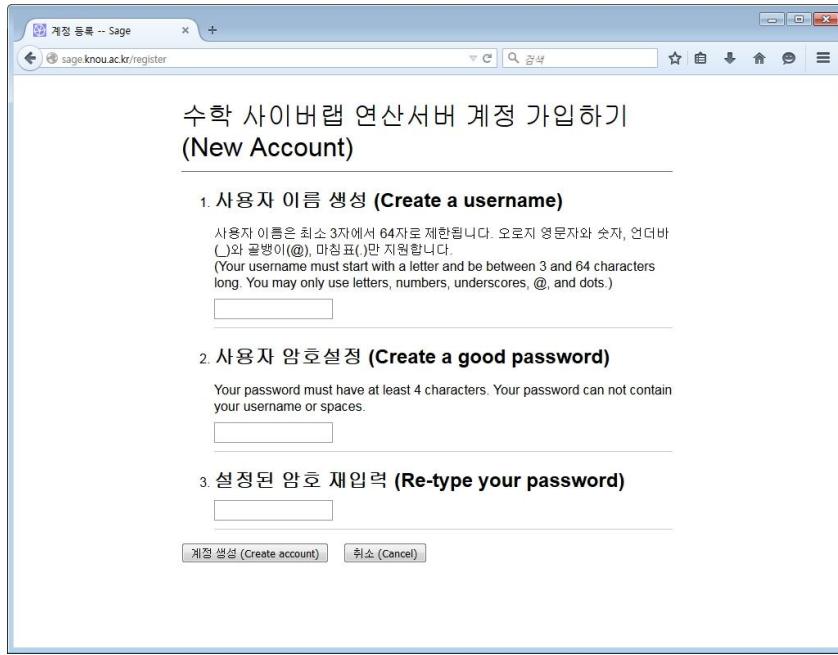
Sage Cell 서버 :

<http://sage.skku.edu>

[그림 1] Sage 노트북 서버

(1) 계정 만들기

- ① SageMath 와 호환성이 좋은 크롬 브라우저 또는 파이어폭스 또는 사파리 브라우저를 설치한다.
 - 크롬(<https://www.google.com/chrome/browser/desktop/index.html>)
 - 파이어폭스(<https://www.mozilla.org/ko/firefox/new/>)
- ② 수학사이버랩 연산서버(<http://sage.knou.ac.kr>)에 접속한 후 [수학 사이버랩 연산서버 계정 가입하기](New Account)] 링크를 클릭하면 [그림 2]와 같은 화면이 나온다.



[그림 2] Sage 노트북 계정 가입을 위한 화면

- ③ 사용자 이름(ID), 사용자 암호, 설정된 암호를 재입력한 후 **계정 생성 (Create account)** 버튼을 누르면 계정이 만들어진다. 이제 생성한 ID, 암호를 이용하여 개인 계정을 이용할 수 있다.

(2) Sage 노트북의 메인 화면 살펴보기

생성한 ID와 암호를 입력한 후 로그인하면 [그림 3]과 같은 화면이 나온다.



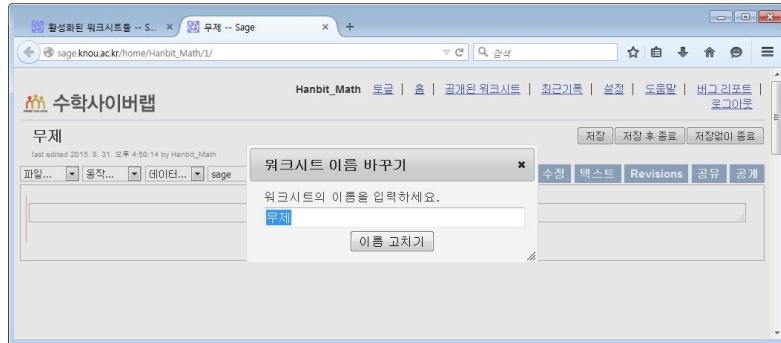
[그림 3] Sage 노트북 메인 화면

메인 화면에 보이는 메뉴 중에서 자주 쓰이는 항목은 다음과 같다.

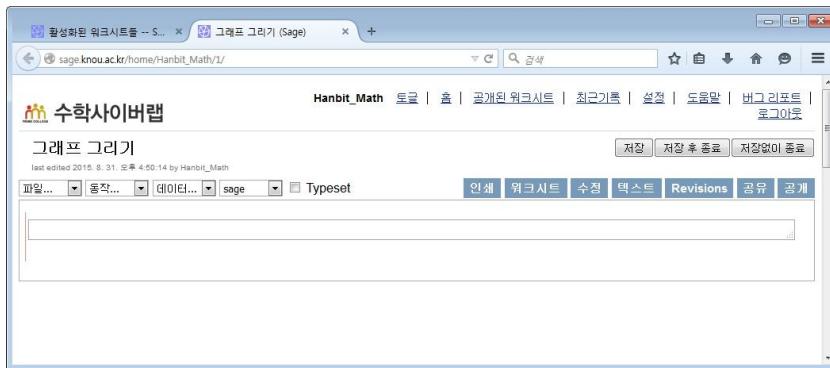
- ① 홈 : Sage 노트북 메인 화면으로 돌아가기
- ② 공개된 워크시트 : 공개된 워크시트 확인하기
- ③ 설정 : 자동 저장간격 및 사용자 계정 정보 수정하기
- ④ 로그아웃 : Sage 노트북 종료하기
- ⑤ 새 워크시트 : 새로운 워크시트 작성
- ⑥ 업로드 : Sage 워크시트 파일(sws 형식) 업로드하기
- ⑦ **다운로드** : 선택한 Sage 워크시트 다운로드(sws 형식)

(3) Sage 노트북의 워크시트 화면

새로운 워크시트를 작성하기 위해 <새 워크시트>를 누르면 [그림 4]와 같이 워크시트 이름을 입력하는 창이 나타난다. 적당한 워크시트 이름을 입력한 후 **이름 고치기** 버튼을 클릭하면 [그림 5]와 같이 Sage 명령어를 입력하여 실행시킬 수 있는 Sage 노트북 워크시트 화면을 확인할 수 있다.



[그림 4] Sage 노트북 워크시트 이름 입력하기



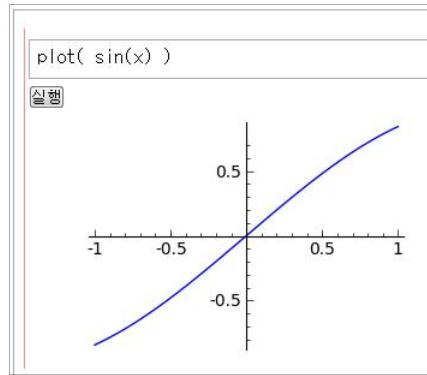
[그림 5] Sage 노트북 워크시트 화면

워크시트 화면에 보이는 메뉴 중에서 자주 쓰이는 항목은 다음과 같다.

- ❶ **저장** **저장 후 종료** **저장없이 종료**: 워크시트 저장, 워크시트 저장 후 종료, 워크시트 저장을 하지 않은 채로 종료
- ❷ **파일...** : sws 파일 업로드, 새 워크시트 작성, 현재 워크시트 다운로드, 인쇄, 워크시트 이름 바꾸기, 워크시트 복사, 워크시트 삭제
- ❸ **동작...** : 연산 강제 종료, 워크시트 재시작, 워크시트 저장 후 종료, Sage 명령어 전부 실행, 결과 숨기기, 결과 보이기, 결과 지우기
- ❹ **데이터...** : Sage 노트북 서버에 활용할 각종 파일 업로드 기능
- ❺ **sage** : Sage, GAP, GP, Python, R 명령어 실행 설정
- ❻ **공개**: 현재 워크시트를 html 형식으로 공개

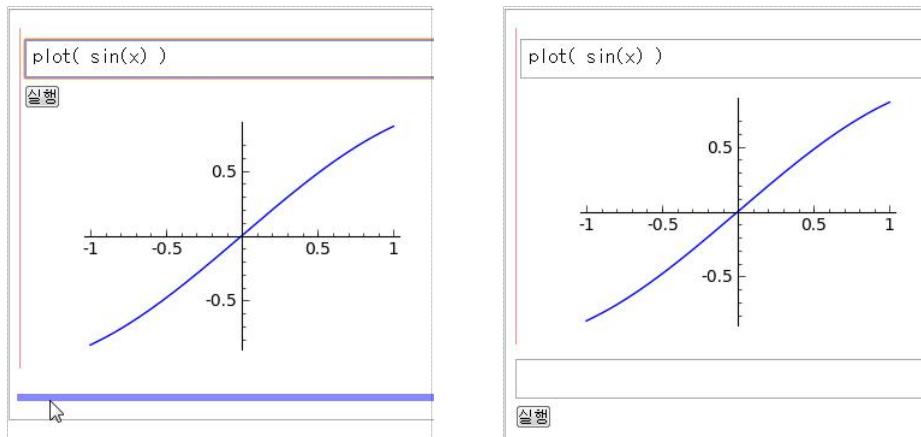
(4) Sage 명령어의 입력 및 실행

[그림 6]과 같이 빈 칸에 Sage 명령어를 입력하고 **실행** 버튼을 누르면 그 결과를 확인할 수 있다.



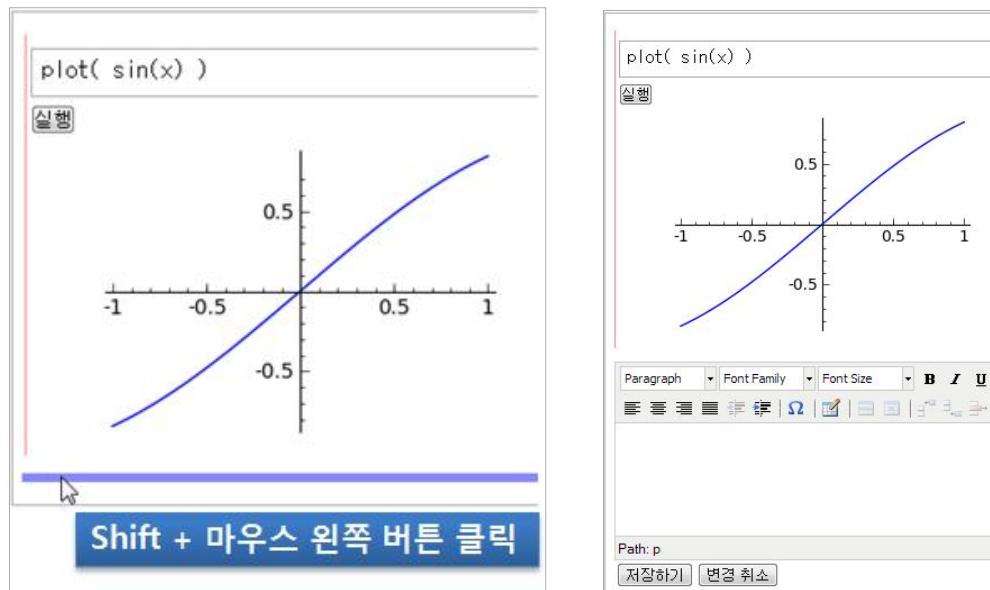
[그림 6] Sage 명령어 입력 및 실행 결과

이때 결과화면 아래쪽으로 마우스를 가져가면 [그림 7]과 같이 파란색 선을 확인할 수 있다. 이 선을 마우스로 클릭하면 Sage 명령어를 입력할 수 있는 새로운 셀(Cell)을 추가할 수 있다.



[그림 7] 새로운 Sage 셀 추가하기

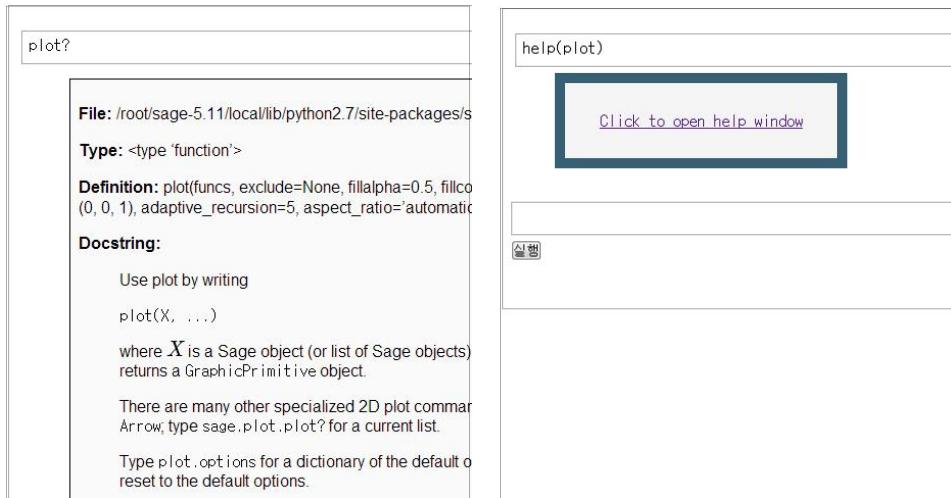
또는 **Shift** 버튼을 누른 채 마우스의 왼쪽 버튼으로 파란색 선을 클릭하면 [그림 8]과 같이 HTML 설명을 추가할 수 있는 텍스트 박스를 열 수 있다. 이 텍스트 박스에서는 다양한 서식과 수식을 이용하여 결과 화면에 설명을 추가할 수 있다. **저장하기** 버튼을 클릭하면 입력한 설명들이 화면에 나타나며, 그 설명을 마우스로 더블클릭하면 다시 텍스트 박스를 열어 내용들을 수정할 수 있다.



[그림 8] HTML 설명을 추가할 수 있는 텍스트 박스

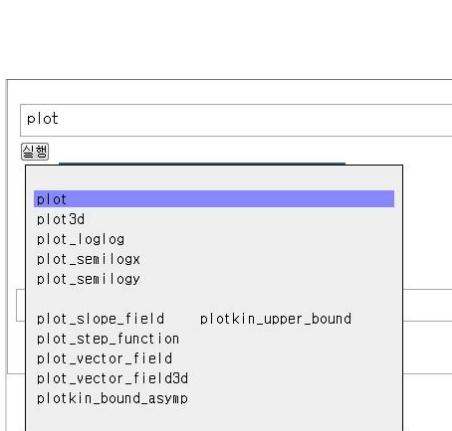
(5) Sage 명령어 도움말 사용하기

특정 명령어의 사용법을 알고 싶을 때는 명령어 뒤에 “?”를 붙이거나 “help(Sage 명령어)” 입력하고 **실행** 버튼을 누르면 [그림 9]와 같이 자세한 사용 방법을 보여준다.

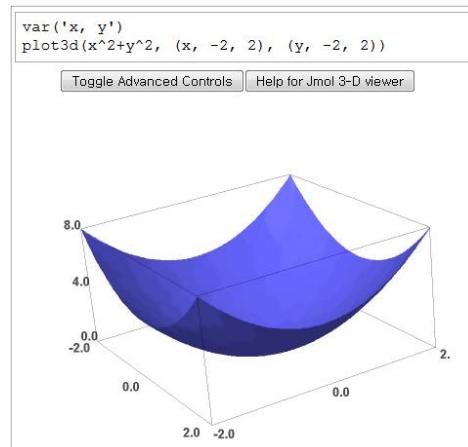


[그림 9] Sage 명령어 류토리얼

또한 길이가 긴 Sage 명령어를 입력할 때 혹은 명령어 사용방법이 잘 떠오르지 않는 경우에는 명령어 자동완성 기능을 사용하기를 권한다. 예를 들어, “plot”을 입력한 후 <TAB> 키를 누르면 [그림 10]과 같이 “plot”으로 시작하는 모든 Sage 명령어를 확인할 수 있다.



[그림 10] Sage 명령어 자동완성 기능



[그림 11] 3D 그래프 예시

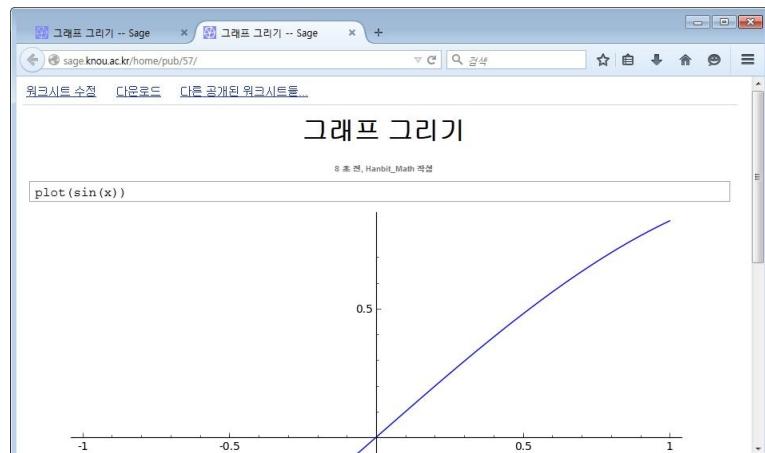
3차원 이미지 관련 명령어는 <https://sagecell.sagemath.org/>에서 실행하면 된다([그림 11] 참조)

(5) 워크시트 저장 및 공개, 종료

앞서 작성한 Sage 워크시트는 **[저장]** **[저장 후 종료]** **[저장없이 종료]** 버튼들을 활용하여 저장하거나 종료시킬 수 있다. 또한 **[공개]** 버튼을 클릭하여([그림 12] 참조) 해당 내용을 [그림 13]과 같이 HTML 형식으로 웹상에 공개할 수도 있다. 이렇게 공개된 자료는 별도의 Sage 노트북 서버에 로그인을 하지 않아도 웹을 통해 확인할 수 있다. (참고로 수업을 통해 작성 및 공개된 워크시트를 <http://math1.skku.ac.kr/pub/>에서 확인하고, 복사하여 사용할 수 있다.)



[그림 12] 워크시트 공개하기



[그림 13] 웹 상에 공개된 워크시트

■ 예제 및 연습문제 실습실

이 책의 본문에는 CAS 예제 및 연습문제에 대해 코드-실행결과를 보여주고 있다. 실제로 학습자들이 코드를 실행해보고, 필요에 따라 수정해볼 수 있도록 본문에 수록된 CAS 예제 및 연습문제 실습실을 구성했다. 실습실을 활용하는 방법은 Sage 노트북과 같지만, 별도로 계정을 만들거나 로그인할 필요는 없다. 본문의 코드가 기본적으로 입력되어 있어 학습자의 시간을 절약할 수 있게 했다.

[기본] 이미 주어진 실습 문제의 코드를 살펴본 후, [실행(Evaluate)] 버튼을 클릭하여 결과를 확인한다.

예제 1-10 : 주어진 벡터들의 합함이 일차종속 합함임을 보여라.

```
1 A=matrix(3, 3, [1, 3, 2, 0, -2, -2, -1, 0, 1]) # 계수행렬
2 print A.rref()
```

실행(Evaluate) Language: Sage ▾
Syntax Highlighting
Permalink, Shortened Temporary Link
Powered by SAGE

```
[ 1 0 -1]
[ 0 1 1]
[ 0 0 0]
```

[그림 14] 실습실 활용하기 ①

[활용] 새로운 문제에 맞게 숫자, 함수 또는 코드를 수정하고, [실행(Evaluate)] 버튼을 클릭하면 새로운 결과를 얻는다. 예를 들어, ① 미분방정식 또는 ② 초기 조건을 수정하여 실습할 수 있다.

[Sage 고정] <http://sage.skku.edu> 또는 <http://mathlab.knou.ac.kr:8080>

```
1 # ① 특성방정식 풀이 ①
2 y = var('y')
3 solve(y^2 + 2*y + 2==0, y)
```

실행(Evaluate) Language: Sage ▾
Syntax Highlighting
Permalink, Shortened Temporary Link
Powered by SAGE

```
[y == (-I - 1), y == (I - 1)]
```



```
1 # ② 초기조건 미분방정식 풀이 ②
2 x = var('x')
3 y = function('y', x)
4 de = desolve(diff(y, x, 2) + 2*diff(y, x) + 2*y ==0, [y, x], [0, 3, -5])
5 print de
```

실행(Evaluate) Language: Sage ▾
Syntax Highlighting
Permalink, Shortened Temporary Link
Powered by SAGE

```
-(2*sin(x) - 3*cos(x))*e^(-x)
```

[그림 14] 실습실 활용하기 ② ■

<SageMath vs Python>

SageMath는 무료(free), 공개(open-source) 소프트웨어로 IPython, LAPACK, Maxima, Numpy, Matplotlib, R 등을 모두 커버한다. Python 언어 기반인 SageMath는 Python을 개량한 Extension으로 볼 수 있다.

SageMath는 사용하기 편리한데, 예를 들어 $\{3x - 2 \mid x \in \{0, 1, \dots, 9\} \text{ and } x \text{ is odd}\}$ 는 다음과 같이 표현된다.

[SageMath]

```
[3*x - 2 for x in range(0, 10) if x%2 == 1]
```

[1, 7, 13, 19, 25]

아래는 SageMath를 사용한 프로그래밍 예시이다.

```
S = 0 ; n = 10
for k in [1..n]:
    S = S + (2*k) * (2*k+1)
S
```

1650

② while loop를 사용하는 방법

```
U = 2.0; V = 50.0
while V-U >= 1.0e-6: # 1.0e-6 stands for 1.0*10^-6
    temp = U
    U = 2 * U * V / (U + V)
    V = (temp + V) / 2
U, V
```

(9.9999999989256, 10.0000000001074)

③ function을 정의하는 법

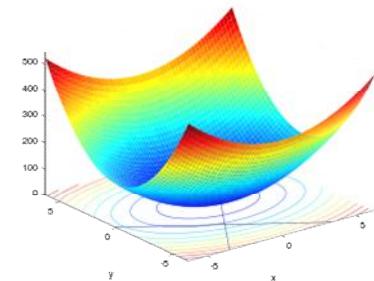
```
def foo (u):
    t = u^2
    return t*(t+1)
t = 1 ; u = 2
foo(3), t, u
```

(90, 1, 2) ■

< SageMath에 관한 자세한 내용은 아래 자료를 참고하라. >

[출처]

- [1] <http://lacim.uqam.ca/~saliola/sage/MiniCourse/Lecture1IntroToSage.pdf>
- [2] Chapter 3 of the book “*Computational Mathematics with SageMath*” written by P. Zimmermann et al, 2018. (download)
<http://dl.lateralis.org/public/sagebook/sagebook-ba6596d.pdf>
- [3] <http://doc.sagemath.org/html/en/tutorial/programming.html> ■



$$H(f) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \dots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \dots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Hessian 행렬과 Local minima(최소값)

0-2. 수학과 코딩! [수학동아 2019년 연재]

<http://matrix.skku.ac.kr/2018-album/Math-Coding.htm>



내용 감수 : 성균관대학교 이상구 자연대 학장 (現 한국수학교육학회장)

< 수학동아 Math & Coding >

<http://dl.dongascience.com/magazine/view/M201901N012>

[Junior Polymath] 주니어 폴리매스

[순열과 조합] <http://www.polymath.co.kr/mini/1330>

in http://www.polymath.co.kr/mini?category_id=15&ready_focus=newlist

글 : 이재윤 연구원(toed00 at gmail.com)

여러분들은 원의 넓이를 구하는 공식을 잘 알고 있을 거예요. 그런데 적분이 원의 넓이를 구하는 방법에서 출발하였다는 사실, 알고 계신가요?

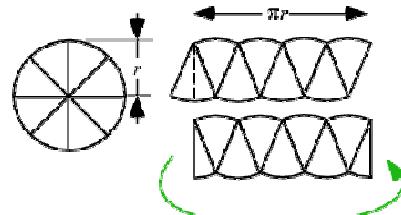
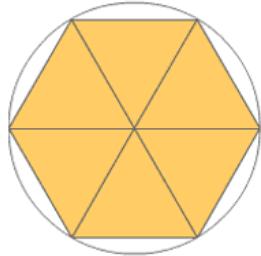
먼저, 우리가 현재 알고 있는 원의 넓이를 구하는 공식인 πr^2 은 고대 그리스 시대의 수학자 아르키메데스에게서 시작되었어요. 그는 정다각형이 충분히 많은 변을 가질수록 원에 가까워진다는 점을 생각해 내었는데, 놀랍게도 이를 이용하여 원주율 π 의 매우 정밀한 근사값인 3.1416을 계산해 내었을 뿐만 아니라 원의 넓이를 구하는 공식인 πr^2 를 수학적으로 증명하였답니다.



이후에 르네상스 시대의 유명한 화가이자 과학자인 레오나르도 다빈치는 아르키메데스의 방법을 좀 더 쉽게 이해할 수 있도록 설명하였어요. 원을 충분히 많은 조각으로 쪼갠 다음, 옆의 그림과 같이 엇갈려 이어붙이면 밑변 πr 그리고 높이 r 인 직사각형으로 만들 수 있어요. 그러면 이 직사각형의 넓이는 [밑변] \times [높이]인 πr^2 이 됩니다.



레오나르도 다빈치

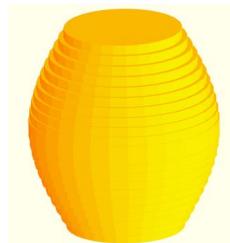


출처: Wolfram Mathworld



케플러

그렇다면, 현재 우리가 사용하고 있는 적분의 개념은 언제 등장하였을까요? 17세기 수학자이자 천문학자인 케플러는 아르키메데스의 방법에서 아이디어를 얻어, 포도주통을 원판 (Disk)들로 채워서 계산하였는데 이는 현재 우리가 사용하고 있는 적분의 개념과 동일해요.



출처: Matematicas Visuales

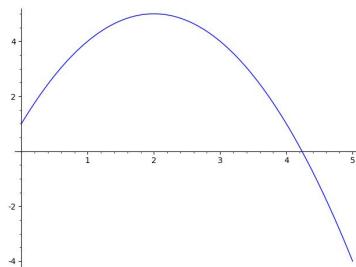
[케플러가 살던 시기, 포도주의 가격은 포도주 통 안에 막대를 넣어 포도주가 채워져 있는 높이를 재서 결정하였는데, 중간이 볼록한 포도주 통의 모양 때문에 통에 채워진 포도주의 높이와 실제 양이 정확히 비례하지 않았음]

적분의 기본개념을 이해하였으니, 이제 간단한 코딩으로 수학문제를 쉽게 해결해 봅시다.

(Code) 수학 코딩해보기

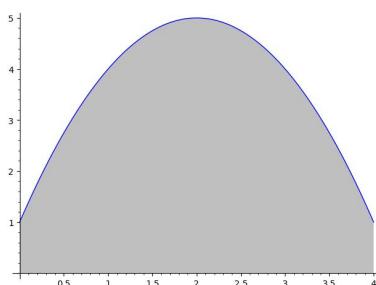
① 구간 $(0, 5)$ 에서 함수 $f(x) = -x^2 + 4x + 1$ 의 그래프를 그려보자.

```
var('x')
f = -x^2+4*x+1
plot(f, x, (0,5))      #구간: 0부터 5까지, 함수: -x^2+4*x+1
```



② 구간 $(0, 4)$ 에서 함수 $f(x) = -x^2 + 4x + 1$ 의 그래프와 x 축이 만드는 영역을 살펴보자.

```
var('x')
f = -x^2+4*x+1
plot(f, x, (0,4), fill="axis")
```



③ 적분 명령어를 이용하여 $(0, 4)$ 구간에서 $f(x) = -x^2 + 4x + 1$ 의 그래프와 x 축이 이

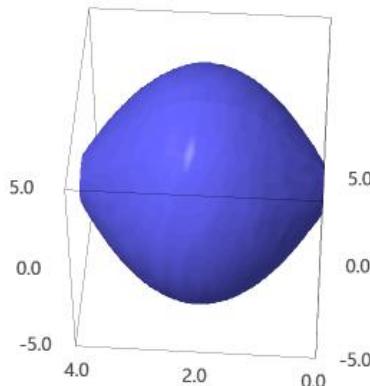
루는 넓이, 즉 적분값을 구하면?

```
var('x')
f = -x^2+4*x+1
integral(f, x, 0, 4)
```

계산값: 44/3

- ④ 구간 $(0, 4)$ 에서 함수 $f(x) = -x^2 + 4x + 1$ 의 그래프를 x 축에 대하여 회전시킨 회전체를 그려보자. <https://sagecell.sagemath.org/>

```
var('x')
f = -x^2+4*x+1
revolution_plot3d(f,(x,0,4), parallel_axis='x')
```



- ⑤ 앞에서 그린 회전체의 부피를, 적분을 통해 구해보자.

```
var('x')
f = -x^2+4*x+1
integral(pi*f^2, x, 0, 4)      # pi는 원주율π
```

계산값: $892/15\pi$

수식을 살펴보면, 구간 $(0, 4)$ 에서 원의 넓이를 구하는 공식 πr^2 을 이용하여 계산하였다,

즉, 케플러가 제시한 방법과 개념적으로 일치한다.

⑥ 회전체와 적분값을 같이 살펴보자. <https://sagecell.sagemath.org/>

```
var('x')
f = -x^2+4*x+1
A=revolution_plot3d(f,(x,0,4), parallel_axis='x')
B=integral(pi*f^2, x, 0, 4)      # pi는 원주율
show(A)
show(B)
```

계산값: $892/15\pi$

행렬과 연립방정식

이번에 코딩으로 정복해 볼 수학 개념은 ‘1차 연립방정식’입니다. 1차 연립방정식은 미지수가 2개 이상인 1차 방정식의 모임을 뜻해요. 보통 미지수가 하나인 1차 방정식은 등식의 성질을 이용해 쉽게 해를 구할 수 있지만, 연립방정식은 방정식을 서로 더하거나 빼서 미지수를 하나로 만든 뒤 해를 구하는 ‘소거법’으로 해를 구하지요.

소거법으로 미지수가 무척 많은 연립방정식의 해를 구하려면 시간이 오래 걸리기 때문에 보통 ‘행렬’을 이용합니다. 행렬은 수를 행과 열에 맞춰 나열해 놓은 것으로, 특히 행 또는 열 1개로 이뤄진 행렬을 ‘벡터’라고 불러요.

연립방정식을 구성하는 식은 모두 같은 미지수를 가졌기 때문에 미지수 앞에 붙어있는 계수와 등호 오른쪽에 있는 상수가 해를 결정합니다. 독일 수학자 가우스는 연립방정식의 계수를 모아 ‘계수 행렬’을 만들고 상수를 모아 ‘상수 벡터’로 만든 뒤, 직접 만든 행렬 전용 소거법인 ‘가우스 소거법’을 이용해 해를 구했습니다. 놀라운 사실은 가우스보다 앞서 동양의 오래된 수학책인 ‘구장산술’에서 행렬을 이용해 해를 구하는 방법을 소개했다는 건데요. 구장산술에서는 행렬을 ‘방정’이라고 불렀고, 가우스와 똑같은 방법으로 연립방정식의 해를 구했습니다. 지금 ‘방정식’이라는 용어가 이곳에서 유래한 거지요.

가우스 소거법은 쉽지만 지루한 긴 과정을 거쳐야 하지만, 그 절차를 코드로 담은 코딩 명령어를 이용하면 계수 행렬과 상수 벡터만 입력해서 쉽게 해를 구할 수 있습니다. 자, 그럼 구장산술에 수록된 미지수가 3개인 연립방정식을 소거법과 코딩, 두 가지 방법으로 풀어보도록 하죠!

<p><구장산술 방정 문제></p> <p>소 2마리와 양 5마리를 팔아 돼지 13마리를 사면 1000전이 남고(⑦), 소 3마리와 돼지 3마리를 팔아 양 9마리를 사면 돈이 남지 않으며(⑧) 양 6마리와 돼지 8마리를 팔아 소 5마리를 사면 600전이 모자란다(⑨). 소, 양, 돼지는 각각 얼마일까?</p>	<p>❶연립방정식 만들기 소, 양, 돼지의 값을 각각 x, y, z로 놓고 식을 세운다. $2x+5y-13z = 1000 \dots ⑦$ $3x-9y+3z = 0 \dots ⑧$ $-5x+6y+8z = -600 \dots ⑨$</p>	<p>❷⑧을 x에 대해 정리한 뒤 ⑦에 대입 $x=3y-z$ $2(3y-z)+5y-13z = 1000$ $\rightarrow 11y-15z = 1000$</p>
<p>❸⑧을 x에 대해 정리한 뒤 ⑨에 대입 $x=3y-z$ $-5(3y-z)+6y+8z=-600$ $\rightarrow -9y+13z=-600$</p>	<p>❹해 구하기 비슷한 방법으로 ❷, ❸의 식에서 y를 제거하고 해를 구한다. $\therefore z=300, y=500, x=1200$</p>	

코딩으로 연립방정식 정복하기

연립방정식을 푸는 코딩 명령어는 다음과 같다. 코딩 수학 계시물에 있는 Sage 코딩창 또는 sage.skku.edu의 코딩창에 아래 명령어를 입력해 연립방정식의 해을 구해보자!

```
a1x + b1y + c1z = d1      A=matrix([[a1, b1, c1], [a2, b2, c2], [a3, b3, c3]])
a2x + b2y + c2z = d2      ⇔ B=vector([d1, d2, d3])
a3x + b3y + c3z = d3      A.solve_right(B)
```

- ❶** 각 행에 있는 계수로 행렬을 만들고, 상수항은 행 1개로 이뤄진 행렬인 ‘벡터’로 지정하자. 각 수는 [와] 안에 순서대로 적어야 한다.

1	A=matrix([[2, 5, -13], [3, -9, 3], [-5, 6, 8]])	2x + 5y - 13z = 1000
2	B=vector([1000, 0, -600])	3x - 9y + 3z = 0
		-5x + 6y + 8z = -600

Evaluate

sage.skku.edu

- ❷** 연립방정식을 푸는 코딩 명령어를 입력했으면 Evaluate 버튼을 클릭! 정답은 x=1200, y=500, z=300이다.

```

1 A=matrix([[2, 5, -13], [3, -9, 3], [-5, 6, 8]])
2 B=vector([1000, 0, -600])
3 A.solve_right(B) # 방정식의 해를 구하는 명령어

```

Evaluate

(1200, 500, 300) # 답: x=1200, y=500, z=300

<http://sage.skku.edu>

경우의 수를 쉽게 계산하는 ‘순열과 조합’

이번에 코딩으로 정복해 볼 수학 개념은 ‘순열과 조합’이에요.

순열(Permutation)은 쉽게 말해, 순서대로 나열하는 경우의 수입니다. 예를 들어 A, B, C가 쓰인 카드 세장을 순서대로 나열한다면 { (A, B, C), (A, C, B), (B, A, C), (B, C, A), (C, A, B), (C, B, A) } 이렇게 총 6가지 경우가 나옵니다.

그런데 이렇게 순서대로 나열하는 경우를 일일이 찾아서 세지 않고 ‘팩토리얼’로 쉽게 계산할 수 있어요. 팩토리얼($n!$)은 자연수 n 부터 1까지의 모든 수를 곱하는 것입니다. 앞에서 살펴본, 서로 다른 카드 세장을 순서대로 나열하는 경우의 수는 $3! = 3 \times 2 \times 1 = 6$ 입니다.

또한, 서로 다른 n 개 중 k 개를 선택하여 순서대로 나열하는 경우의 수는 ${}_n P_k$
 $= \frac{n!}{(n-k)!}$ 로 쉽게 계산할 수 있습니다.

조합(Combination)은, 순서와 상관없이 선택하는 경우의 수입니다. 예를 들어 A, B, C 가 쓰인 카드 3장에서 순서와 상관없이 두 장을 뽑는다면, { (A, B), (A, C) (B, C) } 이렇게 총 3가지 경우가 나옵니다. 서로 다른 n 개 중 k 개를 순서와 상관없이 선택하는 경우의 수는 ${}_n C_k = \frac{n\text{개 중 } k\text{개를 선택하는 순열}}{k\text{개를 순서대로 나열하는 경우의 수}} = \frac{{}_n P_k}{k!} = \frac{n!}{k!(n-k)!}$ (단, $n \geq r$) 입니다.



경우의 수를 찾는 방법을 순열과 조합을 통해 체계적으로 제시한 사람은 17세기에 활동한 프랑스 수학자 파스칼(Pascal)입니다. 파스칼은 친구인 메레(de Mere, 1607~1684)에게, 게임을 하다가 중단되었을 때 판돈을 어떻게 분배해야하는지에 대하여 질문을 받았습니다. 이때 파스칼은 남아있는 게임의 수와 이기는 데 필요한 게임의 수를 이용하여, 승률에 대한 기댓값에 맞추어 판돈을 분배하는 해답을 제시하였습니다.

블레즈 파스칼(1623~1662) 그리고 이 내용을 친구인 페르마(Fermat)에게 소개하면서 공식적으로 세상에 알려지게 되었습니다.

순열(Permutations) 과 조합(Combinations)

정리. 순열(Permutations), 분류

(1) 서로 다른 n 개에서 k 개를 순서대로 순열의 수는 ${}_n P_k = \frac{n!}{(n-k)!}$ 이다. 특히 $k=n$ 일 때 ${}_n P_n = n! = 1 \cdot 2 \cdot 3 \cdots n$ 이다.

(2) n 개의 주어진 물건을 c 개의 그룹으로 분류하고자 한다. 방법의 개수는 다음과 같다. $\frac{n!}{n_1! n_2! \cdots n_c!} \quad n = n_1 + n_2 + \cdots + n_c$

여기서 n_j 는 j 번째 그룹에 있는 물건의 개수이다.

예제. 박스 안에 뺄간 공 6개와 파란 공 4개가 들어 있다. 처음 뺄간 공을 꺼내고, 두 번째 파란 공을 꺼내 확률은 다음과 같다.

$$P = \frac{6! 4!}{10!} = \frac{1}{210}.$$

[Sage code]

```
print(factorial(6)*factorial(4)/factorial(10))
```

1/210

정리. 조합(Combinations), 중복조합

(1) 서로 다른 n 개에서 중복없이 k 개를 택하는 방법의 수는 다음과 같다.

$$\binom{n}{k} = \frac{n!}{k! (n-k)!}$$

(2) 서로 다른 n 개에서 중복을 허락하여 k 개를 택하는 방법의 수는 다음과 같다.

$$\binom{n+k-1}{k}$$

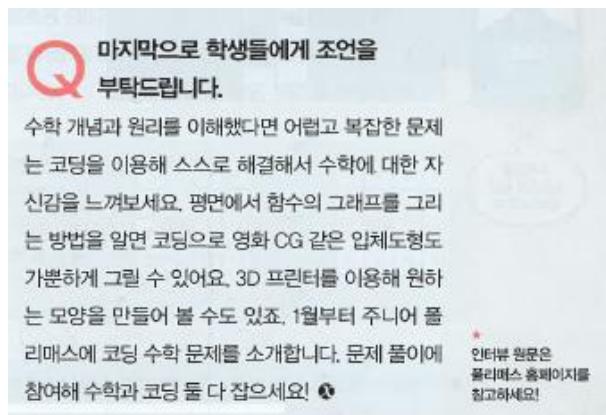
예제. 500개의 전구로부터 5개의 백열전구를 택하는 방법의 개수는 $\binom{500}{5} = \frac{500!}{5! 495!}$ 이다.

[Sage code]

```
print(binomial(500, 5))
```

255244687600

<http://www.polymath.co.kr/mini/1330>



Q 마지막으로 학생들에게 조언을 부탁드립니다.

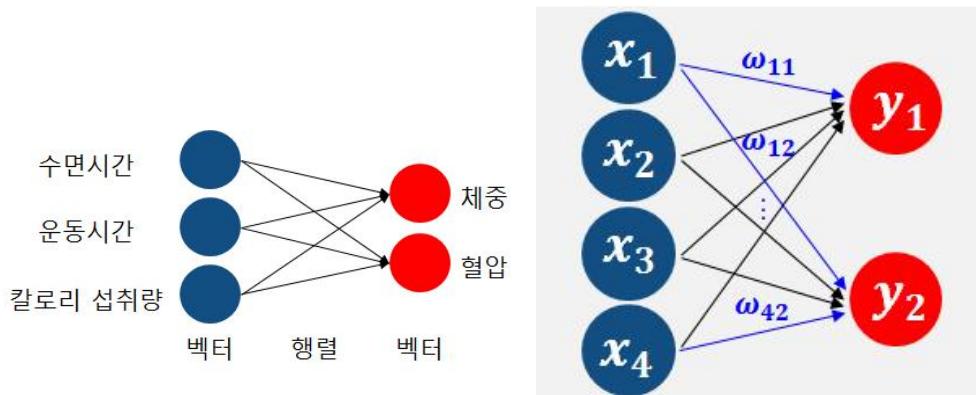
수학 개념과 원리를 이해했다면 어렵고 복잡한 문제는 코딩을 이용해 스스로 해결해서 수학에 대한 자신감을 느껴보세요. 평면에서 함수의 그래프를 그리는 방법을 알면 코딩으로 영화 CG 같은 입체도형도 가뿐하게 그릴 수 있어요. 3D 프린터를 이용해 원하는 모양을 만들어 볼 수도 있죠. 1월부터 주니어 플리마스에 코딩 수학 문제를 소개합니다. 문제 풀이에 참여해 수학과 코딩 둘 다 잡으세요! ☺

* 인터뷰 원문은 [플리마스 홈페이지](#)를 참고하세요!

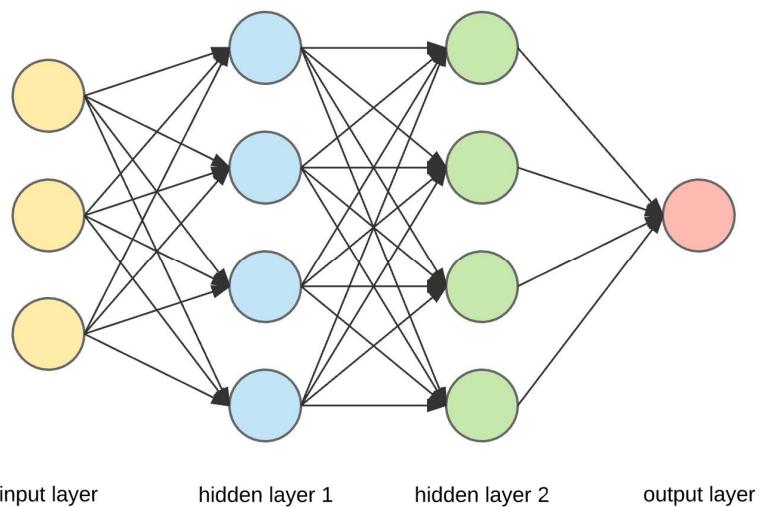
■ 2019년 10월 28일, 정부는 인공지능에 대한 기본구상을 바탕으로 인공지능 국가전략을 제시하고, 이어서 정부 스스로가 가장 적극적으로 인공지능을 활용·지원하겠다고 밝혔다. 인공지능이란 사람들이 일반적으로 하는 것, 특히 지능적으로 행동하는 사람들이 하는 일을 대신 할 수 있는 컴퓨터의 개발을 목표로 하는 컴퓨터 과학(Computer Science)의 하위 분야이다. 1956년 여름에 있었던 "다트머스 컨퍼런스"에서 인용된 정의에서 출발하면, 사람들이 지능적인 행동이라고 생각할 만한 일들을 대신해 주는 어떤 소프트웨어(프로그램)라도 우리는 그것을 AI 시스템으로 볼 수 있다. 60년 전에 논의된 AI가 요즘 다시 부각되는 이유는, 60년 전에 생각했던 것보다 그런 발전을 실제로 이루어 내는 것이 쉽지 않았기 때문이다. 최근에 와서 데이터가 양적으로 증가하고, 최신의 알고리즘과, 컴퓨팅 파워로 인하여 AI의 활용이 활발히 진행되고 있다.

■ 인공지능은 컴퓨터가 마치 인간의 뇌와 같이 작동하여 이미지를 인식하거나 자율주행과 같은 업무를 수행하도록 한다는 개념이다. 기계학습(머신러닝)은 인공지능의 일부이다. 지능을 기반으로 하는 프로세스 자체는 블랙박스가 아니며, 지능형 시스템의 구성은 생각보다 단순하며, 우리가 이해할 수 있는 이론과 절차에 따라 만들어진 것이라는 것을 이해하는 것이 가장 중요하다. AI는 마술이 아니라 데이터, 규모 및 처리 능력에 의해 구동되는 알고리즘 모음을 적용하는 것이다. Amazon이 당신에게 책을 추천해줄 때, 그 추천 뒤에 숨은 AI 시스템을 이해하는 것이 중요하다. Amazon은 당신의 구매 패턴에 대한 정보를 모으고, 당신이 누구인지 그리고 당신이 다른 사람들과 얼마나 비슷한지를 찾아낸다. 그리고 당신과 비슷한 사람들이 좋아하는 것을 바탕으로, 당신에게 새로운 상품을 추천해준다. 즉 우리를 포함한 모든 지능형 시스템의 경우, 현재 진행 중인 상황을 이해하고 이로 부터 추론하여, 다음에 어떤 일이 일어날 지 예측한다. 이런 예측 분석(Predictive Analytics) 작업은 회귀 분석(Regression Analysis)이나 시계열 분석(Time Series Analysis)과 같은 통계 기법을 통해 주어진 과거의 행태(Behavior)를 이용하여 구매활동의 미래 경로를 추측한다. 최근에는 딥러닝과 같은 기계학습 이론을

사용하는 경우가 많다. 이 과정에서 사용되는 수학인 선형대수학과, 미적분학, 그리고 통계학의 기초를 이론과 함께 파이썬과 R 언어 기반의 코딩을 이용하여 실제 실습해 보면서, 인공지능에 활용되는 아래와 같은 기초수학을 본 교재에서 다룬다.



$$\mathbf{y} = \mathbf{A} \mathbf{x} \quad \text{또는} \quad \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix} \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \\ w_{41} & w_{42} \end{bmatrix} = \begin{bmatrix} y_1 & y_2 \end{bmatrix}$$



[참고문헌] Kristian Hammond, Practical Artificial Intelligence for Dummies, John Wiley & Sons, 2015. <Dummy를 위한 인공지능> <https://youtu.be/VZbv6BG-xIY>

1. 거듭제곱법(Power Method)

참고 동영상: <https://youtu.be/CLxjkZuNjXw>

- n 차 정사각행렬은 복소수 범위에서 n 개의 고윳값을 갖는다. 그러나 행렬이 조금만 커지면 정확한 고윳값들을 모두 구하는 것은 힘들어진다. 따라서 크기가 큰 행렬의 경우에 모든 고윳값을 찾기보다는 가장 큰 고윳값만을 손쉽게 찾는 새로운 방법을 연구해 낸 것이 바로 행렬의 거듭제곱을 이용하는 거듭제곱법(Power Method)이다. 구글 검색엔진에서 PageRank를 구할 때도 사용된다. 또 PCA에서 크기가 큰 공분산 행렬의 고윳값을 구하는 데도 활용한다.

예제 1 행렬 A 가 다음과 같다고 하자.

$$A = \begin{bmatrix} 1 & 2 & 0 \\ 2 & 1 & 2 \\ 1 & 2 & 3 \end{bmatrix}$$

$\mathbf{x} = (1, 1, 1)$ 로 하여 Sage를 이용하여 행렬의 충분히 큰 (약 10 정도) 거듭제곱을 구하면 다음과 같다.

[Sage code] <http://mathlab.knou.ac.kr:8080/> <http://sage.skku.edu/> <https://sagecell.sagemath.org/>

```
A = matrix(3, 3, [1, 2, 0, 2, 1, 2, 1, 2, 3]) # A 입력
x = vector([1, 1, 1]) # 벡터 x 입력
K10 = (A^10)*x # K10은 A의 10 거듭제곱에 열벡터를 곱한 벡터
K11 = (A^11)*x # K11은 A의 11 거듭제곱에 열벡터를 곱한 벡터
print("A^10x =", K10)
print("A^11x =", K11)
# 우세한(dominant) 고윳값은 K10, K11 각각의 첫 번째 성분을
# 나눈 값으로 구할 수 있다.
dom_ev = K11[0]/K10[0]
print("dominant eigenvalue = ", dom_ev.n()) # dom_ev 출력
print("corres. eigenvector = ", n((1/K10[0])*K10)) # 고유벡터 구하기
print(A.eigenvalues()) # A의 고윳값
```

```

A^10x = (3498205, 6681695, 9264127)
A^11x = (16861595, 32206359, 44653976)
dominant eigenvalue = 4.82007057905412
corres. eigenvector = (1.00000000000000, 1.91003528952706, 2.64825160332228)
[-1.279452315768607?, 1.459362941393819?, 4.820089374374788?]

```

$$A^{10}\mathbf{x} = \begin{bmatrix} 3498205 \\ 6681695 \\ 9264127 \end{bmatrix}, \quad A^{11}\mathbf{x} = \begin{bmatrix} 16861595 \\ 32206359 \\ 44653976 \end{bmatrix}$$

이 값들만으로 보면 두 벡터 사이의 연관성이 적어 보인다. 그러나 각각을 자신의 첫 번째 성분으로 나누면 다음을 얻게 된다.

$$A^{10}\mathbf{x} = 3498205 \begin{bmatrix} 1 \\ 1.910 \\ 2.648 \end{bmatrix}, \quad A^{11}\mathbf{x} = 16861595 \begin{bmatrix} 1 \\ 1.910 \\ 2.648 \end{bmatrix}$$

이제 연관성이 쉽게 확인된다. 만약 이 단계에서 이런 연관성(수렴하는 고유벡터)이 없으면 극한에 가까이 갈 정도로 충분히 큰 k 값을 이용하면 궁극적으로 위와 같은 연관성을 갖는다.

우리는 고윳값/고유벡터의 정의를 이용하여 다음의 결과를 확인할 수 있다. 즉 우세한(dominant) 고윳값은

$$\lambda_1 = \frac{16861595}{3498205} = 4.820$$

이며 그것에 대응하는 고유벡터는 $[1 \ 1.910 \ 2.648]^T$ 이다. 실제로

$$\begin{bmatrix} 1 & 2 & 0 \\ 2 & 1 & 2 \\ 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 1.910 \\ 2.648 \end{bmatrix} = 4.820 \begin{bmatrix} 1 \\ 1.910 \\ 2.648 \end{bmatrix}$$

을 만족한다.

예제 1에서 보듯이 우리는 가장 큰 고윳값과 그에 대응하는 고유벡터를 구하기 위해 행렬 A 의 거듭제곱을 이용하였다. 그러면 도대체 얼마나 큰 거듭제곱이 필요할까? 위의 예에서는 임의로 $A^{10}\mathbf{x}$, $A^{11}\mathbf{x}$ 를 소수 3자리까지 계산할 수 있었다. 그러나 더욱 정확한 고윳값을 원한다면 연속적인 더 높은 거듭제곱의 행렬을 이용하면 된다. 아래 웹사이트에서 제공하는 페론-프로베니

우스 정리(Perron–Frobenius theorem)는 “음이 아닌 어떤 행렬에 대해서도 실수값으로 유일하게 존재하는 크기가 가장 큰 고윳값은 위의 방법으로 언제나 구할 수 있다”는 것을 이론적으로 보여준다.

[참고] http://matrix.skku.ac.kr/sglee/perron_frobenius/perron_frobenius.html

[참고문헌]

- Xanthopoulos, P. et al. Robust Data Mining, Springer, 2013
- Horn, R.A. and Johnson, C.R. Matrix Analysis (2nd ed), Cambridge University Press, 2013
- Trefethen, L.N. and Bau, D. Numerical Linear Algebra, SIAM, 1997
- Golub, G.H. and Van Loan, C.F. Matrix Computations, Johns Hopkins, 2012 (4th ed)
- Wilkinson, J.H. The Algebraic Eigenvalue Problem, Clarendon Press, 1988

2. MNIST 데이터셋을 활용한 손 글씨 숫자 인식 (패턴인식)

동영상, 빅데이터와 인공지능, MNIST <https://youtu.be/UqmV4wEzKIY> (15:57)

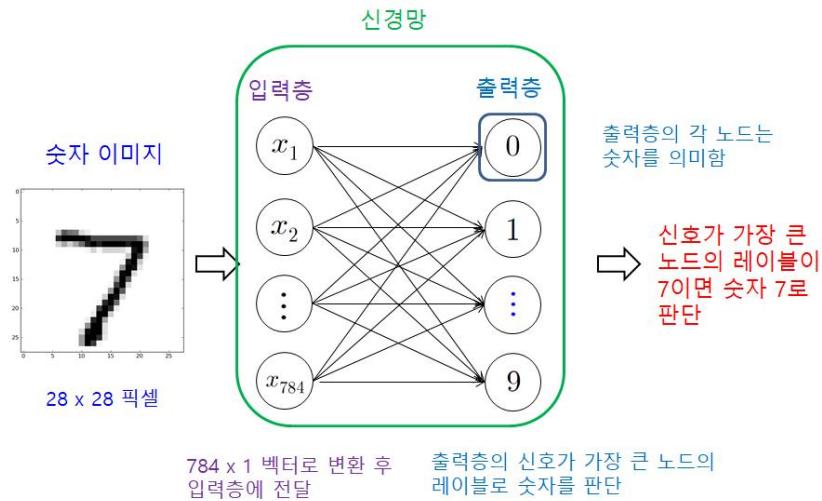
- 다음은 신경망을 활용하여 MNIST 데이터 세트의 손 글씨 숫자를 인식한 사례이다. 자료의 출처는 다음과 같다. 데이터 처리를 위한 코드는 Python/Sage로 새로 만들었다.

[출처] THE MNIST DATABASE of handwritten digits <http://yann.lecun.com/exdb/mnist/>
타리크라시드 지음, 송교석 옮김, 신경망 첫걸음, 한빛미디어, 2017년.

- 기계 학습의 성과를 벤치마크 하는데 가장 널리 사용되는 MNIST 데이터 세트는 60,000개의 학습 데이터와 10,000개의 테스트 데이터로 구성되어 있다. 각 데이터에는 손으로 쓴 0부터 9까지의 숫자의 이미지와 그 레이블(실제 숫자)이 담겨 있다. 데이터 세트의 각 이미지는 가로, 세로가 각 28픽셀(pixel)로 $28 \times 28 = 784$ 개의 숫자를 갖는 벡터로 생각할 수 있으므로, 숫자를 인식하기 위한 신경망의 입력층은 784개의 노드, 출력층은 0부터 9까지의 숫자를 나타내는 10개의 노드로 구성된다. 은닉층(hidden layer)의 노드의 개수에 관하여 정해진 규정은 따로 없으나 이 절의 예시코드에서는 100개로 하였다.

- 이제 어떤 숫자를 나타내는 이미지가 있다고 하자. 그러면 이것을 784×1 의

벡터로 변환하여 신경망의 입력층에 전달한다. 그리고 신경망을 거쳐 얻은 출력층의 신호가 가장 큰 노드의 레이블로 숫자를 판단하게 된다. 예를 들어, 출력층의 신호가 가장 큰 노드의 레이블이 7이면 이를 숫자 7로 판단하게 된다.



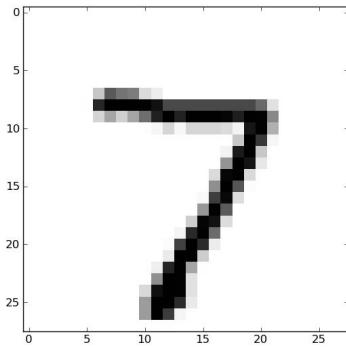
- 아래는 <https://github.com/freebz/Make-Your-Own-Neural-Network> 에 공개된 Python 코드를 일부 수정하여 Sage에서 구현한 것이다. 예를 들어, MNIST 데이터세트로부터 숫자 7의 이미지를 읽어서 보여준다.

[Sage code] <https://sagecell.sagemath.org/> <http://sage.skku.edu/> <http://mathlab.knou.ac.kr:8080/>

```
# mnist 학습 데이터인 csv 파일을 리스트로 불러오기
import numpy
# 행렬을 시각화하기 위한 라이브러리
import matplotlib.pyplot
import csv
import urllib.request
import codecs

url =
'https://media.githubusercontent.com/media/freebz/Make-Your-Own-Neural-Net
work/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.asarray(all_values[1:]).reshape((28,28))
matplotlib.pyplot.imshow(image_array, cmap = 'Greys', interpolation = 'None')
matplotlib.pyplot.savefig('foo.png')
response.close()
```



- 이제 주어진 학습 데이터를 활용하여, 신경망의 적절한 가중치를 얻은 후, 테스트 데이터를 이용하여 신경망의 성능을 판단해보자. 다음은 입력, 은닉, 출력 계층의 신경망으로, MNIST 데이터세트 일부를 학습시켜 테스트용 숫자 이미지를 잘 인식하는지 보여주는 코드이다. 학습 데이터 100개, 테스트 데이터 10개를 아래의 자료로부터 사용하였다.

[출처]

<https://github.com/freebz/Make-Your-Own-Neural-Network>
https://media.githubusercontent.com/media/freebz/Make-Your-Own-Neural-Network/master/mnist_dataset/mnist_train_100.csv
https://media.githubusercontent.com/media/freebz/Make-Your-Own-Neural-Network/master/mnist_dataset/mnist_test_10.csv

[Sage code] <https://sagecell.sagemath.org/> <http://sage.skku.edu/> <http://mathlab.knou.ac.kr:8080/>

3계층의 신경망으로 MNIST 데이터를 학습하는 코드

```
import numpy
# 시그모이드 함수 expit() 사용을 위해 scipy.special 불러오기
import scipy.special
# 행렬을 시각화하기 위한 라이브러리
import matplotlib.pyplot
```

#신경망 클래스의 정의

```

class neuralNetwork:

    # 신경망 초기화하기
    def __init__(self, inputnodes, hiddennodes, outputnodes, learningrate):
        # 입력, 은닉, 출력 계층의 노드 개수 설정
        self.inodes = inputnodes
        self.hnodes = hiddennodes
        self.onodes = outputnodes

        # 가중치 행렬 wih와 who
        # 배열 내 가중치는 w_i_j로 표기. 노드 i에서 다음 계층의 노드 j로 연결됨
        # 의미
        # w11 w21
        # w12 w22 등
        self.wih = numpy.random.normal(0.0, pow(self.hnodes, -0.5),
                                       (self.hnodes, self.inodes))
        self.who = numpy.random.normal(0.0, pow(self.onodes, -0.5),
                                       (self.onodes, self.hnodes))

        # 학습률
        self.lr = learningrate
        # 활성화 함수로는 시그모이드 함수를 이용
        self.activation_function = lambda x: scipy.special.expit(x)

    pass

    # 신경망 학습시키기
    def train(self, inputs_list, targets_list):
        # 입력 리스트를 2차원의 행렬로 변환
        inputs = numpy.array(inputs_list, ndmin=2).T
        targets = numpy.array(targets_list, ndmin=2).T

        # 은닉 계층으로 들어오는 신호를 계산
        hidden_inputs = numpy.dot(self.wih, inputs)
        # 은닉 계층에서 나가는 신호를 계산
        hidden_outputs = self.activation_function(hidden_inputs)

```

```

# 최종 출력 계층으로 들어오는 신호를 계산
final_inputs = numpy.dot(self.who, hidden_outputs)
# 최종 출력 계층에서 나가는 신호를 계산
final_outputs = self.activation_function(final_inputs)
# 출력 계층의 오차는 (실제 값 - 계산 값)
output_errors = targets - final_outputs
# 은닉 계층의 오차는 가중치에 의해 나뉜 출력 계층의 오차들을 재조합해
# 계산
hidden_errors = numpy.dot(self.who.T, output_errors)
# 은닉 계층과 출력 계층 간의 가중치 업데이트
self.who += self.lr * numpy.dot((output_errors * final_outputs * (1.0 - final_outputs)), numpy.transpose(hidden_outputs))
# 입력 계층과 은닉 계층 간의 가중치 업데이트
self.wih += self.lr * numpy.dot((hidden_errors * hidden_outputs * (1.0 - hidden_outputs)), numpy.transpose(inputs))

pass

# 신경망에 질의하기
def query(self, inputs_list):
    # 입력 리스트를 2차원 행렬로 변환
    inputs = numpy.array(inputs_list, ndmin=2).T
    # 은닉 계층으로 들어오는 신호를 계산
    hidden_inputs = numpy.dot(self.wih, inputs)
    # 은닉 계층에서 나가는 신호를 계산
    hidden_outputs = self.activation_function(hidden_inputs)
    # 최종 출력 계층으로 들어오는 신호를 계산
    final_inputs = numpy.dot(self.who, hidden_outputs)
    # 최종 출력 계층에서 나가는 신호를 계산
    final_outputs = self.activation_function(final_inputs)

    return final_outputs

# 입력, 은닉, 출력 노드의 수
input_nodes = 784

```

```

hidden_nodes = 100      # 3000, 300 으로 높이면 65%에서 70%
output_nodes = 10       # 60000, 10000 으로 높이면 65%에서 95%로 정확도 향상

# 학습률
learning_rate = 0.3

# 신경망의 인스턴스를 생성
n = neuralNetwork(input_nodes, hidden_nodes, output_nodes, learning_rate)

# mnist 학습 데이터인 csv 파일을 리스트로 불러오기
import csv
import urllib.request
import codecs

url =
'https://media.githubusercontent.com/media/freebz/Make-Your-Own-Neural-Net-
work/master/mnist_dataset/mnist_train_100.csv'
response = urllib.request.urlopen(url)
training_data_file = csv.reader(codecs.iterdecode(response, 'utf-8'))
training_data_list = list(training_data_file)
response.close()

# 신경망 학습시키기
# 주기(epoch)란 학습 데이터가 학습을 위해 사용되는 횟수를 의미
epochs = 5

for e in range(epochs):
    # 학습 데이터 모음 내의 모든 레코드 탐색
    for record in training_data_list:
        all_values = record
        # 입력 값의 범위와 값 조정
        inputs = (numpy.asarray(all_values[1:]) / 255.0 * 0.99) + 0.01
        # 결과 값 생성 (실제 값인 0.99 외에는 모두 0.01)
        targets = numpy.zeros(output_nodes) + 0.01
        # all_values[0]은 이 레코드에 대한 결과 값
        targets[int(all_values[0])] = 0.99
        n.train(inputs, targets)

```

```

pass
pass

# mnist 테스트 데이터인 csv 파일을 리스트로 불러오기
url = 'https://media.githubusercontent.com/media/freebz/Make-Your-Own-Neural-Net'
work/master/mnist_dataset/mnist_test_10.csv'
response1 = urllib.request.urlopen(url)
test_data_file = csv.reader(codecs.iterdecode(response1, 'utf-8'))
test_data_list = list(test_data_file)
response1.close()

# 신경망 테스트하기
# 신경망의 성능의 지표가되는 성적표를 아무 값도 가지지 않도록 초기화
scorecard = []

# 테스트 데이터 모음 내의 모든 레코드 탐색
for record in test_data_list:
    all_values = record
    # 정답은 첫 번째 값
    correct_label = int(all_values[0])
    # 입력 값의 범위와 값 조정
    inputs = (numpy.asarray(all_values[1:]) / 255.0 * 0.99) + 0.01
    # 신경망에 질의
    outputs = n.query(inputs)
    # 가장 높은 값의 인덱스는 레이블의 인덱스와 일치
    label = numpy.argmax(outputs)
    # 정답 또는 오답을 리스트에 추가
    if (label == correct_label):
        # 정답인 경우 성적표에 1을 더함
        scorecard.append(1)
    else:
        # 정답이 아닌 경우 성적표에 0을 더함
        scorecard.append(0)
    pass
pass

```

정답의 비율인 성적을 계산해 출력

```
scorecard_array = numpy.asarray(scorecard)
print("performance = ", float(scorecard_array.sum()) / scorecard_array.size)
```

```
performance = 0.7
```

- 위의 결과로부터 학습 데이터 100개, 테스트 데이터 10개를 사용했을 때, 신경망의 정확도는 70%임을 확인할 수 있다. 학습 데이터의 수를 늘리게 되면, 이보다 더 높은 정확도를 기대할 수 있는데, 실제로 여기 소개된 코드의 신경망을 60,000 개의 학습 데이터에 의해 학습시키고, 10,000개의 테스트 데이터에 적용하면, (신경망의 가중치의 초기치가 임의로 주어지므로, 정확도의 수치가 일부 다를 수 있으나) 대략 95%의 정확도를 얻게 된다.
- MNIST 데이터 세트의 손 글씨 숫자를 인식한 사례는 신경망을 학습시키기 위한 데이터가 충분히 확보되었기에 가능했다. 이전에는 사용할 수 있는 데이터가 많지 않았으나, 오늘의 세계에서는 매일 2.5×10^{18} (quintillion) 바이트의 데이터가 생성된다고 알려져 있다. 인공지능 시스템이 많은 양의 학습 데이터를 이용하여 특징을 파악하게 되면, 좀 더 나은 결과를 추론할 수 있을 것이다.
- MNIST 숫자 인식 사례와 마찬가지로, 음성 또는 신호를 인식하거나 문자를 인식하는 것도 유사한 방법으로 가능하다. 이보다 더 높은 수준의 분석이 요구되는 자연어 처리(Natural Language Processing, NLP)는 우리가 일상생활에서 사용하는 언어를 컴퓨터가 의미를 분석하여 처리하도록 하는 것으로, 단순히 음성 또는 문자를 인식하는 것 이외에 말하는 사람이 의미한 것, 필요한 것들을 추론을 해야 한다. 예를 들어, 사용자가 "피자 먹고 싶다!"라는 문장을 말했을 때, 인공지능 시스템은 음성의 파형으로부터 "피자 먹고 싶다!"라는 문장을 인식하는 것에서 더 나아가 사용자가 실제로 원하는 정보를 추론을 거쳐 파악하고 찾아서 제공해줘야 한다. 피자 예 ("피자 먹고 싶다!")에서 보면, 음식(food)으로 표기된 '피자'라는 용어의 사용에 주목하여, 말하는 사람이 피자를 만드는 방법을 알고 싶어 한다는 걸 암시하는 '조리법'과 같은 용어가 본문에 없음을 파악하고, 말하는 사람이 피자집을 찾고 있다고 판단할 수 있다. 이 경우 GPS 정보를 가지고 피자를 제공하는 식당을

찾아 근접성, 등급, 가격 순으로 순위를 매긴다. 만약 사용자가 이용한 피자 식당들의 기록이 있다면, 인공지능 시스템은 사용자가 자주 이용하는 식당 중 추천이 많은 가까운 곳을 찾아, 자연어 문장을 생성하여, 우리말로 다음과 같이 추천할 수도 있다.

"여기서 2km 정도 떨어진 곳에 ***라는 피자집이 있습니다."

[참고 문헌]

Bitna Kim and Young Ho Park, Beginner's guide to neural networks for the MNIST dataset using MATLAB, Korean J. Math. 26 (2018), No. 2, pp. 337–348.

Bitna Kim, Handwritten digits classification by neural networks with small data, Master thesis, Kangwon National University, 2018.

타리크라시드 지음, 송교석 옮김, 신경망 첫걸음, 한빛미디어, 2017년.



<2006년 필즈 메달 수상자 Okunkov 교수 연구실, Columbia 대학교>





읽을거리1

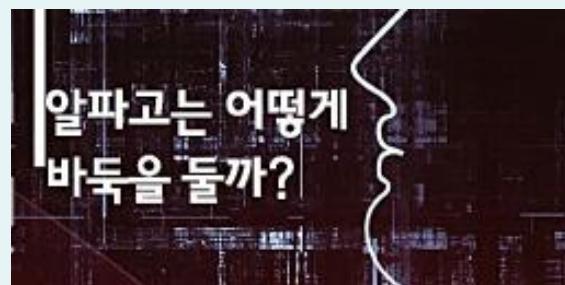
[수학동아 2016년 4월호 특별기획] AlphaGo—Special—p.50—p.57

인공지능, 인간을 뛰어넘은 비밀

지난 3월 9일부터 3월 15일까지 서울 광화문 포시즌스 호텔 특별 대국장에서 펼쳐졌던 바둑 최강자 이세돌 9단과 인공지능 프로그램 아파고의 대결은 알파고의 승리로 끝났다. 대부분의 전문가들이 이세돌 9단의 압승을 예견했으나, 결과는 알파고의 승리였다.

알파고는 어떻게 바둑을 공부해서 이세돌을 이길 수 있었던 것인지 궁금하다. 또, 알파고 같은 인공지능 프로그램을 만들기 위해서는 어떤 수학이 필요할까?

글 김경환 기자(dallgudot@donga.com) 도움 이상구(성균관대 수학과 교수), 김용대(서울대 통계학과 교수), 김동근(아주대 전자공학과 교수), 신진우(KAIST 전기 및 전자공학과 교수), 최승진(포항공과대 컴퓨터공학과 교수)



알파고는 어떻게 바둑을 둘까?

1. 학습하는 컴퓨터

알파고는 바둑 두는 법을 어떻게 배웠을까? 사람이 학교에서 공부하는 것처럼 정해진 교육과정에 따라 공부했을까? 알파고는 ‘딥러닝’을 활용해 프로 바둑기사들의 패턴을 학습했다.

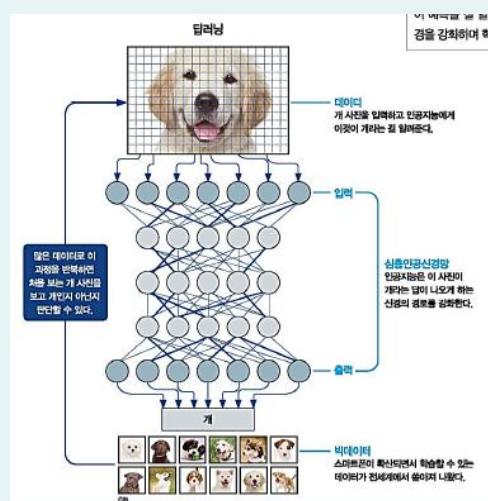
딥러닝은 ‘기계학습’의 방법 중 하나다. 기계학습이란 컴퓨터에게 방대한 데이터를

주고 스스로 일반적인 패턴을 찾는 것을 의미한다. 딥러닝은 데이터만 넣어주면 알아서 각각의 중요한 특징을 찾아 스스로 학습한다.

파블로프의 개 실험

보통의 개는 음식을 보면 침이 나온다. 그런데 음식을 줄 때 종소리를 함께 들려주는 훈련을 반복하면, 청각 신경과 침을 분비하는 신경의 ‘연결’이 강화된다. 결과적으로 개는 종소리만 흔들어줘도 침이 분비된다.

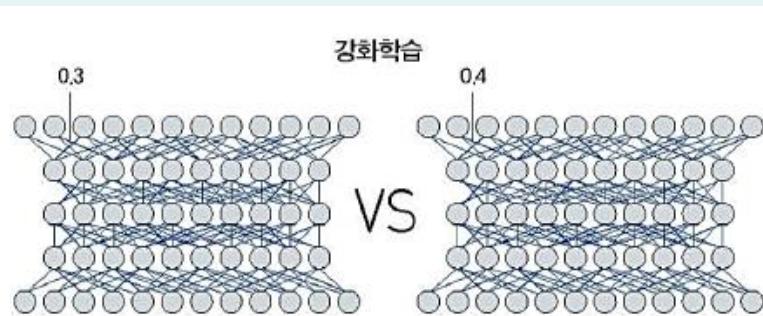
이 실험은 러시아의 생리학자 이반 파블로프의 대표적인 실험으로, 이 원리는 인공지능을 학습시키는 데도 이용된다. 인공지능이 예측을 잘 할 수 있는 방향으로 인공신경을 강화하여 학습하는 것이다.



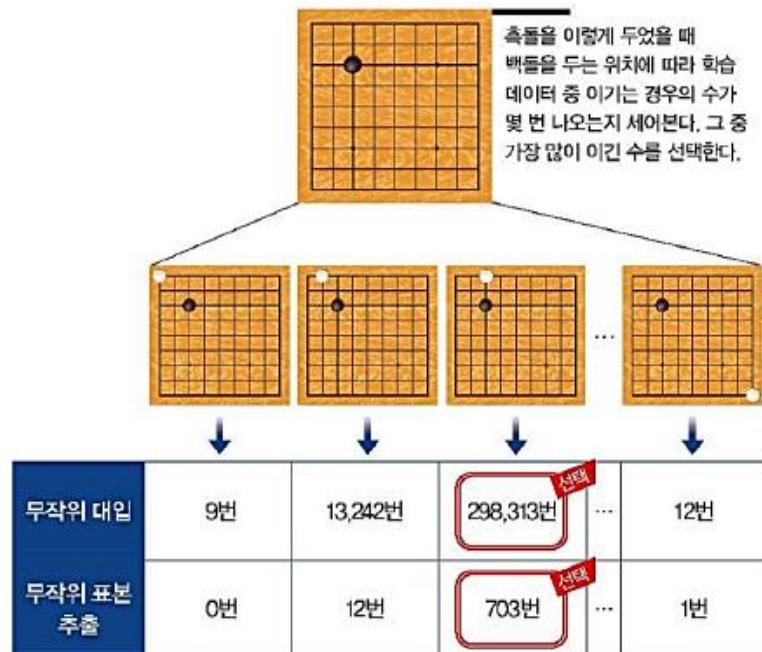
2. 스스로 공부한다!

알파고를 개발한 구글의 인공지능 개발사 ‘딥마인드’의 트레이드 마크는 ‘강화학습’이다. 강화학습은 알파고와 알파고가 경기하며 배우는 것이다. 기존의 데이터만으로는 프로만큼 바둑을 두기 힘들었던 알파고는 강화학습으로 새로운 양질의 데이터를 만들어 바둑 실력을 키웠다.

알파고는 수많은 인공신경 중 하나의 수치를 무작위로 바꿔가며 승률이 더 높은 경우를 강화했다. 예를 들어, 한 신경의 수치가 0.3인 알파고와 0.4인 알파고가 바둑을 여러 번 두게 해 이긴 쪽 수치로 신경을 강화한다.



알파고는 수많은 인공신경 중 하나의 수치를 무작위로 바꿔가며 승률이 더 높은 경우를 강화했다. 예를 들어, 한 신경의 수치가 0.3인 알파고와 0.4인 알파고가 바둑을 여러 번 두게 해 이긴 쪽 수치로 신경을 강화한다.



무작위 대입은 모든 데이터를 다 따지는 것이고,
무작위 표본 추출은 일부 데이터만 뽑은 뒤 그 안에서 따지는 것이다.
무작위로 표본을 뽑으면 결과는 거의 같으면서도 경우의 수가 줄어든다.

3. 무작위 대입 VS 무작위 표본 추출

바둑은 한 경기에서 나올 수 있는 경우의 수가 2.08×10^{170} 으로 어마어마하다. 때문에 모든 경우를 탐색해 수를 생각하는 알고리즘인 ‘무작위 대입’은 비효율적이다. 바둑에 비해 경우의 수가 적은 체스에서는 무작위 대입으로 사람을 이길 수 있었지만, 바둑은 그것이 불가능했다.

그래서 알파고는 ‘무작위 표본 추출’¹⁾이라는 알고리즘을 이용해 표본을 뽑아 탐색하며 바둑을 둔다. 표본이 전체를 나타낼 수 있다는 것을 활용한 것이다. 표본 집단의 크기가 커질수록 정확도가 높아진다. 무작위 표본 추출은 알파고가 다양한 상황에서 가장 좋은 수를 고르도록 한다.

무작위 대입	9번	13,242번	298,313번	12번
무작위 표본 추출	0번	12번	703번	1번

무작위 대입은 모든 데이터를 다 따지는 것이고, 무작위 표본 추출은 일부 데이터만 뽑은 뒤 그 안에서 따지는 것이다. 무작위로 표본을 뽑으면 결과는 거의 같으면서도 경우의 수가 줄어든다.

인공지능이 궁금해? 수학부터 공부하자!

이세돌 9단과 알파고 간의 바둑 대결은 그 대결 자체로도 관심을 끌었지만, 인공지능에 대한 관심으로도 이어졌다. 그러면 훗날 인공지능 연구에 동참하려면 무엇을 공부해야 할까? 인공지능을 이해하고, 개발하기 위해 반드시 알아야 하는 수학의 분야를 알아보자.

선형대수학²⁾은 필수!

인공지능의 기반인 컴퓨터가 어떤 문제를 해결하게 하기 위해서는 ‘선형화’라는 과정이 필요하다. 선형화는 문제를 컴퓨터가 이해하는 언어로 바꾸는 과정으로, 수학적으로 ‘행렬’이 포함된 식으로 바꾸는 것을 말한다. 예를 들어 복잡한 3차 미분방정식을 선형화하면 훨씬 간단한 1차 미분방정식 3개로 된 식이 된다. 그러면 행렬의 성질을 이용해 쉽게 답을 구할 수 있다.

그런데 행렬의 크기가 100만 × 100만 같이 커지면 슈퍼컴퓨터가 풀더라도 시간이 굉장히 오래 걸린다. 이것을 해결하는 답은 선형대수학에 있다.

선형대수학은 크기가 큰 행렬을 컴퓨터가 효과적으로 다루도록 도와준다. 푸는 데 10년 걸릴 문제를 순식간에 풀도록 바꿔주는 것이다.

컴퓨터에게 선형화한 문제를 해결하도록 시키려면 컴퓨터가 이해할 수 있도록 알고리즘을 짜야 한다. 여기에는 ‘수치적 선형대수학’과 ‘수치해석학’을 이용한다. 컴퓨터 프로그래머는 이 알고리즘으로 코딩해 프로그램을 만든다.

선형대수학은 이런 곳에도 쓰인다!

선형대수학은 인공지능뿐만 아니라 자연과학과 공학, 사회과학의 여러 분야에서 활용되고 있다. 1890년에 혜성 궤도를 관측할 때부터 쓰이기 시작했고, 현대에는 빅데이터 분석에도 쓰이고 있다. 또, 우리가 많이 쓰는 구글 검색 엔진 속에도 선형대수학이 이용된다. 전 세계 1위 브랜드 가치를 가진 구글 신화를 창조할 수 있었던 이유는 선형대수학을 이용한 검색엔진 덕분이었던 것이다.

빠지면 안 되는 확률

인공지능은 빅데이터를 입력해서 의사결정과 데이터를 연결해주는 모형(함수)을 만드는 것이라 할 수 있다. 이때 다양한 모형을 만들어 어떤 모형이 예측력이 가장 좋은지 알아내야 한다. 예측을 잘할수록 진보한 인공지능이다. 예를 들어, 알파고의 경우 현재 바둑판의 상황에서 어느 곳에 놓아야 이길지를 예측해 가장 좋은 수를 선택한다.

확률론은 모형을 어떻게 만들어야 예측을 잘 할 수 있는지 연구할 때 쓰인다. 인공지능은 데이터를 기반으로 하기 때문에 모든 판단을 확률적으로 하기 때문이다. 확률론은 모형을 만들 때 직접적으로 쓰이지는 않지만, 모형을 만드는 원리를 제공한다.

이상구 성균관대 수학과 교수는 “알파고 같은 인공지능을 포함해 사물인터넷, 드론 등 관심이 가는 분야가 있다면, 그것에 쓰인 학문을 직접 찾아 공부하는 것이 중요하다”며, “수학, 통계학, 컴퓨터 과학 등 다양한 학문이 관련 있을 텐데, 이를 통해 창의적인 아이디어를 스스로 내보면 큰 도움이 될 것”이라며 관심 있는 분야에 대한 적극적인 행동을 당부했다.

확률론을 쓰는 인공지능도 있다

확률론을 이용한 인공지능 알고리즘에는 ‘베이즈 확률론’이 쓰인다. 베이즈 확률론은 18세기 통계학자 토마스 베이즈의 이름을 딴 이론으로 확률을 ‘지식의 상태를 측정’하는 것이라고 본다. 현대의 많은 기계 학습 방법은 이 원리에 의해 만들어졌다. 하지만 딥러닝에는 이 방법이 쓰이지 않는다. 최근 인공지능 개발에 있어서 베이즈 확률론은 대세가 아니다.

인공지능 수학자도 등장할까?

인공지능이 넘어서기 힘들 것이라던 바둑에서 인공지능이 승리를 거두었다. 그런데 여기서 한 가지 의문이 든다. 미래에는 인공지능이 수학을 연구하는 날이 올 수도 있지 않을까? 박형주 국가수리과학연구소 소장은 “이미 수학의 여러 정리를 컴퓨터가 증명하게 하는 분야가 있다”며, “아직 사람의 손에 의해 증명된 문제를 시켜보는 단계에 지나지 않지만, 미래에는 사람처럼 새로운 것을 증명해내는 인공지능이 나올 수도 있다”고 말했다.

엄상일 KAIST 수리과학과 교수는 “컴퓨터가 사람이 한 증명을 보고 맞는지 틀린지를 판단하는 소프트웨어는 이미 쓰이고 있다”며, “하지만 수학자처럼 새로운 것을 창조해내는 인공지능이 나올지는 아직 지켜봐야 할 것”이라고 조심스럽게 미래를 전망했다.

인공지능의 발전은 우리의 미래를 어떻게 바꿔놓을까? 영화 ‘매트릭스’나 ‘터미네이터’에 나오는 암울한 미래일지, 영화 ‘로봇, 소리’처럼 사람과 인공지능이 서로 도우며 공존하는 아름다운 미래일지 궁금하다. 만약 이번 대국으로 인공지능에 관심이 생겼다면, 관련된 지식을 들려보는 것은 어떨까? 그렇게 시작한 공부가 미래 인공지능 역사의 중심에 서게 되는 출발점이 될지도 모른다.

[수학동아 2016년 4월호 특별기획] AlphaGo-Special-p.50-p.57

-
- 1) 몬테카를로 트리 탐색이라고도 한다.
 - 2) 행렬, 연립 선형 방정식, 벡터, 선형 변환 등을 연구하는 대수학의 한 분야로, 컴퓨터가 문제를 효과적으로 다루게 하는 대부분의 알고리즘은 선형대수학에 기반을 두고 있다.

- [명령어 모음] <http://matrix.skku.ac.kr/Lab-Book/Sage-Lab-Manual-2.htm>
 [연습문제] <http://matrix.skku.ac.kr/LA-Lab/>
 [디지털교과서] <http://matrix.skku.ac.kr/LA-K/>

0. Big Picture

1. 벡터
2. 선형연립방정식
3. 행렬과 행렬식
4. 일차독립과 기저 및 차원
5. 선형변환
6. 고윳값, 고유벡터, 대각화
7. SVD (특이값 분해)
8. 이차형식(quadratic form)

선형대수학은 이런 곳에도 쓰인다!



선형대수학은 인공지능뿐만 아니라 자연과학과 공학, 사회과학의 여러 분야에서 활용되고 있다. 1890년에 혜성 궤도를 관측할 때부터 쓰이기 시작했고, 현대에는 빅데이터 분석에도 쓰이고 있다. 또, 우리가 많이 쓰는 구글 검색 엔진 속에서도 선형대수학이 이용된다. 전 세계 1위 브랜드 가치를 가진 구글 신화를 창조할 수 있었던 이유는 선형대수학을 이용한 검색엔진 덕분이었던 것이다.

0. Big Picture (기초 선형대수학 개념의 구성)

2강 동영상, 선형대수학 Big Picture <https://youtu.be/3tBXcnxZb8M> (39분)

■ 벡터 :

예를 들어 힘, 속도, 가속도 등 방향과 크기를 모두 포함하는 물리량을 벡터 (vector)라 하고, 일반적으로 n 개 실수의 순서조 (순서쌍)

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = (x_1, \dots, x_n) \text{로 } n\text{차원 공간의 벡터를 나타낸다.}$$

■ 벡터 \mathbf{x} 의 노름(norm, length, magnitude), $\|\mathbf{x}\| = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2}$

- 벡터 $\mathbf{x} = (x_1, x_2, \dots, x_n)$, $\mathbf{y} = (y_1, y_2, \dots, y_n)$ 의 내적(Euclidean inner product, dot product)

$$\mathbf{x} \cdot \mathbf{y} = x_1y_1 + x_2y_2 + \dots + x_ny_n = \langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{y}^T \mathbf{x}$$

- 벡터공간 :

임의의 집합 $V (= \emptyset)$ 에 두 연산, 덧셈(vector addition, $+$)과 스칼라 배(scalar multiplication, \cdot)가 정의되고, 2개의 기본성질(두 연산에 대하여 닫혀있다-closed)과 8개의 연산 성질을 만족하면 '($V, +, \cdot$)' 가 벡터공간(vector space))을 이룬다'고 한다. 예, \mathbb{R}^3 , \mathbb{R}^n , \mathbb{C}^n , 행렬공간 등

- 부분공간 :

$(\emptyset \neq) W \subset V$ 가 V 의 부분공간(subspace)이라는 것은 벡터공간의 부분집합이면서 동시에 그 자체로 벡터공간이 되는 경우를 말한다.

- 2 step 부분공간 test :

W 가 V 의 부분공간이 되는지 확인하려면 다음의 두 가지를 보이면 된다.

$$(1) \mathbf{u} + \mathbf{v} \in W \quad \forall \mathbf{u}, \mathbf{v} \in W \quad (2) \alpha \mathbf{u} \in W \quad \forall \mathbf{u} \in W, \alpha \in \mathbb{R}$$

- 일차결합 :

$U = \{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_k\} \subseteq V$ 의 한 일차결합(linear combination)이란 적당한 스칼라 $\alpha_i \in \mathbb{R}$ 가 존재하여 다음을 만족하는 경우를 말한다.;

$$\mathbf{v} = \alpha_1 \mathbf{u}_1 + \alpha_2 \mathbf{u}_2 + \dots + \alpha_k \mathbf{u}_k$$

- Span(일차결합들의 전체집합) :

$U = \{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_k\} \subseteq V$ 의 일차결합 전체를 모은 집합을 U 에 의하여 생성된(spanned) 부분공간이라 하고 다음과 같이 표기한다.

$$S = \{\alpha_1 \mathbf{u}_1 + \alpha_2 \mathbf{u}_2 + \dots + \alpha_k \mathbf{u}_k \mid \alpha_i \in \mathbb{R}, \mathbf{u}_i \in U\} := \langle U \rangle = U \text{의 span}$$

■ 일차독립 :

$U = \{ \mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_k \}$ 에 대하여

$$\alpha_1 \mathbf{u}_1 + \alpha_2 \mathbf{u}_2 + \cdots + \alpha_k \mathbf{u}_k = \mathbf{0} \Rightarrow \alpha_1 = \alpha_2 = \cdots = \alpha_k = 0$$

을 만족하면 U 는 일차독립(*linearly independent*)이라 한다.

■ 일차종속 :

U 가 일차독립이 아니면 일차종속(*linearly dependent*)이라 한다. U 가 일차종속이면 모두는 영은 아닌 스칼라 α_i (즉, α_i 중 적어도 하나는 영이 아닌) 가 존재하여 다음을 만족한다.

$$\alpha_1 \mathbf{u}_1 + \alpha_2 \mathbf{u}_2 + \cdots + \alpha_k \mathbf{u}_k = \mathbf{0}$$

■ 벡터공간의 기저 :

다음을 만족하는 B 를 V 의 기저(*basis*)라 한다.

- (1) B 가 일차독립이다. (2) $\langle B \rangle = V$

■ 벡터공간의 차원 :

B 가 V 의 기저라 할 때 V 의 차원(*dimension*)은 다음과 같이 정의된다.

$$\dim V = |B|$$

■ 행공간 :

행렬 $A = \begin{bmatrix} A_{(1)} \\ A_{(2)} \\ \vdots \\ A_{(m)} \end{bmatrix}$ 의 행벡터들의 일차결합 전체의 집합을 행공간(*row space*)이라 한다. 즉, $\text{Row}(A) = \langle A_{(1)}, A_{(2)}, \dots, A_{(m)} \rangle$

■ 열공간 :

행렬 $A = [A^{(1)} \ A^{(2)} \ \cdots \ A^{(n)}]$ 의 열벡터들의 일차결합 전체의 집합을 열공간(*column space*)이라 한다. 즉, $\text{Col}(A) = \langle A^{(1)}, A^{(2)}, \dots, A^{(n)} \rangle$

■ 계수 :

행공간(열공간)의 차원을 행렬의 계수(rank)라 한다. 즉

$$\text{rank}(A) = \dim \text{Row}(A) = \dim \text{Col}(A).$$

■ 벡터공간의 기저는 다음을 만족한다.

(<http://matrix.skku.ac.kr/mt-04/chp3/3p.html> 의 Lemma 3.9 참조)

(1) 기저는 maximal linearly independent subset 이다.

(2) 기저는 minimal spanning subset 이다.

■ 초평면(hyperplane) :

$\mathbf{a} (\neq 0) \in \mathbb{R}^n$ 에 대하여 $\mathbf{a}^\perp = \{(x_1, x_2, \dots, x_n) \in \mathbb{R}^n \mid a_1x_1 + \dots + a_nx_n = 0\}$ 를 \mathbb{R}^n 의 초평면(hyperplane)이라 한다.

<http://matrix.skku.ac.kr/knowls/cla-week-9-sec-7-3.html>

■ 정사영 :

$\mathbf{p} = \text{proj}_{\langle \mathbf{a} \rangle} \mathbf{x} = \frac{\mathbf{x} \cdot \mathbf{a}}{\|\mathbf{a}\|^2} \mathbf{a}$ 에서 \mathbf{p} 벡터를 \mathbf{x} 의 $\text{span}\{\mathbf{a}\}$ 상의 정사영(또는 직교사영, orthogonal projection)이라 하고, $\mathbf{w} = \mathbf{x} - \mathbf{p} = \mathbf{x} - \text{proj}_{\langle \mathbf{a} \rangle} \mathbf{x}$ 를 \mathbf{x} 의 $\text{span}\{\mathbf{a}\}$ 상의 직교성분(orthogonal component)이라 한다.

<http://matrix.skku.ac.kr/knowls/cla-week-10-sec-7-5.html>

■ 행렬의 고윳값과 고유벡터 :

A 가 n 차의 정사각행렬일 때, $\det(\lambda I - A) = 0$ 을 만족하는 스칼라 λ 를 A 의 고윳값(eigenvalue)이라 하고, $A\mathbf{x} = \lambda\mathbf{x}$ 을 만족하는 0 아닌 벡터 $\mathbf{x} \in \mathbb{R}^n$ 를 고윳값 λ 에 대응하는 A 의 고유벡터(eigenvector)라고 한다.

<http://matrix.skku.ac.kr/knowls/cla-week-6.html>

■ 행렬의 대각화 :

A 가 어떤 대각선행렬과 닮음행렬일 때, 즉 적당한 가역행렬 P 가 존재하여 $P^{-1}AP$ 가 대각선행렬일 때 A 를 대각화가능한(diagonalizable) 행렬이라 하며, 이 때 행렬 P 를 A 를 대각화하는(diagonalizing) 행렬이라고 한다.

<http://matrix.skku.ac.kr/knowls/cla-week-11-sec-8-2.html>

- SVD(특이값분해, singular value decomposition), 일반화된 역행렬, 최소제곱해, 선형회귀 :
행렬 A 는 $A = U\Sigma V^T$ 로 분해될 수 있다는 이론을 행렬의 특이값분해(SVD) 또는 간단히 행렬 A 의 SVD라 부른다. <http://matrix.skku.ac.kr/2018-album/SVD.html>
- 차원축소(dimension reduction)와 변수추출(feature extraction) :
어떤 목적에 따라서 <주요 성질을 대부분 보존하면서도> 데이터(행렬)의 크기(차원)를 크게 줄이는 다양한 방법
- 주축정리(主軸定理, 영어: principal axis theorem) :
이차 형식을 기술하는 문제에서, 어떤 이차 형식이든 적절한 변수 변환을 통해 혼합항이 없는 행렬 형태로 표현할 수 있음을 보장하는 정리
<http://matrix.skku.ac.kr/knowls/CLA-Week-12.html>
- 고윳값 분해(eigen decomposition) :
고윳값과 고유벡터를 이용하여 주어진 행렬을 고윳값 행렬과 고유벡터 행렬의 곱으로 분해하는 표현
- PCA (주성분분석, Principal Component Analysis) :
PCA는 데이터의 분산(variance)을 최대한 보존하면서 서로 직교하는 새 기저(축)를 찾아, 고차원 공간의 표본들을 선형 연관성이 없는 저차원 공간으로 변환하는 기법
<https://youtu.be/0IKbslNH7xk>
- 서포트 벡터 머신(support vector machine, SVM) :
<https://www.youtube.com/watch?v=eHsErIPJWUU&hd=1>
데이터를 분류하는 것은 기계학습에 있어서 일반적인 작업이다. 주어진 데이터 점들이 두 개의 클래스 안에 각각 속해 있다고 가정할 때, 새로운 데이터 점이 두 클래스 중 어느 곳에 속하는지 결정하는 것은 중요한 과제이다. SVM에서 풀고자 하는 문제는 "How do we divide the space with decision boundaries?"이다. SVM은 텍스트와 하이퍼텍스트를 분류하는 과정에서, 학습 데이터를 상당히 줄일 수 있게 해준다. ■



읽을거리

“선형대수학과 구글(Google) 검색엔진”

– 페이지랭크 알고리즘

<https://www.scienceall.com/미래를-여는-수학-⑦-Google-검색의-비밀은/>

글 | 이상구 성균관대 수학과 교수

‘수학 잘하는 법’, ‘친구를 사귀는 법’, ‘어버이날 선물’….

검색 엔진은 어느새 무언가가 궁금한 사람들이 몰려드는 신과 같은 존재가 됐다. 검색 창에 궁금한 점을 입력하면 검색엔진은 전 세계에 흩어져 있는 데이터에서 해당 검색어에 가장 알맞은 자료를 찾아 보여 준다. 물론 검색엔진이 찾아준 자료가 모두 사용자에게 딱 맞기는 어렵다. 엉뚱한 정보만 잔뜩 나온다면 화가 나서 그 검색엔진을 다시는 이용하지 않을지도 모른다. 그래서 마치 마음을 읽은 듯이 원하는 정보를 가장 잘 찾아주는 검색엔진이 인기가 좋다. 세계적으로 그 선두에는 구글이 있다. 그리고 그 비결에는 수학이 있다. 구글의 역사는 어디서 시작된 걸까? 구글의 창립자 래리 페이지와 세르게이 브린은 1995년 스탠퍼드대 대학원에서 만났다. 브린이 2년 선배이긴 하지만 동갑이었기 때문에 둘은 급속도로 친해졌다. 그리고 수학에 재능이 있는 브린과 컴퓨터에 관해 박학다식한 페이지는 수학을 이용한 컴퓨터 연구에 박차를 가하게 된다. 구글 엔진에서는 한 웹페이지에서 다른 웹페이지로 연결하는 링크가 있으면, 그 링크를 일종의 투표로 본다. 즉 투표수가 많은 웹페이지를 좋은 정보가 있는 사이트라고 가정하고 검색 결과 상단에 배치시킨다. 이것이 바로 페이지랭크 알고리즘의 원리로, 웹페이지의 관계를 행렬로 나타낸 뒤 가장큰 고윳값과 대응하는 고유벡터를 이용하여 PageRank를 구한다. (자세한 내용은 [Part4 읽을거리 1](#)에 소개한다.)

Example 4.9 (Google’s PageRank – Webpages as Eigenvectors)

Google uses the eigenvector corresponding to the maximal eigenvalue of a matrix A to determine the rank of a page for search. The idea for the PageRank algorithm, developed at Stanford University by Larry Page and Sergey Brin in 1996, was that the importance of any web page can be approximated by the importance of pages that link to it. For this, they write down all web sites as a huge directed graph that shows which page links to which. PageRank computes the weight (importance) $x_i \geq 0$ of a web site a_i by counting the number of pages pointing to a_i . Moreover, PageRank takes into account the importance of the web sites that link to a_i . The navigation behavior of a user is then modeled by a transition matrix A of this graph that tells us with what (click) probability somebody will end up

on a different web site. The matrix A has the property that for any initial rank/importance vector x of a web site the sequence x, Ax, A^2x, \dots converges to a vector x^* . This vector is called the *PageRank* and satisfies $Ax^* = x^*$, i.e., it is an eigenvector (with corresponding eigenvalue 1) of A . After normalizing x^* , such that $\|x^*\| = 1$, we can interpret the entries as probabilities. More details and different perspectives on PageRank can be found in the original technical report (Page et al., 1999).

1. 벡터

<http://matrix.skku.ac.kr/math4ai/part1/>

2강 동영상, 벡터, 정사영, 최단거리 <https://youtu.be/UdCJCK2MWDU> (52:42) 중 30분~52분

1.1 벡터 (Vector)

참고 동영상: <http://youtu.be/aeLVQoPQMPE> <http://youtu.be/85kGK6bJLns>

실습 사이트: <http://matrix.skku.ac.kr/knowls/CLA-Week-1-Sec-1-1.html>

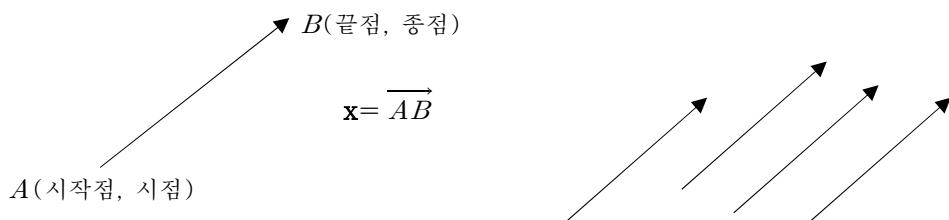
- 스칼라(scalar) :

길이, 넓이, 질량, 온도 – 크기만 주어지만 완전히 표시되는 양

- 벡터(vector) :

속도, 위치이동, 힘 – 크기뿐만 아니라 방향까지 지정하지 않으면 완전히 표현할 수 없는 양

- 벡터는 크기와 방향을 갖는 유향선분 – 2차원, 3차원 공간의 벡터는 화살표로 표현 가능



- 시작점과 끝점이 같아서 크기가 0인 벡터를 영벡터라 한다(영벡터는 크기가 0이므로 방향은 임의의 방향으로 한다).

정의. [*n*차원 벡터(*n*-dimensional vector)]

*n*개의 실수의 순서조 (x_1, x_2, \dots, x_n) 을 *n*차원 벡터(*n*-dimensional vector)라 하고

$$\mathbf{x} = (x_1, x_2, \dots, x_n) = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}_{n \times 1}$$

로 나타낸다. 이때 실수 x_1, x_2, \dots, x_n 을 \mathbf{x} 의 성분이라 한다.

정의. [벡터의 상등] \mathbb{R}^n 의 벡터

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

에 대하여 $x_i = y_i$ ($i = 1, 2, \dots, n$)이면 $\mathbf{x} = \mathbf{y}$ 라고 한다.

정의. [벡터 합과 스칼라배]

\mathbb{R}^n 의 벡터

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

와 스칼라 k 에 대하여 두 벡터의 합 $\mathbf{x} + \mathbf{y}$ 와 k 에 의한 \mathbf{x} 의 스칼라배 $k\mathbf{x}$ 를 각각 다음과 같이 정의한다.

$$(i) \quad \mathbf{x} + \mathbf{y} = \begin{bmatrix} x_1 + y_1 \\ x_2 + y_2 \\ \vdots \\ x_n + y_n \end{bmatrix} \quad (ii) \quad k\mathbf{x} = \begin{bmatrix} kx_1 \\ kx_2 \\ \vdots \\ kx_n \end{bmatrix}$$

또한 \mathbb{R}^n 에서 모든 성분이 0인 벡터를 영벡터 또는 원점이라 하고 $\mathbf{0}$ 으로 나타낸다. 그러면 임의의 벡터 $\mathbf{x} \in \mathbb{R}^n$ 에 대하여

$$\mathbf{x} + \mathbf{0} = \mathbf{x}, \quad \mathbf{x} + (-1)\mathbf{x} = \mathbf{0}$$

이 성립함을 쉽게 알 수 있다. 여기서 $(-1)\mathbf{x} = -\mathbf{x}$ 로 정의하며 $-\mathbf{x}$ 를 \mathbf{x} 의 음벡터라 한다.

- 모든 n 차원 벡터 전체의 집합을 n -공간 (n 차원 공간) \mathbb{R}^n 으로 나타낸다. 즉

$$\mathbb{R}^n = \{(x_1, x_2, \dots, x_n) \mid x_i \in \mathbb{R}, i = 1, 2, \dots, n\}$$

예제 1 \mathbb{R}^4 의 벡터 $\mathbf{x} = \begin{bmatrix} 1 \\ 2 \\ -3 \\ 4 \end{bmatrix}$, $\mathbf{y} = \begin{bmatrix} -2 \\ 4 \\ 1 \\ 0 \end{bmatrix}$ 에 대하여 $\mathbf{x} + \mathbf{y}$, $\mathbf{x} - \mathbf{y}$, $(-2)\mathbf{x}$ 를 구하여라.

$$\begin{aligned} \mathbf{x} + \mathbf{y} &= \begin{bmatrix} 1 + (-2) \\ 2 + 4 \\ -3 + 1 \\ 4 + 0 \end{bmatrix} = \begin{bmatrix} -1 \\ 6 \\ -2 \\ 4 \end{bmatrix} = (-1, 6, -2, 4), \quad \mathbf{x} - \mathbf{y} = \begin{bmatrix} 1 - (-2) \\ 2 - 4 \\ -3 - 1 \\ 4 - 0 \end{bmatrix} = \begin{bmatrix} 3 \\ -2 \\ -4 \\ 4 \end{bmatrix} \\ &= (3, -2, -4, 4) \quad (-2)\mathbf{x} = (-2) \begin{bmatrix} 1 \\ 2 \\ -3 \\ 4 \end{bmatrix} = \begin{bmatrix} -2 \\ -4 \\ 6 \\ -8 \end{bmatrix} = (-2, -4, 6, -8) \quad \blacksquare \end{aligned}$$

[실습 : <http://sage.skku.edu/>]

```
a = vector([1, 2, -3, 4])    # 벡터 생성, 형식은 vector([성분, 성분, ..., 성분])
b = vector([-2, 4, 1, 0])
```

```

print("a =", a)
print("b =", b)
print()
print("a + b =", a + b)    # 벡터의 합
print("a - b =", a - b)    # 벡터의 차
print("-2*a =", -2*a)     # 벡터의 스칼라배

```

a= (1, 2, -3, 4)

b= (-2, 4, 1, 0)

a+b= (-1, 6, -2, 4)

a-b= (3, -2, -4, 4)

-2*a= (-2, -4, 6, -8)

<http://matrix.skku.ac.kr/2018-album/LA-Sec-1-1-lab.html>

1.2 내적

참고 동영상: <http://youtu.be/g55dfkmlTHE>

실습 사이트: <http://matrix.skku.ac.kr/knowls/CLA-Week-1-Sec-1-2.html>

정의. \mathbf{x} 의 노름(norm, length, magnitude)

\mathbb{R}^n 의 벡터 $\mathbf{x} = (x_1, x_2, \dots, x_n)$ 에 대하여

$$\|\mathbf{x}\| = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2}$$

을 \mathbf{x} 의 노름(norm, length, magnitude)이라 한다.

위의 정의에서 $\|\mathbf{x}\|$ 는 원점에서 점 $P(x_1, x_2, \dots, x_n)$ 에 이르는 거리로 정의됨을 의미한다. 따라서 \mathbb{R}^n 의 두 벡터 $\mathbf{x} = (x_1, x_2, \dots, x_n)$, $\mathbf{y} = (y_1, y_2, \dots, y_n)$ 에 대하여 $\|\mathbf{x} - \mathbf{y}\|$ 는 두 점 $P(x_1, x_2, \dots, x_n)$ 과 $Q(y_1, y_2, \dots, y_n)$ 사이의 거리(distance)로 정의한다. 즉,

$$\|\mathbf{x} - \mathbf{y}\| = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \dots + (x_n - y_n)^2}$$

예제 2 \mathbb{R}^4 의 벡터 $\mathbf{x} = (2, -1, 3, 2)$, $\mathbf{y} = (3, 2, 1, -4)$ 에 대하여 $\|\mathbf{x}\|$, $\|\mathbf{y}\|$, $\|\mathbf{x} - \mathbf{y}\|$ 를 구하여라.

$$\begin{aligned}\|\mathbf{x}\| &= \sqrt{2^2 + (-1)^2 + 3^2 + 2^2} = \sqrt{18} = 3\text{sqrt}(2) \\ \|\mathbf{y}\| &= \sqrt{3^2 + 2^2 + 1^2 + (-4)^2} = \sqrt{30} = \text{sqrt}(30) \\ \|\mathbf{x} + \mathbf{y}\| &= \sqrt{(2+3)^2 + (-1+2)^2 + (3+1)^2 + (2+(-4))^2} = \sqrt{46} \\ \|\mathbf{x} - \mathbf{y}\| &= \sqrt{(2-3)^2 + (-1-2)^2 + (3-1)^2 + (2-(-4))^2} = \sqrt{50} = 5\text{sqrt}(2)\end{aligned}\blacksquare$$

```
a = vector([2, -1, 3, 2])
b = vector([3, 2, 1, -4])
print(a.norm())    # 노름 구하기, 형식은 a.norm()
print(b.norm())    # 노름 구하기
print((a - b).norm())  # 거리 구하기
```

$3\text{sqrt}(2)$ # sqrt(2) 는 $\sqrt{2}$ 를 의미한다. (벡터 a의 norm)

$\text{sqrt}(30)$ # sqrt(30) 은 벡터 b의 norm 이다.

$5\text{sqrt}(2)$ # (a - b)의 norm [실습 : <http://sage.skku.edu/>]

정의. [내적(Euclidean inner product, dot product)]

\mathbb{R}^n 의 벡터 $\mathbf{x} = (x_1, x_2, \dots, x_n)$, $\mathbf{y} = (y_1, y_2, \dots, y_n)$ 에 대하여 실수

$$x_1y_1 + x_2y_2 + \cdots + x_ny_n$$

을 \mathbf{x} 와 \mathbf{y} 의 내적(Euclidean inner product, dot product)이라 하고 $\mathbf{x} \cdot \mathbf{y}$ 로 나타낸다. 즉

$$\mathbf{x} \cdot \mathbf{y} = x_1y_1 + x_2y_2 + \cdots + x_ny_n = \langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{y}^T \mathbf{x}$$

■ $\mathbf{x} \cdot \mathbf{x} = x_1^2 + x_2^2 + \cdots + x_n^2 = (\sqrt{x_1^2 + x_2^2 + \cdots + x_n^2})^2 = \|\mathbf{x}\|^2$

정리.

코시-슈바르츠 부등식

\mathbb{R}^n 의 임의의 벡터 \mathbf{x} , \mathbf{y} 에 대하여 다음 부등식이 성립한다.

$$|\mathbf{x} \cdot \mathbf{y}| \leq \|\mathbf{x}\| \|\mathbf{y}\|$$

단, 등호는 \mathbf{x} , \mathbf{y} 중 하나가 다른 것의 실수배일 때만 성립한다.

예제 3 \mathbb{R}^4 의 벡터 $\mathbf{x} = (2, -1, 3, 2)$, $\mathbf{y} = (3, 2, 1, -4)$ 에 대하여 내적 $\mathbf{x} \cdot \mathbf{y}$ 를 구하여라.

답: $x_1y_1 + x_2y_2 + x_3y_3 + x_4y_4 = -1$

```
a = vector([2, -1, 3, 2])
b = vector([3, 2, 1, -4])
print("a =", a)
print("b =", b)
print()
print(a.inner_product(b)) # 내적 구하기, 형식은 a.inner_product(b)
```

```
a= (2, -1, 3, 2)
b= (3, 2, 1, -4)
```

-1 # 내적 $\mathbf{x} \cdot \mathbf{y} = x_1y_1 + x_2y_2 + x_3y_3 + x_4y_4 = -1$ [답]

<http://matrix.skku.ac.kr/2018-album/LA-Sec-1-2-lab.html>

1.3 정사영

참고 동영상: <http://youtu.be/4UGACWyoG> <http://youtu.be/YB976T1w0kE>

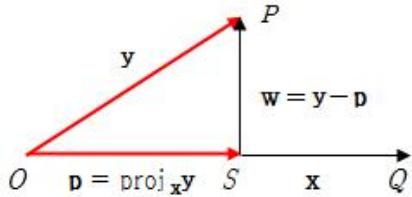
실습 사이트: <http://matrix.skku.ac.kr/knowls/CLA-Week-1-Sec-1-3.html>

정의.

\mathbf{x} 위로의 \mathbf{y} 의 정사영 (projection of \mathbf{y} onto \mathbf{x})

벡터 $\mathbf{x} = \overrightarrow{OQ}$ 와 $\mathbf{y} = \overrightarrow{OP}$ 가 \mathbb{R}^3 에 있고, $\mathbf{x} \neq \mathbf{0}$ 라 하자. 그러면 점 P 에서 OQ 에 내린 수선의 발을 S 라 할 때, 벡터 $\mathbf{p} = \overrightarrow{OS}$ 를 \mathbf{x} 위로의 \mathbf{y} 의 정사영(projection of \mathbf{y}

onto \mathbf{x})이라 하고 $\text{proj}_{\mathbf{x}} \mathbf{y} = \mathbf{p}$ 로 나타낸다. 이때 벡터 $\mathbf{w} = \overrightarrow{SP} = \mathbf{y} - \mathbf{p}$ 를 \mathbf{x} 에 직교인 \mathbf{y} 의 벡터성분(vector component)이라 한다. 따라서 \mathbf{y} 는 두 벡터의 합 $\mathbf{y} = \mathbf{p} + \mathbf{w}$ 로 나타내진다.



- \mathbf{p} 가 \mathbf{x} 에 평행이므로, $\mathbf{p} = t \mathbf{x}$ 이다. 그리고 $\mathbf{y} - \mathbf{p}$ 는 \mathbf{x} 에 직교(orthogonal)이므로,

$$\mathbf{x} \cdot (\mathbf{y} - \mathbf{p}) = 0 \Rightarrow 0 = \mathbf{x} \cdot \mathbf{y} - \mathbf{x} \cdot \mathbf{p} = \mathbf{x} \cdot \mathbf{y} - t \mathbf{x} \cdot \mathbf{x}.$$

따라서 $\mathbf{x} \cdot \mathbf{y} = t \mathbf{x} \cdot \mathbf{x}$ 이고

$$t = \frac{\mathbf{x} \cdot \mathbf{y}}{\mathbf{x} \cdot \mathbf{x}} = \frac{\mathbf{y} \cdot \mathbf{x}}{\mathbf{x} \cdot \mathbf{x}} = \frac{\mathbf{y} \cdot \mathbf{x}}{\|\mathbf{x}\|^2}. \quad \blacksquare$$

정리. 정사영

\mathbb{R}^3 의 벡터 $\mathbf{x} (\neq 0)$, \mathbf{y} 에 대하여 다음이 성립한다.

$$\text{proj}_{\mathbf{x}} \mathbf{y} = t \mathbf{x} \quad \text{여기서 } t = \frac{\mathbf{y} \cdot \mathbf{x}}{\mathbf{x} \cdot \mathbf{x}} = \frac{\mathbf{y} \cdot \mathbf{x}}{\|\mathbf{x}\|^2}. \quad \blacksquare$$

예제 4 $\mathbf{x} = (2, -1, 3)$, $\mathbf{y} = (4, -1, 2)$ 에 대하여 \mathbf{x} 위로의 \mathbf{y} 의 정사영 $\text{proj}_{\mathbf{x}} \mathbf{y}$ 와 \mathbf{x} 에 직교인 \mathbf{y} 의 벡터성분 \mathbf{w} 를 구하여라.

$$(\text{proj}_{\mathbf{x}} \mathbf{y} = \frac{\mathbf{y} \cdot \mathbf{x}}{\mathbf{x} \cdot \mathbf{x}} \mathbf{x} = \frac{\mathbf{y} \cdot \mathbf{x}}{\|\mathbf{x}\|^2} \mathbf{x}, \quad \mathbf{w} = \mathbf{y} - \text{proj}_{\mathbf{x}} \mathbf{y})$$

풀이 $\mathbf{y} \cdot \mathbf{x} = 15$ 이므로

$$\text{proj}_{\mathbf{x}} \mathbf{y} = \frac{(\mathbf{y} \cdot \mathbf{x})}{\|\mathbf{x}\|^2} \mathbf{x} = \frac{15}{14} (2, -1, 3) = \left(\frac{15}{7}, -\frac{15}{14}, \frac{45}{14} \right) = (15/7, -15/14, 45/14),$$

$$\mathbf{w} = \mathbf{y} - \text{proj}_{\mathbf{x}} \mathbf{y} = (4, -1, 2) - \left(\frac{15}{7}, -\frac{15}{14}, \frac{45}{14} \right) = \left(\frac{13}{7}, \frac{1}{14}, -\frac{17}{14} \right) = (13/7, 1/14, -17/14) \blacksquare$$

```

x = vector([2, -1, 3])
y = vector([4, -1, 2])
yx = y.inner_product(x)    # 벡터 x와 y의 내적 구하기
xx = x.inner_product(x)    # 벡터 x와 x의 내적 구하기
p = yx/xx*x   # 벡터의 정사영 구하기
w = y - p    # 벡터 x에 직교인 y의 벡터 성분 w 구하기
print("x =", x)
print("y =", y)
print()
print("p =", p)
print("w =", w)

```

x= (2, -1, 3)
y= (4, -1, 2)

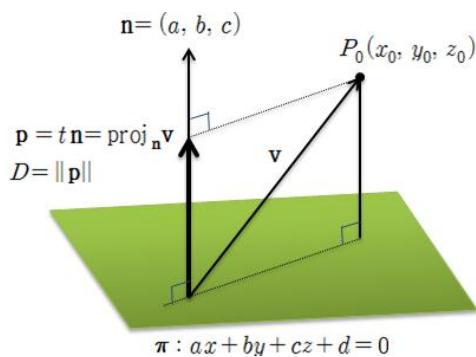
p= (15/7, -15/14, 45/14)

w= (13/7, 1/14, -17/14) [실습 : <http://sage.skku.edu/> 또는 <https://sagecell.sagemath.org/>]

정리. 점과 평면사이의 (최단) 거리 \mathbf{n} 위로의 \mathbf{v} 의 정사영(projection of \mathbf{v} onto \mathbf{n})의 노름

점 $P_0(x_0, y_0, z_0)$ 와 평면 $\pi : ax + by + cz + d = 0$ 사이의 거리 D 는 다음과 같다.

$$D = \frac{|ax_0 + by_0 + cz_0 + d|}{\sqrt{a^2 + b^2 + c^2}}$$



예제 5 점 $P(3, -1, 2)$ 에서 평면 $x + 3y - 2z - 6 = 0$ 에 이르는 거리 D 를 구하여라.

($\mathbf{n} = \text{vector}([1, 3, -2])$ 위로의 $\mathbf{V} = \text{vector}([3, -1, 2])$ 의 정사영(projection of \mathbf{V} onto \mathbf{n})의 노름)

```
v = vector([3, -1, 2])
n = vector([1, 3, -2])
d = -6
vn = v.inner_product(n)    # 벡터 v와 n의 내적 구하기,  $ax_0 + by_0 + cz_0$ 
nn = n.norm()    # 벡터 n의 노름 구하기
Dist = abs(vn + d)/nn    # 점과 평면사이의 거리 구하기,  $D = \frac{|ax_0 + by_0 + cz_0 + d|}{\sqrt{a^2 + b^2 + c^2}}$ 
print("P =", v)
print("n =", n)
print()
print(Dist)
```

```
P= (3, -1, 2)
n= (1, 3, -2)
```

```
5/7*sqrt(14)
```

따라서 점 $P(3, -1, 2)$ 에서 평면 $x + 3y - 2z - 6 = 0$ 에 이르는 거리는 $\frac{5}{7}\sqrt{14}$ 이다.

<http://matrix.skku.ac.kr/2018-album/LA-Sec-1-3-lab.html> (여기서 실습하세요!)



2. 선형연립방정식

2.1 선형연립방정식 (Linear System of Equations)

참고 동영상: <http://youtu.be/CiLn1F2pmvY> <http://youtu.be/AAUQvdjQ-qk>

실습 사이트: <http://matrix.skku.ac.kr/knowls/CLA-Week-2-Sec-2-1.html>

3강 동영상, 선형연립방정식, 행렬과 행렬식 https://youtu.be/qwQX_zPiICU (47:45)

정의. 선형연립방정식(system of linear equations)

(1) 일반적으로, 미지수 x_1, x_2, \dots, x_n 에 관한 유한개의 선형방정식의 모임

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &= b_m \end{aligned}$$

을 선형연립방정식(system of linear equations)이라고 한다. 만일 상수항 b_1, b_2, \dots, b_m 이 모두 0일 경우를 동차선형연립방정식(homogeneous system of linear equations, 동차선형방정식시스템)이라 한다.

(2) 선형연립방정식의 미지수 x_1, x_2, \dots, x_n 에 어떤 수 s_1, s_2, \dots, s_n 을 각각 대입하였을 때, 각 방정식이 모두 성립하면 (s_1, s_2, \dots, s_n) 을 이 선형연립방정식의 해 (solution)라고 한다. 예를 들어, 선형연립방정식

$$\begin{aligned} 4x_1 - x_2 + 3x_3 &= -1 \\ 3x_1 + x_2 + 9x_3 &= -4 \end{aligned} \tag{1}$$

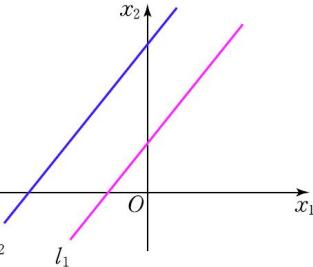
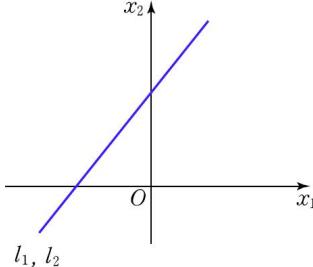
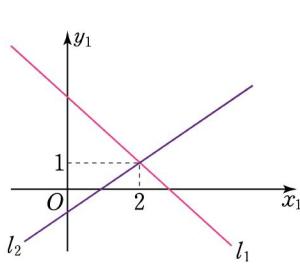
의 x_1, x_2, x_3 에 각각 1, 2, -1을 대입하면 두 방정식이 모두 성립하므로 (1, 2, -1)은 선형연립방정식 (1)의 해이다. 일반적으로 선형연립방정식의 해가 존재하는 경우를 **consistent(일관된)**한 연립방정식이라 하고, 해가 존재하지 않는 연립방정식은 **inconsistent** 한 연립방정식이라 한다.

■ (미지수가 2개인 선형연립방정식의) 해집합의 다양한 모습

$$(1) \begin{aligned} x_1 + x_2 &= 3 \\ x_1 - x_2 &= 1 \end{aligned}$$

$$(2) \begin{aligned} 2x_1 - x_2 &= -2 \\ -2x_1 + x_2 &= 2 \end{aligned}$$

$$(3) \begin{aligned} 2x_1 - x_2 &= -2 \\ -2x_1 + x_2 &= 4 \end{aligned}$$



일반적으로, 주어진 선형연립방정식은 다음 중 하나만(one and only)을 만족한다.

- (1) 유일한 해를 갖는다.
- (2) 무수히 많은 해를 갖는다.
- (3) 해를 갖지 않는다. (**inconsistent** 한 선형연립방정식)

정의.

첨가행렬(augmented matrix)

n 개의 미지수를 갖는 m 개의 일차방정식으로 이루어진 선형연립방정식

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &= b_m \end{aligned} \quad (3)$$

에 대하여

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

이라 하면 선형연립방정식 (3)은 행렬의 곱을 이용하여 다음과 같이 간단히 쓸 수 있다.

$$A\mathbf{x} = \mathbf{b}$$

이때 행렬 A 를 선형연립방정식 (3)의 계수행렬(coefficient matrix)이라 하며, A 에 \mathbf{b} 를 붙여서 만든 행렬

$$[A : \mathbf{b}] = \left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_m \end{array} \right]$$

을 선형연립방정식 (3)의 첨가행렬(augmented matrix)이라고 한다.

예제 1 다음 선형연립방정식을 행렬의 곱을 이용하여 나타내고, 첨가행렬을 구하여라.

$$x + y + 2z = 9$$

$$2x + 4y - 3z = 1$$

$$3x + 6y - 5z = 0$$

풀이 $A = \begin{bmatrix} 1 & 1 & 2 \\ 2 & 4 & -3 \\ 3 & 6 & -5 \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 9 \\ 1 \\ 0 \end{bmatrix}$

이라 할 때 $A\mathbf{x}=\mathbf{b}$ 이다. 그리고 첨가행렬은

$$[A : \mathbf{b}] = \left[\begin{array}{ccc|c} 1 & 1 & 2 & 9 \\ 2 & 4 & -3 & 1 \\ 3 & 6 & -5 & 0 \end{array} \right]. \quad \blacksquare$$

[실습 : <http://sage.skku.edu/>]

```
A = matrix([[1, 1, 2], [2, 4, -3], [3, 6, -5]])    # 3x3 행렬 입력
b = vector([9, 1, 0])      # 상수항 벡터 입력
print(A.augment(b))      # 첨가행렬
```

```
[ 1  1  2  9]
[ 2  4 -3  1]
[ 3  6 -5  0]
```

<http://matrix.skku.ac.kr/2018-album/LA-Sec-2-1-lab.html>



읽을거리

솔로몬 레프쉐츠(Solomon Lefschetz, 1884~1972)

– 양손을 잃고 수학을 시작한 수학자 –

러시아에서 태어난 솔로몬 레프쉐츠(Solomon Lefschetz, 1884~1972)는 프랑스의 고급기술자를 양성하는 에콜 상트랄 파리(École Centrale Paris)에서 전기공학을 전공한 엔지니어였다. 그 뒤 미국으로 이민을 가서 웨스팅하우스 전기(Westinghouse Electric Company)에 취직했으나, 1910년 실험 도중 변압기가 폭발하는 사고로 양손을 잃으면서 더 이상 일을 할 수 없게 되었다. 이 사고로 심한 우울증에 빠졌지만, 손을 반드시 쓰지 않고도 연구를 할 수 있는 수학을 공부하기로 결심하고 미국에 두 번째로



생긴 연구 중심대학이며 실베스터의 PostDoc이었던 Story가 수학과 교수로 부임한 신설 Clark 대학 대학원에 입학하여 Story의 지도하에 1911년 박사 학위를 취득한다. 그가 쓴 학위논문 주제는 어떤 점이 주어질 때 그 점을 특이점으로 갖는 대수곡선에 관한 대수기하학 분야의 문제였다. 레프쉐츠 자신과 수학계로서는 다행스럽게도 그는 수학, 특히 19세기에 생겨난 이래 20세기 초 활발한 발전 도상에 있던 위상수학이 자신의 적성에 맞는다는 것을 알았고, 푸앵카레(Jules Henri Poincaré, 1854~1912)나 브로우어(Luitzen Egbertus Jan Brouwer, 1881~1966) 등의 연구 결과를 확장하는 과정에서 오늘날 대수적 위상수학(algebraic topology)이라 불리는 분야의 기초를 닦는 데 기여했다. 지금도 위상수학교재에서 그의 이름이 붙은 정리나 개념들을 볼 수 있으며, 그가 쓴 대수적 위상수학 교과서는 고전으로 꼽힌다. 한편 1960년대에는 물리학자 파인만(Richard Feynman, 1918~1988)이 고안한 파인만 적분의 위상적 성질을 밝히기도 했다.

레프쉐츠는 1916년부터 1923년까지 캔자스 대학교 로렌스 캠퍼스 교수, 1924년 이후 프린스턴 고등연구소(IAS, 사진) 교수로 미국수학사에 큰 기여를 하였으며, 1935년부터 1936년에 걸쳐 미국 수학회(American Mathematical Society) 회장직을 역임했다.



<프린스턴고등연구소 (IAS)>

2.2 Gauss 소거법과 Gauss–Jordan 소거법

참고 동영상: <http://youtu.be/jnC66zvqHJI> <http://youtu.be/HSm69YigRr4>

- 선형연립방정식의 풀이: 소거법

$$\begin{aligned} \begin{cases} 2x + 3y = 1 \\ x - 2y = 4 \end{cases} &\Rightarrow \begin{cases} 2x + 3y = 1 \\ 2x - 4y = 8 \end{cases} \Rightarrow \begin{cases} 2x + 3y = 1 \\ 7y = -7 \end{cases} \\ \Rightarrow \begin{cases} 2x + 3y = 1 \\ y = -1 \end{cases} &\Rightarrow \begin{cases} 2x = 4 \\ y = -1 \end{cases} \Rightarrow \begin{cases} x = 2 \\ y = -1 \end{cases} \end{aligned}$$

- 위의 소거법에서 행한 연산: 선형연립방정식의 해집합을 바꾸지 않는다.

(1) 두 식을 교환한다. $R_i \leftrightarrow R_j$

(2) 한 식에 0 아닌 실수를 곱한다. kR_i

(3) 한 식에 0 아닌 실수배를 하여 다른 식에 더한다. $kR_i + R_j$

이를 기본행 연산(ERO, Elementary Row Operations)이라 한다.

정의. 기본행 연산(elementary row operation, ERO)

$m \times n$ 행렬 A 에 관한 다음의 연산을 기본행 연산(elementary row operation, ERO)이라고 한다.

E1: A 의 두 행 i 행과 j 행을 서로 바꾼다. $R_i \leftrightarrow R_j$

E2: A 의 i 행에 0이 아닌 상수 k 를 곱한다. kR_i

E3: A 의 i 행을 k 배 하여 j 행에 더한다. $kR_i + R_j$

정의. 행 사다리꼴(row echelon form, REF)

$m \times n$ 행렬 E 가 다음 3가지 성질을 만족할 때, 행 사다리꼴(row echelon form, REF)이라고 한다.

- (1) 성분이 모두 0인 행이 존재하면 그 행은 행렬의 맨 아래에 위치한다.
- (2) 각 행에서 처음으로 나타나는 0이 아닌 성분은 1이다. 이때 이 1을 그 행의 선행성분 (leading entry, leading 1)이라고 한다.
- (3) i 행과 $(i+1)$ 행 모두에 선행성분이 존재하면 $(i+1)$ 행의 선행성분은 i 행의 선행성분보다 오른쪽에 위치한다.

또 행 사다리꼴 행렬 E 가 다음 4번째 성질을 추가로 만족하면 E 를 기약 행 사다리꼴(reduced row echelon form, RREF)이라고 한다.

(4) 선행성분(leading entry in row)을 포함하는 열의 선행선분 외의 성분은 모두 0이다.

실습 사이트: <http://matrix.skku.ac.kr/knowls/CLA-Week-2-Sec-2-2.html> (실습하세요!)

예제 2 행렬 A 의 기약 행 사다리꼴(reduced row echelon form, RREF)를 구하시오.

$$A = \begin{bmatrix} 1 & 1 & 1 & 4 & 4 \\ 2 & 3 & 4 & 9 & 16 \\ -2 & 0 & 3 & -7 & 11 \end{bmatrix}$$

[실습 : <http://sage.skku.edu/>]

```
A = matrix([[1, 1, 1, 4, 4],
           [2, 3, 4, 9, 16],
           [-2, 0, 3, -7, 11]])    # 3x5 행렬 입력
print("A =")
print(A)
print()
print("RREF(A) =")
print(A.rref())  # A의 RREF 구하기
```

```
A=
[ 1  1  1  4  4]
[ 2  3  4  9 16]
[-2  0  3 -7 11]      [ 실습 : http://sage.skku.edu/ ]
```

```
RREF(A)=
[ 1  0  0  2 -1]
[ 0  1  0  3  2]
[ 0  0  1 -1  3]
```

정리. 행동치인 선형연립방정식은 해집합이 같다

첨가행렬이 행동치인 두 선형연립방정식은 동치이다(즉, 해집합이 같다).

- Gauss 소거법: 선형연립방정식의 첨가행렬을 REF로 변형하여 푸는 방법이다.
- Gauss–Jordan 소거법: 선형연립방정식의 첨가행렬을 RREF로 변형하여 푸는 방법이다.

예제 3 Gauss–Jordan 소거법을 이용하여 다음 연립방정식의 해를 구하여라.

$$\begin{array}{l} 2x + 4y + 6z = 18 \\ 2x - y + z = 8 \\ 3x \quad - z = 3 \end{array}$$

```
A = matrix([[2, 4, 6], [2, -1, 1], [3, 0, -1]])
b = vector([18, 8, 3])
print("[A: b] =")
print(A.augment(b))
print()
print("RREF([A: b]) =")
print(A.augment(b).rref()) # 행렬 A와 벡터 b의 첨가행렬의 RREF 구하기
print()
print("x =", A.solve_right(b)) # .solve_right( )를 이용하여 구할 수도 있다.
```

```
[A: b] =
[ 2  4  6 18]
[ 2 -1  1  8]
[ 3  0 -1  3]

RREF([A: b]) = [ 1  0  0  2]
                [ 0  1  0 -1]
                [ 0  0  1  3]
x = (2, -1, 3) # x= (x, y, z) = (2,-1,3)
```

따라서 주어진 선형연립방정식의 해는 $x = 2$, $y = -1$, $z = 3$ 이다. 또한, `A.solve_right(b)` 명령어를 이용하여 해를 구할 수도 있다.

예제 4 Gauss–Jordan 소거법을 이용하여 다음 연립방정식의 해를 하나 구하여라.

$$\begin{aligned}
 x_1 + 3x_2 - 2x_3 &+ 2x_5 = 0 \\
 2x_1 + 6x_2 - 5x_3 - 2x_4 + 4x_5 - 3x_6 &= -1 \\
 5x_3 + 10x_4 &+ 15x_6 = 5 \\
 2x_1 + 6x_2 &+ 8x_4 + 4x_5 + 18x_6 = 6
 \end{aligned}$$

```

A = matrix([[1, 3, -2, 0, 2, 0], [2, 6, -5, -2, 4, -3],
           [0, 0, 5, 10, 0, 15], [2, 6, 0, 8, 4, 18]])
b = vector([0, -1, 5, 6])
print("[A: b] =")
print(A.augment(b))
print()
print("RREF([A: b]) =")
print(A.augment(b).rref()) # 행렬 A와 벡터 b의 첨가행렬의 RREF 구하기
print()
print("x =", A.solve_right(b)) # .solve_right( )를 이용하여 구할 수도 있다.

```

```

[A:b]=
[ 1 3 -2 0 2 0 0]
[ 2 6 -5 -2 4 -3 -1]
[ 0 0 5 10 0 15 5]
[ 2 6 0 8 4 18 6]

```

```

RREF([A:b])=
[ 1 3 0 4 2 0 0]
[ 0 0 1 2 0 0 0]
[ 0 0 0 0 0 1 1/3]
[ 0 0 0 0 0 0 0]

```

(0, 0, 0, 0, 0, 1/3) # 이것은 하나의 해, RREF의 마지막 행이 영이면 무수히 많은 해가 존재한다!

따라서 주어진 선형연립방정식의 해는 자유변수 r_1, r_2, r_3 에 대하여 $x_1 = -3r_1 - 4r_2 - 2r_3$, $x_2 = r_1$, $x_3 = -2r_2$, $x_4 = r_2$, $x_5 = r_3$, $x_6 = \frac{1}{3}$ 이다. 이와 같이 무수히 많은 해를 갖는 선형연립방정식인 경우에, A.solve_right(b) 명령어는 모든 자유변수에 0을 대입한 해를 찾아준다.

<http://matrix.skku.ac.kr/2018-album/LA-Sec-2-2-lab.html> (실습하세요!)

3. 행렬과 행렬식

3.1 행렬 연산

참고 동영상: <http://youtu.be/jnC66zvqHJI> <http://youtu.be/HSm69YigRr4>

실습 사이트: <http://matrix.skku.ac.kr/knowls/CLA-Week-2-Sec-2-2.html>

정의. 합(vector sum), 스칼라배(scalar multiple)

두 행렬 $A = [a_{ij}]_{m \times n}$, $B = [b_{ij}]_{m \times n}$ 와 실수 k 에 대하여 A 와 B 의 합(vector sum) $A + B$ 와 A 의 스칼라배(scalar multiple) kA 를 다음과 같이 정의한다.

$$A + B = [a_{ij} + b_{ij}]_{m \times n}, \quad kA = [ka_{ij}]_{m \times n}$$

- 행렬 덧셈이 정의되려면 두 행렬의 크기가 같아야 한다.

예제 1 $A = \begin{bmatrix} 1 & 2 & -4 \\ -2 & 1 & 3 \end{bmatrix}$, $B = \begin{bmatrix} 0 & 1 & 4 \\ -1 & 3 & 1 \end{bmatrix}$, $C = \begin{bmatrix} 1 & 1 \\ 2 & 2 \end{bmatrix}$ 일 때, $A + B$, $2A$, $(-1)C$ 는?

[실습 : <http://sage.skku.edu/>]

```
A = matrix([[1, 2, -4], [-2, 1, 3]])
B = matrix([[0, 1, 4], [-1, 3, 1]])
C = matrix([[1, 1], [2, 2]])
print("A + B =")
print(A + B) # 행렬의 덧셈
print()
print("2*A =")
print(2*A) # 행렬의 스칼라배
print()
print("(-1)*C =")
print((-1)*C) # 행렬의 스칼라배
```

```
A+B =
[ 1  3  0]
```

$$[-3 \ 4 \ 4]$$

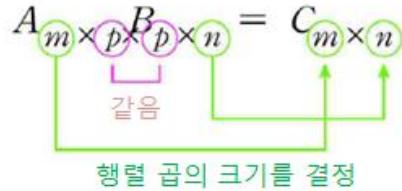
$$\begin{aligned} 2*A &= \\ [2 &\ 4 \ -8] \\ [-4 &\ 2 \ 6] \\ (-1)*C &= \\ [-1 &\ -1] \\ [-2 &\ -2] \end{aligned}$$

정의. 행렬의 곱(product), AB

두 행렬 $A = [a_{ij}]_{m \times p}$, $B = [b_{ij}]_{p \times n}$ 에 대하여 A 와 B 의 곱(product) AB 를 다음과 같이 정의한다.

$$AB = [c_{ij}]_{m \times n}$$

$$\text{여기서, } c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \dots + a_{ip}b_{pj} = \sum_{k=1}^p a_{ik}b_{kj} \quad (1 \leq i \leq m, 1 \leq j \leq n)$$



- $A = [a_{ij}]_{m \times p}$, $B = [b_{ij}]_{p \times n}$ 라 하고, A 의 i 번째 행을 $A_{(i)}$, A 의 j 번째 열을 $A^{(j)}$ 로 표기하자. 그러면

$$C = AB = \begin{bmatrix} A_{(1)} \\ A_{(2)} \\ \vdots \\ A_{(m)} \end{bmatrix} \left[B^{(1)} \ B^{(2)} \ \dots \ B^{(n)} \right] = \begin{bmatrix} A_{(1)}B^{(1)} & A_{(1)}B^{(2)} & \dots & A_{(1)}B^{(n)} \\ \vdots & & & \vdots \\ A_{(m)}B^{(1)} & \dots & & A_{(m)}B^{(n)} \end{bmatrix}_{m \times n}$$

$$\therefore c_{ij} = A_{(i)}B^{(j)} = \begin{bmatrix} a_{i1} & \dots & a_{ip} \end{bmatrix} \begin{bmatrix} b_{1j} \\ \vdots \\ b_{pj} \end{bmatrix} = a_{i1}b_{1j} + a_{i2}b_{2j} + \dots + a_{ip}b_{pj} = \sum_{k=1}^p a_{ik}b_{kj}$$

- 앞 행렬 A 의 행벡터와 뒤 행렬 B 의 열벡터의 내적(inner product)이 AB 행렬의 해당 위치에 대응하는 성분과 일치한다. [King Sejong's 법칙, 세종대왕]

[의 기역 법칙]

$$AB = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1p} \\ \vdots & \vdots & & \vdots \\ a_{i1} & a_{i2} & \cdots & a_{ip} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mp} \end{bmatrix} \begin{bmatrix} b_{11} & \cdots & b_{1j} & \cdots & b_{1n} \\ b_{21} & \cdots & b_{2j} & \cdots & b_{2n} \\ \vdots & & \vdots & & \vdots \\ b_{p1} & \cdots & b_{pj} & \cdots & b_{pn} \end{bmatrix} = [c_{ij}] = [A_{(i)}B^{(j)}]_{m \times n}$$

예제 2 행렬 $A = \begin{bmatrix} 1 & 2 & -1 \\ 3 & 1 & 0 \end{bmatrix}$, $B = \begin{bmatrix} -2 & 1 \\ 0 & -3 \\ 2 & 1 \end{bmatrix}$ 에 대하여 AB 를 구하여라.

```
A = matrix([[1, 2, -1], [3, 1, 0]])
B = matrix([[-2, 1], [0, -3], [2, 1]])
print(A*B) # 행렬의 곱(곱하기 기호(*)를 빼먹지 않도록!)
```

$$\begin{bmatrix} -4 & -6 \\ -6 & 0 \end{bmatrix} \# AB = [c_{ij}] = [A_{(i)}B^{(j)}]_{2 \times 2}$$

☞ 행렬의 곱을 이용하면 선형연립방정식을 쉽게 표현할 수 있다. 아래 선형연립방정식

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1$$

$$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2$$

⋮

$$a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m$$

에서 계수와 미지수, 상수항을 각각 $A = [a_{ij}]_{m \times n}$, $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$ 이라 하면,

다음과 같이 나타낼 수 있다.

$$A\mathbf{x} = \mathbf{b} \Leftrightarrow x_1A^{(1)} + x_2A^{(2)} + \cdots + x_nA^{(n)} = \mathbf{b}$$

선형연립방정식의 일반적 형식

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \end{cases}$$

선형연립방정식의 벡터 형식

$$x_1 \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix} + x_2 \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{m2} \end{bmatrix} + \cdots + x_n \begin{bmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

정의. 영행렬(zero matrix)

(1) 영행렬(zero matrix)은 성분이 모두 0인 행렬로 O (또는 $O_{m \times n}$)로 나타낸다.

$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, [0] \dots$$

(2) 주대각선성분이 모두 1이고 나머지 성분은 모두 0인 n 차의 정사각행렬을 단위행렬(identity matrix)이라 하고, I_n 으로 나타낸다.

$$I_n = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

(3) A 가 n 차의 정사각행렬일 때, A 의 거듭제곱을 다음과 같이 정의한다.

$$A^0 = I_n, A^k = AA \cdots A (k \text{개})$$

(4) 행렬 $A = [a_{ij}]_{m \times n}$ 에 대하여 A 의 전치행렬(transpose)을 A^T 로 나타내고 다음과 같이 정의한다.

$$A^T = [a_{ij}']_{n \times m}, a_{ij}' = a_{ji} \quad (1 \leq i \leq n, 1 \leq j \leq m)$$

- 전치행렬은 원 행렬의 행과 열을 바꾸어 얻어진 행렬이다.

예제 3 다음 행렬들의 전치행렬을 각각 구하여라.

$$A = \begin{bmatrix} 1 & -2 & 3 \\ 4 & 5 & 0 \end{bmatrix}, B = \begin{bmatrix} 1 & 2 & -4 \\ 3 & -1 & 2 \\ 0 & 5 & 3 \end{bmatrix}$$

```
A = matrix([[1, -2, 3], [4, 5, 0]])
B = matrix([[1, 2, -4], [3, -1, 2], [0, 5, 3]])
C = matrix([[5, 4], [-3, 2], [2, 1]])
D = matrix([[3, 0, 1]])
print("A^T =")
print(A.transpose()) # 행렬의 전치행렬 A.transpose() 구하기
print()
print("B^T =")
print(B.transpose())
```

```
A^T =
[ 1  4]
[-2  5]
[ 3  0]      #    $A^T = [a_{ij}']_{3 \times 2}$ 
```

```
B^T =
[ 1  3  0]
[ 2 -1  5]
[-4  2  3]      #    $B^T = [b_{ij}']_{3 \times 3}$ 
```

예제 3-1 SageMath cell 서버 <http://sage.skku.edu/>에서 임의로 행렬을 하나 생성하고, 그 행렬의 전치행렬을 구하라.

```
A = random_matrix(ZZ, 7, 12) # 임의의 7x12 정수(ZZ) 행렬 A를 생성!
print("A =")
print(A)
print()
print("A^T =")
print(A.transpose()) # 생성한 행렬 A의 전치행렬 A.transpose() 구하기
```

```
A =
[ -1   1  -2   1  -1   1  -3   1   3  -1  -1   3]
```

```
[ 1 -1  1  2  0  0 12  1 -25  0  0 -3]
[ -1 -1  1  1 -1 -1  0 -7 -8  0 -1  2]
[ -4  1  0 -1 -3 -12  1 -1  0 -2  1 -1]
[ -1  0 -1 -1  0 -2  1 -1 -1 -1  1  1]
[ -3 -1 -1  0  1 -1  1 -1 -36  3 -7 30]
[ -3 -1 -1  0 -2 -1 -1 -7  1  0  2 -1]
```

```
A^T =
[ -1  1 -1 -4 -1 -3 -3]
[ 1 -1 -1  1  0 -1 -1]
[ -2  1  1  0 -1 -1 -1]
[ 1  2  1 -1 -1  0  0]
[ -1  0 -1 -3  0  1 -2]
[ 1  0 -1 -12 -2 -1 -1]
[ -3 12  0  1  1  1 -1]
[ 1  1 -7 -1 -1 -1 -7]
[ 3 -25 -8  0 -1 -36  1]
[ -1  0  0 -2 -1  3  0]
[ -1  0 -1  1  1 -7  2]
[ 3 -3  2 -1  1  30 -1]      # 12×7 행렬 A^T !
```

```
import numpy as np
A = np.random.randint(5, size=(5, 7))    # 임의의 (0~4 사이의 수만으로 이루어진) 5x7 자연수 행렬
A를 생성!
print("A =")
print(A)
print()
print("A^T =")
print(A.transpose()) # 생성한 행렬 A의 전치행렬 A.transpose() 구하기
```

```
A =
[[0 2 1 1 4 1 2]
 [2 2 3 1 3 1 1]
 [4 3 2 0 4 1 2]
 [0 4 2 3 4 2 0]
 [2 4 3 3 3 4 0]]
```

```

A^T =
[[0 2 4 0 2]
 [2 2 3 4 4]
 [1 3 2 2 3]
 [1 1 0 3 3]
 [4 3 4 4 3]
 [1 1 1 2 4]
 [2 1 2 0 0]]      # 7×5 행렬 A^T !

```

정리. 전치행렬의 성질

두 행렬 A, B 와 임의의 스칼라 k 에 대하여 다음이 성립한다.

- (1) $(A^T)^T = A$
- (2) $(A + B)^T = A^T + B^T$
- (3) $(AB)^T = B^T A^T$
- (4) $(kA)^T = kA^T$

정의. 대각합(trace)

행렬 $A = [a_{ij}]_{n \times n}$ 의 대각합(trace)은 $\text{tr}(A) = a_{11} + a_{22} + \dots + a_{nn} = \sum_{i=1}^n a_{ii}$ 이다.

<http://matrix.skku.ac.kr/2018-album/LA-Sec-3-1-lab.html> (실습하세요!)

3.2 역행렬

참고 동영상: <http://youtu.be/GCKM2VIU7bw> <http://youtu.be/yeCUPdRx7Bk>

실습 사이트: <http://matrix.skku.ac.kr/knowls/CLA-Week-3-Sec-3-2.html>

정의. 역행렬

n 차의 정사각행렬 A 에 대하여 다음을 만족하는 행렬 B 가 존재하면 A 는 가역 (invertible, nonsingular)이라고 한다.

$$AB = I_n = BA$$

이때 B 를 A 의 역행렬(inverse matrix)이라고 하며, 이러한 B 가 존재하지 않으면 A 는 비가역(noninvertible, singular)이라고 한다.

예제 4 행렬 $A = \begin{bmatrix} 1 & 4 & 3 \\ 2 & 5 & 6 \\ 0 & 0 & 0 \end{bmatrix}$ 는 가역인가 비가역인가?

[실습 : <http://sage.skku.edu/>]

```
A = matrix([[1, 4, 3], [2, 5, 6], [0, 0, 0]])
A.is_invertible() # 행렬이 가역인지 확인 A.is_invertible()
```

False

따라서 행렬 $A = \begin{bmatrix} 1 & 4 & 3 \\ 2 & 5 & 6 \\ 0 & 0 & 0 \end{bmatrix}$ 는 비가역행렬이다.

정리. 가역행렬(invertible matrix)의 성질 1

n 차의 정사각행렬 A, B 가 가역이고 k 가 0이 아닌 스칼라일 때, 다음이 성립한다.

- (1) A^{-1} 은 가역이고, $(A^{-1})^{-1} = A$ 이다.
- (2) AB 는 가역이고, $(AB)^{-1} = B^{-1}A^{-1}$ 이다.
- (3) kA 는 가역이고, $(kA)^{-1} = \frac{1}{k}A^{-1}$ 이다.
- (4) A^n 도 가역이고, $(A^n)^{-1} = A^{-n} = (A^{-1})^n$

정리. 가역행렬(invertible matrix)의 성질 2

만일 A 가 가역행렬이면, A^T 도 가역행렬이고 다음이 성립한다.

$$(A^T)^{-1} = (A^{-1})^T$$

예제 5 행렬 $A = \begin{bmatrix} 3 & 5 \\ 1 & 2 \end{bmatrix}$ 의 역행렬을 구하여라.

풀이 $A^{-1} = \frac{1}{ad-bc} \begin{bmatrix} d-b & -b \\ -c & a \end{bmatrix} = \begin{bmatrix} \frac{d}{ad-bc} & \frac{-b}{ad-bc} \\ \frac{-c}{ad-bc} & \frac{a}{ad-bc} \end{bmatrix} = \begin{bmatrix} 2 & -5 \\ -1 & 3 \end{bmatrix}$

```

A = matrix(2, 2, [3, 5, 1, 2])
print("A =")
print(A)
print()
print("inverse of A =")
print(A.inverse())  # A의 역행렬 구하기, 형식은 A.inverse()

```

A =
[3 5]
[1 2]

inverse of A =
[2 -5]
[-1 3]

<http://matrix.skku.ac.kr/2018-album/LA-Sec-3-2-lab.html> (실습하세요!)

3.3 기본행렬

참고 동영상: <http://youtu.be/GCKM2VIU7bw> <http://youtu.be/oQ2m6SSSquc>

실습 사이트: <http://matrix.skku.ac.kr/knowls/CLA-Week-3-Sec-3-3.html>

정의. 기본행연산(elementary row operation, ERO)

I_n 에 기본행연산(elementary row operation, ERO)을 한 번 적용해서 얻은 행렬을 기본행렬(elementary matrices)이라 한다. 그리고 치환(permutation)행렬은 I_n 의 행들을 교환하여 얻어진 행렬이다.

- 세 가지 기본행 연산에 대응하는 기본행렬들은 다음과 같다.

(1) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$: 두 행을 교환한다. $R_i \leftrightarrow R_j$

(2) $\begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$: 한 행에 영 아닌 상수배를 해서 다른 행에 더한다. $kR_j + R_i$

(3) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 1 \end{bmatrix}$: 한 행에 영 아닌 상수배를 한다. kR_i

주의!! Sage의 index는 0부터 시작한다.

elementary_matrix(n, row1=i, row2=j): i행과 j행을 교환

E1 = elementary_matrix(4, row1=1, row2=2) # 기본행렬 $r2 \leftrightarrow r3$

elementary_matrix(n, row1=i, scale=m): i행에 m을 곱한다.

E2 = elementary_matrix(4, row1=2, scale=-3) # 기본행렬 $(-3)r3$

elementary_matrix(n, row1=i, row2=j, scale=m): j행에 m을 곱하여 i행에 더함

E3 = elementary_matrix(4, row1=0, row2=3, scale=7) # 기본행렬 $7r4 + r1$

```
print("E1 =")
print(E1)
print()
print("E2 =")
print(E2)
print()
print("E3 =")
print(E3)
```

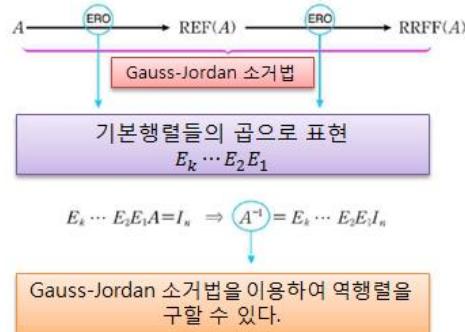
```
E1 =
[1 0 0 0]
[0 0 1 0]
[0 1 0 0]
[0 0 0 1]
```

```
E2 =
[ 1  0  0  0]
[ 0  1  0  0]
[ 0  0 -3  0]
[ 0  0  0  1]
```

```
E3 =
[1 0 0 7]
[0 1 0 0]
[0 0 1 0]
[0 0 0 1]
```

정리. 역행렬 구하는 방법 1

$$[A : I_n] \rightarrow [I_n : A^{-1}]$$



[단계 1] 주어진 행렬 A 에 단위행렬 I_n 을 첨가하여 $n \times 2n$ 행렬 $[A : I_n]$ 을 만든다.

[단계 2] 단계 1에서 만든 행렬 $[A : I_n]$ 의 RREF를 구한다.

[단계 3] 단계 2에서 얻어진 RREF를 $[C : D]$ 라고 하면 다음이 성립한다.

- (1) $C = I_n$ 이면 $D = A^{-1}$ 이다.
- (2) $C \neq I_n$ 이면 A 는 비가역이고 A^{-1} 은 존재하지 않는다.

예제 6 첨가행렬 $[A : I]$ 의 RREF를 이용하여 행렬 A 의 역행렬을 구하여라.

$$A = \begin{bmatrix} 1 & -1 & 2 \\ -1 & 0 & 2 \\ -6 & 4 & 11 \end{bmatrix}$$

```
A = matrix([[1, -1, 2], [-1, 0, 2], [-6, 4, 11]])
I = identity_matrix(3)
print("A =")
print(A)
```

```

print()
AI = A.augment(I).rref()    # 첨가행렬 [A : I] 의 RREF 구하기
print("RREF[A: I] =")
print(AI)
print()

A2 = AI.submatrix(0, 3, 3, 3)  # A.submatrix(a, b, c, d)
# 행렬의 (a+1, b+1) 성분부터 c개의 행, d개의 열로 이루어진 부분행렬 구하기
print("inverse of A =")
print(A2)

```

```

A =
[ 1 -1  2]
[-1  0  2]
[-6  4 11]

RREF[A: I] =      [   1     0     0   8/15 -19/15   2/15]
                  [   0     1     0   1/15 -23/15   4/15]
                  [   0     0     1   4/15 -2/15   1/15]

```

```

inverse of A =
[ 8/15 -19/15  2/15]
[ 1/15 -23/15  4/15]
[ 4/15 -2/15  1/15]

```

<http://matrix.skku.ac.kr/2018-album/LA-Sec-3-3-lab.html>
[\(실습하세요!\)](http://matrix.skku.ac.kr/2014-Album/MC-2.html)

3.4 선형연립방정식의 해집합, 특수행렬

참고 동영상: http://youtu.be/daIxHJBHL_g http://youtu.be/O0TPCpKW_eY
<http://youtu.be/jLh77sZOaM8>

실습 사이트: <http://matrix.skku.ac.kr/knou-knowls/CLA-Week-4-Sec-3-5.html>
<http://matrix.skku.ac.kr/knou-knowls/CLA-Week-4-Sec-3-6.html>

정리. 행렬의 가역성과 선형연립방정식의 해 사이의 관계

n 차의 정사각행렬 A 가 가역이고 \mathbf{b} 가 \mathbb{R}^n 의 벡터일 때, 연립방정식 $A\mathbf{x} = \mathbf{b}$ 는 유일한 해 $\mathbf{x} = A^{-1}\mathbf{b}$ 를 갖는다.

예제 7 역행렬을 이용하여 다음 연립방정식 $A\mathbf{x} = \mathbf{b}$ 의 해를 구하여라.

$$\begin{aligned}x_1 + 2x_2 + 3x_3 &= 1 \\2x_1 + 5x_2 + 3x_3 &= 3 \\x_1 + 8x_3 &= -1\end{aligned}$$

```
A = matrix(3, 3, [1, 2, 3, 2, 5, 3, 1, 0, 8])
b = vector([1, 3, -1])
print("x =", A.inverse()*b) # 역행렬을 이용한 연립방정식의 해 구하기
print()
print("x =", A.solve_right(b)) # .solve_right( )를 이용하여 구할 수도 있다.
```

$\mathbf{x} = (-1, 1, 0)$

$\mathbf{x} = (-1, 1, 0)$ [실습 : <http://sage.skku.edu/> 또는 <https://sagecell.sagemath.org/>]

따라서 주어진 선형연립방정식의 해는 $x_1 = -1, x_2 = 1, x_3 = 0$ 이다.

<http://matrix.skku.ac.kr/2018-album/LA-Sec-3-5-lab.html> (실습하세요!)

정의. 대각선행렬(diagonal matrix), 스칼라행렬(scalar matrix)

(1) 대각선행렬(diagonal matrix): 주대각선성분 이외의 모든 성분이 0인 정사각행렬. 주대각선성분이 $a_{11}, a_{22}, \dots, a_{nn}$ 인 대각선행렬 $\text{diag}(a_{11}, a_{22}, \dots, a_{nn})$ 은 다음과 같아 쓴다.

$$\text{diag}(a_{11}, a_{22}, \dots, a_{nn}) = \begin{bmatrix} a_{11} & & & \\ & a_{22} & & \\ & & \ddots & \\ & & & a_{nn} \end{bmatrix}$$

(2) 단위행렬(identity matrix): 주대각선성분이 모두 1인 행렬 I_n

(3) 스칼라행렬(scalar matrix): kI_n

$$I_n = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix}, \quad kI_n = \begin{bmatrix} k & & & \\ & k & & \\ & & \ddots & \\ & & & k \end{bmatrix}$$

예제 8 다음 대각선행렬 H 와 단위행렬 K , 영행렬 J 를 구하여라.

$$H = \begin{bmatrix} -3 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad K = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad J = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

```
H = diagonal_matrix([-3, -2, 1])    # 대각선행렬. diagonal_matrix([d1, d2, d3])
K = identity_matrix(3)    # 단위행렬. identity_matrix(n)
J = matrix(2, 2, 0)    # 영행렬. matrix(m, n, 0)
print("H =")
print(H)
print()
print("K =")
print(K)
print()
print("J =")
print(J)
```

```
H=
[-3  0  0]
[ 0 -2  0]
[ 0  0  1]
```

```
K=
[1 0 0]
[0 1 0]
[0 0 1]
```

```
J=
[0 0]
[0 0]
```

정의. 대칭행렬(symmetric matrix)

- (1) 정사각행렬 A 가 $A^T = A$ 를 만족하면 A 를 대칭행렬(symmetric matrix)이라고 하고, $A^T = -A$ 를 만족하면 반대칭행렬(skew-symmetric matrix)이라고 한다.
- (2) 하삼각행렬(lower triangular matrix): 주대각선 위의 모든 성분이 0인 정사각행렬

상삼각행렬(upper triangular matrix): 주대각선 아래의 모든 성분이 0인 정사각행렬

일반적인 4×4 삼각행렬은 다음과 같은 형태이다.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{bmatrix} : \text{상삼각행렬} \quad \begin{bmatrix} a_{11} & 0 & 0 & 0 \\ a_{21} & a_{22} & 0 & 0 \\ a_{31} & a_{32} & a_{33} & 0 \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} : \text{하삼각행렬}$$

예제 9 행렬 A 는 대칭행렬이고, 행렬 B 는 반대칭행렬임을 보여라.

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 1-2 \\ -1 & 0 & 3 \\ 2-3 & 0 \end{bmatrix}$$

```
A = matrix([[1, 2, 3], [2, 4, 5], [3, 5, 6]])
B = matrix([[0, 1, -2], [-1, 0, 3], [2, -3, 0]])
print(bool(A == A.transpose())) # A가 대칭인지 확인
print(bool(B == -B.transpose())) # B가 반대칭인지 확인
```

```
True
True
```

따라서 $A = A^T$ 이므로 A 는 대칭행렬이고, $B = -B^T$ 이므로 B 는 반대칭행렬이다.

<http://matrix.skku.ac.kr/2018-album/LA-Sec-3-6-lab.html> (실습하세요!)

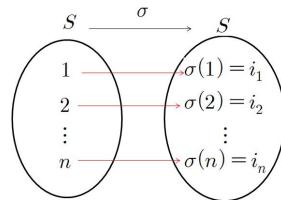
3.5 행렬식

참고 동영상: <http://youtu.be/DM-q2ZuQtI0> <http://youtu.be/Vf8LlkKKHgg>

실습 사이트: <http://matrix.skku.ac.kr/knowls/CLA-Week-5-Sec-4-1.html>

정의. 치환(permutation, 순열)

- (1) 자연수의 집합 $S = \{1, 2, \dots, n\}$ 의 치환(permutation, 순열)이란 S 에서 S 로의 일대일 대응함수이다.



- 앞으로 치환을 간단히 $\sigma = (\sigma(1) \ \sigma(2) \ \dots \ \sigma(n)) = (i_1 \ i_2 \ \dots \ i_n)$ 로 나타낸다. 치환 σ 는 일대일대응이므로 치역 $\{i_1, i_2, \dots, i_n\}$ 은 $1, 2, \dots, n$ 의 숫자를 일렬로 배열하는 것에 지나지 않는다. 따라서 $S = \{1, 2, \dots, n\}$ 의 치환은 모두 $n!$ 개이다. 집합 S 의 모든 치환의 집합을 S_n 으로 표시한다.

- (2) 치환 $(j_1 \ j_2 \ \dots \ j_n)$ 에서 반전(inversion)이란 큰 자연수가 작은 자연수보다 더 왼쪽에 먼저 나타나는 경우를 말한다. 예를 들어 아래 그림의 치환 $(1 \ 4 \ 2 \ 3)$ 에서 4는 2보다 더 왼쪽에 있으므로 $(4 \ 2)$ 에서 반전이 일어났다. 마찬가지로 $(4 \ 3)$ 에서도 반전이 일어났다.



- (3) 치환이 가진 반전의 총 개수가 짝수이면 이 치환은 짹치환(even permutation), 홀수이면 홀치환(odd permutation)이라고 한다.

예제 10 S_5 의 치환 $\sigma = (5 \ 1 \ 2 \ 4 \ 3)$ 의 반전의 총수를 구하여 짹치환인지 홀치환인지 결정하여라.

```
print(Permutation([5, 1, 2, 4, 3]).inversions()) # 반전이 일어난 부분 찾기
print(Permutation([5, 1, 2, 4, 3]).number_of_inversions()) # 반전의 개수 구하기
print(Permutation([5, 1, 2, 4, 3]).is_even()) # 짝치환인지 확인
```

`[[0, 1], [0, 2], [0, 3], [0, 4], [3, 4]]` # 주의!! index는 0부터 시작

5

False

따라서 S_5 의 치환 $\sigma = (5 \ 1 \ 2 \ 4 \ 3)$ 은 짝치환이 아니므로 홀치환이다.

정의. 치환(permutation, 순열), 행렬식

(1) S_n 의 각 치환을 $+1$ 또는 -1 이라는 수에 대응시키는 부호화 함수(signature function) $\text{sgn} : S_n \rightarrow \{+1, -1\}$ 을 다음과 같이 정의한다.

$$\text{sgn}(\sigma) = \begin{cases} +1 & (\sigma : \text{짝치환}) \\ -1 & (\sigma : \text{홀치환}) \end{cases}$$

(2) 행렬 $A = [a_{ij}]$ 가 n 차의 정사각행렬일 때, A 의 행렬식을 $\det(A)$ 또는 $|A|$ 로 나타내고 다음과 같이 정의한다.

$$\det(A) = \sum_{\sigma \in S_n} \text{sgn}(\sigma) a_{1\sigma(1)} a_{2\sigma(2)} \cdots a_{n\sigma(n)}$$

- 정의에 의하여 1차 정사각행렬 $A = [a]$ 의 행렬식은 $\det(A) = a$ 이다.
- 행렬식의 각 항 $\text{sgn}(\sigma) a_{1\sigma(1)} a_{2\sigma(2)} \cdots a_{n\sigma(n)}$ 은 행렬 A 의 행과 열에서 중복됨 없이 하나씩 뽑아서 곱한 후 대응되는 치환의 부호를 붙인 것이다.

예제 11 행렬 B 의 행렬식을 구하여라.

$$B = \begin{bmatrix} 1 & 2 & 3 \\ -4 & 5 & 6 \\ 7 & -8 & 9 \end{bmatrix}$$

```
B = matrix([[1, 2, 3], [-4, 5, 6], [7, -8, 9]])
print(B.det()) # 행렬식 구하기
```

240

정리. 가역행렬

- (1) A 가 가역행렬일 필요충분조건은 $\det A \neq 0$ 이다.
- (2) 두 행렬 A, B 가 n 차의 정사각행렬일 때, $|AB| = |A||B|$ 성립한다.
- (3) 행렬 A 가 가역이면 $|A| \neq 0$ 이고, $|A^{-1}| = \frac{1}{|A|}$ 성립한다.

```
A = random_matrix(QQ, 3, 3)
B = random_matrix(QQ, 3, 3)
AB = A*B
print("|AB| = |A| |B| ?", AB.det() == A.det()*B.det())
print("|A| =", A.det())
if A.det() != 0:
    print("Is A invertible ?", A.is_invertible())
    print("|A^-1| = |A|^-1 ?", 1/A.det() == A.inverse().det())
else:
    print("Is A invertible ?", A.is_invertible())
```

```
|AB| = |A| |B| ? True
|A| = 1/2
Is A invertible ? True
|A^-1| = |A|^-1 ? True
```

[\(실습하세요!\)](http://matrix.skku.ac.kr/2018-album/LA-Sec-4-1-lab.html)

정의. 수반행렬(adjugate, adjunct 또는 classical adjoint matrix)

n 차의 정사각행렬 $A = [a_{ij}]$ 의 성분 a_{ij} 에 대한 여인자를 A_{ij} 라 할 때, 행렬 $[A_{ij}]^T$ 를 A 의 수반행렬(adjugate, adjunct 또는 classical adjoint matrix)이라 하고, $\text{adj } A$ 로 나타낸다. 즉,

$$\text{adj } A = \begin{bmatrix} A_{11} & A_{21} & \cdots & A_{n1} \\ A_{12} & A_{22} & \cdots & A_{n2} \\ \vdots & \vdots & & \vdots \\ A_{1n} & A_{2n} & \cdots & A_{nn} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & \cdots & A_{2n} \\ \vdots & \vdots & & \vdots \\ A_{n1} & A_{n2} & \cdots & A_{nn} \end{bmatrix}^T = [A_{ij}]^T$$

정리. 수반행렬을 이용한 가역행렬의 역행렬

n 차의 정사각행렬 A 가 가역일 때, A 의 역행렬은 $A^{-1} = \frac{1}{|A|} \text{adj } A$ 이다.

예제 12 행렬식과 수반행렬을 이용하여 행렬 $A = \begin{bmatrix} 3 & -2 & 1 \\ 5 & 6 & 2 \\ 1 & 0 & -3 \end{bmatrix}$ 의 역행렬을 구하 여라.

```
A = matrix([[3, -2, 1], [5, 6, 2], [1, 0, -3]])
dA = A.det()    # 행렬식 구하기
adjA = A.adjugate()  # 수반행렬 구하기
print("inverse of A = (1/dA)*adjA =")
print((1/dA)*adjA)  # 수반행렬을 이용한 역행렬 구하기
```

```
inverse of A = (1/dA)*adjA =
[ 9/47   3/47   5/47]
[-17/94   5/47   1/94]
[ 3/47   1/47 -14/47]
```

<http://matrix.skku.ac.kr/2018-album/LA-Sec-4-2-lab.html> (실습하세요!)



읽을거리

튜링(Alan Turing, 1912~1954)

- 컴퓨터의 아버지, 최초의 해커 (탄생 100주년) -

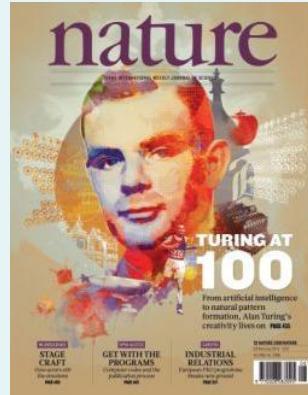
행렬의 LU-분해에 관한 첫 번째 연구는 영국의 수학자 튜링에 의해 이루어졌다. 튜링은 인공지능연구의 창시자로도 불린다. 얼마 전까지만 하더라도 세계 최초의 컴퓨터는 1946년 2월에 공개된 에니악(ENIAC)으로 알려져 있었다. 그러나 이러한 미국 중심의 컴퓨터 역사는 그동안 일급비밀로 감춰졌던 영국의 콜로서스(Colossus)가 일반에 공개됨에 따라 그보다 먼저 영국에서 세계 최초의 컴퓨터(1943년 12월)가 만들어졌다는 사실이 알려졌다.

튜링은 다른 모든 기계를 대체할 수 있고, 계산할 수 있는 모든 문제를 풀 수 있는 만능 기계를 만들 수 있다는 사실을 발견했다. 1937년 튜링은 <결정문제에의 응용을 고려한, 계산 가능한 수들에 관하여>라는 논문에서 위에 간단하게 설명된 인식을 제시했다. 이 무렵 튜링은 미국에 있었고, 1936년부터 1938년까지 프린스턴에서 지냈으며 <서수에 근거한 논리시스템>으로 박사학위를 받았다. 이때 튜링은 체스게임 같이 복잡한 계산 능력을 필요로 하는 연산을 포함하는 많은 계산을 수행할 수 있는 기계는, 숫자와 부호의 이진법으로 작동해야만 한다는 아주 친숙한 개념을 처음으로 발견하였다. 오늘날 우리는 그것을 ‘튜링 머신(Machine)’이라고 부른다. 튜링이 수학적이고 논리적인 발견을 바탕으로 착안한 것은 전자두뇌와 인공지능기계였고, 그는 그것을 현실화시켰다. 또 2차 세계대전 중에는 영국의 암호 학교에서 ‘이니그마(Enigma)’라는 독일 암호기계의 수수께끼를 풀기 위한 작업에 들어가서 결국 나치의 암호를 깨는 개가를 올린다. 튜링의 암호해독 기술로 연합군은 독일이 북아프리카에서 어떤 전략을 쓰는지 알 수 있었을 뿐 아니라 대서양 어느 지점에 독일 잠수함이 배치되어 있는지도 알 수 있었다. 역사가들은 그가 없었다면 영국이 전쟁에서 고전했으리라는 주장이 타당하다고 주장한다. 그러나 영국군과 정치가들은 종전 후 자신들의 업적을 내세우기 위해 튜링의 공로에 관해서는 함구로 일관했다.

더구나 1952년에 튜링이 동성애자라는 소문이 나며 모든 명예를 잃었다. 이런 상황을 혼신의 힘을 다해 견뎌내다 마침내 1954년 6월 7일, 침대에 누워 치명적인 용량의 시안화칼륨을 정확하게 계산해 주사된 사과를 먹고 자살을 했다. 그의 나이 겨우 사십이 되었을 때의 일이다.

‘컴퓨터의 아버지’ 앤런 튜링의 탄생 100주년

<http://dongascience.donga.com/news.php?idx=-5426053>



4. 일차독립과 기저(basis) 및 차원(Dimension)

4장 동영상, 기저, 차원 <https://youtu.be/UHqhrN38ps> (30:12)

4.1 일차독립과 부분공간

참고 동영상: http://youtu.be/HFq_-8B47xM <http://youtu.be/UTTUg6JUFQM>

실습 사이트: <http://matrix.skku.ac.kr/knowls/CLA-Week-4-Sec-3-4.html>

정의. 일차독립(linearly independent)

(1) \mathbb{R}^n 의 부분집합 $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k\}$ 에 대하여, 벡터 $\mathbf{x} \in \mathbb{R}^n$ 가

$$\mathbf{x} = c_1\mathbf{x}_1 + c_2\mathbf{x}_2 + \dots + c_k\mathbf{x}_k, \quad c_1, c_2, \dots, c_k \in \mathbb{R}$$

의 꼴로 표시되면, \mathbf{x} 를 벡터 $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k$ 의 일차결합(linear combination)이라고 한다.

(2) $S = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k\} \subseteq \mathbb{R}^n$ 에 대하여

$$c_1\mathbf{x}_1 + c_2\mathbf{x}_2 + \dots + c_k\mathbf{x}_k = \mathbf{0} \quad (c_1, c_2, \dots, c_k \in \mathbb{R}) \Rightarrow c_1 = c_2 = \dots = c_k = 0$$

이면, 벡터 $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k$ (또는 집합 S)는 일차독립(linearly independent)이라고 하고, 벡터 $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k$ (또는 집합 S)가 일차독립이 아니면 일차종속(linearly dependent)이라고 한다.

- 집합 S 가 일차종속이면 $c_1\mathbf{x}_1 + c_2\mathbf{x}_2 + \dots + c_k\mathbf{x}_k = \mathbf{0}$ 을 만족하는 모든 영이 아닌 스칼라 c_1, c_2, \dots, c_k 가 존재한다.

정리. 일차종속

집합 $S = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k\} \subseteq \mathbb{R}^n$ 에 대하여, 다음이 성립한다.

- 집합 S 가 일차종속일 필요충분조건은 S 에 속하는 한 벡터가 나머지 벡터들의 일차결합으로 표시되는 것이다.
- 집합 S 가 영벡터를 포함하면 S 는 일차종속이다.

(3) 집합 S 의 부분집합 S' 이 일차종속이면 S 도 일차종속이고, S 가 일차독립이면 S' 도 일차독립이다.

- $A = [\mathbf{x}_1 : \mathbf{x}_2 : \dots : \mathbf{x}_m]$ 는 \mathbf{x}_i 들을 열벡터(column vector)로 가지는 행렬, $\mathbf{c} = [c_1 \dots c_m]^T$ 라 하자. 그렇다면 벡터들이 일차독립임을 보이기 위해서는 동차 연립방정식 $A\mathbf{c} = \mathbf{0}$ 이 유일한 해 $\mathbf{c} = \mathbf{0}$ 을 가져야 한다. 특히 $m = n$ 일 경우 $\det A \neq 0$ 이면 유일해를 갖는다.

<http://matrix.skku.ac.kr/2018-album/LA-Sec-3-4-lab.html> (실습하세요!)

정리. 일차독립(L.I.) 판정법

벡터공간 \mathbb{R}^n 에서 n 개의 (열)벡터

$$\mathbf{x}_1 = (x_{11}, x_{21}, \dots, x_{n1}), \dots, \mathbf{x}_n = (x_{1n}, x_{2n}, \dots, x_{nn})$$

이 일차독립(L.I.)일 필요충분조건은 다음과 같다.

$$\Delta = \begin{vmatrix} x_{11} & x_{21} & \cdots & x_{n1} \\ \vdots & \ddots & & \vdots \\ x_{1n} & x_{2n} & \cdots & x_{nn} \end{vmatrix} \neq 0$$

예제 1 행렬식을 이용한 일차독립 판정법을 이용하여, \mathbb{R}^3 의 세 벡터

$$\mathbf{x}_1 = (1, 2, 3), \quad \mathbf{x}_2 = (-1, 0, 2), \quad \mathbf{x}_3 = (3, 1, 1)$$

에 대하여 $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$ 은 일차독립임을 확인하여라.

[실습 : <http://sage.skku.edu/>]

```
x1 = vector([1, 2, 3])
x2 = vector([-1, 0, 2])
x3 = vector([3, 1, 1])
```

```

A = column_matrix([x1, x2, x3])    # x1, x2, x3를 열벡터로 하는 행렬 생성
print("A =")
print(A)
print()
print("det(A) =", A.det())    # 행렬 A의 행렬식 구하기

```

```

A=
[ 1 -1  3]
[ 2  0  1]
[ 3  2  1]
det(A) = 9

```

따라서 $\det(A) \neq 0$ 이므로 $\{x_1, x_2, x_3\}$ 은 일차독립이다.

정의. 2 Step 부분공간(subspace) 판정법

- (1) 집합 $W (\neq \emptyset)$ 가 \mathbb{R}^n 의 부분집합이라 하자. 이때 다음 두 조건을 만족하면 W 는 \mathbb{R}^n 의 부분공간(subspace)이다.

$$\begin{aligned} \mathbf{x}, \mathbf{y} \in W &\Rightarrow \mathbf{x} + \mathbf{y} \in W \quad (\text{덧셈에 닫혀 있다.}) \\ \mathbf{x} \in W, k \in \mathbb{R} &\Rightarrow k\mathbf{x} \in W \quad (\text{스칼라배에 닫혀 있다.}) \end{aligned}$$

- (2) 행렬 $A = [a_{ij}]_{m \times n}$ 에 대하여 집합 $W = \{\mathbf{x} \in \mathbb{R}^n | A\mathbf{x} = \mathbf{0}\}$ 은 \mathbb{R}^n 의 부분공간이다. 이러한 W 를 $A\mathbf{x} = \mathbf{0}$ 의 해공간(solution space) 또는 A 의 영공간(null space)이라 하며 기호로 $\text{Null}(A)$ 로 나타낸다.
- (3) \mathbb{R}^n 의 부분집합 $S = \{x_1, x_2, \dots, x_k\}$ 에 대하여 S 에 있는 k 개 벡터들의 일차결합 전체의 집합, 즉 $W = \{c_1x_1 + c_2x_2 + \dots + c_kx_k | c_1, c_2, \dots, c_k \in \mathbb{R}\}$ 는 \mathbb{R}^n 의 부분공간이다. 이러한 W 를 S 에 의하여 생성된(spanned) \mathbb{R}^n 의 부분공간이라 한다. 집합 S 는 W 를 생성(span)한다고 하고, S 를 W 의 생성집합(spanning set)이라고 하며 기호로는 다음과 같이 나타낸다.

$$W = \text{span}(S) \text{ 또는 } W = \langle S \rangle$$

특히, \mathbb{R}^n 에 있는 모든 벡터가 S 에 있는 k 개 벡터들의 일차결합이면 집합 S 는 \mathbb{R}^n 을 생성한다. 즉

$$\mathbb{R}^n = \langle S \rangle = \{c_1\mathbf{x}_1 + c_2\mathbf{x}_2 + \dots + c_k\mathbf{x}_k \mid c_1, c_2, \dots, c_k \in \mathbb{R}\} = \text{span}(S) \text{ 이면,}$$

S 는 \mathbb{R}^n 을 생성한다고 한다.

예제 2 다음 동차연립방정식의 해공간(solution space) 을 구하여라.

$$\begin{aligned} 4x_1 + 12x_2 - 7x_3 + 6x_4 &= 0 \\ x_1 + 3x_2 - 2x_3 + x_4 &= 0 \\ 3x_1 + 9x_2 - 2x_3 + 11x_4 &= 0 \end{aligned}$$

```
A = matrix([[4, 12, -7, 6], [1, 3, -2, 1], [3, 9, -2, 11]])
print("Null space of A is", A.right_kernel()) # 행렬 A의 해공간 구하기
```

Null space of A is Free module of degree 4 and rank 2 over Integer Ring

Echelon basis matrix:

[1 3 4 -2]

[0 5 6 -3] # 이 두 벡터의 모든 일차결합이 다 해공간 안의 해이다.

풀이: 해공간(solution space) = $\langle (1, 3, 4, -2), (0, 5, 6, -3) \rangle = \text{span}(S)$

4.2 기저와 차원

참고 동영상: <http://youtu.be/or9c97J3Uk0> <http://youtu.be/172stJmormk>

실습 사이트: <http://matrix.skku.ac.kr/knowls/cla-week-9-sec-7-1.html>

정의. 기저(basis)

\mathbb{R}^n 의 부분집합 $S = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_s\}$ 가 아래 두 조건을 만족하면 S 를 \mathbb{R}^n 의 기저(basis)라 한다.

(1) S 가 일차독립

(2) $\text{span}(S) = \mathbb{R}^n$

정리. 일차독립, 일차종속, 각 기저에 속하는 벡터의 개수

- (1) 집합 $S = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ 이 \mathbb{R}^n 의 기저일 때, \mathbb{R}^n 의 $r (> n)$ 개의 벡터들의 집합 $T = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_r\}$ 은 항상 일차종속이다. 그러므로 T 가 일차독립이면 언제나 $r \leq n$ 이다.
- (2) 집합 $S = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ 과 $T = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_m\}$ 이 \mathbb{R}^n 의 기저이면 $n = m$ 이다.
- \mathbb{R}^n 의 기저는 무수히 많다. 그러나 각 기저에 속하는 벡터의 개수는 항상 같다.

정의. 차원(dimension)

집합 S 가 \mathbb{R}^n 의 한 기저일 때, S 에 속하는 벡터의 개수를 \mathbb{R}^n 의 차원(dimension)이라 하며 $\dim \mathbb{R}^n$ 로 나타낸다.

정리. V 의 모든 벡터 \mathbf{v} 는 기저 벡터들의 유일한 일차결합으로 표현된다.

$S = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k\}$ 을 \mathbb{R}^n 의 부분공간 V 의 기저라 하면, V 의 모든 벡터 \mathbf{v} 는 기저 벡터들의 유일한 일차결합으로 표현된다.

<http://matrix.skku.ac.kr/2018-album/LA-Sec-7-1-lab.html> (실습하세요!)

- \mathbb{R}^n 의 n 개의 일차독립인 벡터들은 임의의 벡터 $\mathbf{x} \in \mathbb{R}^n$ 를 일차결합으로 유일하게 표현하므로 기저이다!

정의. nullity(A)

$m \times n$ 행렬 A 에 대하여 $A\mathbf{x} = \mathbf{0}$ 의 해공간, 즉 A 의 영공간의 차원을 nullity(A)라고 나타낸다.

즉, $\dim \text{Null}(A) = \text{nullity}(A)$ 이다.

- Gauss-Jordan 소거법을 이용하여 행렬 $[B : \mathbf{0}]$ 을 선형연립방정식의 첨가행렬 $[A : \mathbf{0}]$ 의 RREF라 하고 행렬 B 는 첫 행부터 r ($1 \leq r \leq n$)개의 영이 아닌 행을 갖는다고 하자.

- (1) $r = n$ 이면 $A\mathbf{x} = \mathbf{0}$ 의 해는 $\mathbf{x} = \mathbf{0}$ 만을 갖는다. 따라서 해공간의 차원은 0이다.
- (2) $r < n$ 이면 (필요한 경우 열을 교환하면—변수의 위치만 변경하면 되므로)
일반성을 잃지 않고

$$[B : \mathbf{0}] = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & b_{1r+1} & \cdots & b_{1n} & : & 0 \\ 0 & 1 & 0 & \cdots & 0 & b_{2r+1} & \cdots & b_{2n} & : & 0 \\ \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & b_{rr+1} & \cdots & b_{rn} & : & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & : & 0 \\ \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & : & 0 \end{bmatrix}$$

라고 할 수 있다. 그러면, 선형연립방정식은 다음과 동치이다.

$$\begin{aligned} x_1 &= -b_{1r+1}x_{r+1} - b_{1r+2}x_{r+2} - \cdots - b_{1n}x_n \\ x_2 &= -b_{2r+1}x_{r+1} - b_{2r+2}x_{r+2} - \cdots - b_{2n}x_n \\ &\vdots \\ x_r &= -b_{rr+1}x_{r+1} - b_{rr+2}x_{r+2} - \cdots - b_{rn}x_n \end{aligned}$$

즉, $x_{r+1}, x_{r+2}, \dots, x_n$ 은 $n-r$ 개의 자유변수이다. 따라서 임의의 실수 s_1, \dots, s_{n-r} 에 대하여 $x_{r+1} = s_1, \dots, x_n = s_{n-r}$ 이라 하면, 연립방정식의 일반해는 다음과 같으므로 $n-r$ 개 벡터들의 일차결합으로 표현이 가능하다.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_r \\ x_{r+1} \\ x_{r+2} \\ x_{r+3} \\ \vdots \\ x_n \end{bmatrix} = s_1 \begin{bmatrix} -b_{1r+1} \\ -b_{2r+1} \\ \vdots \\ -b_{rr+1} \\ 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} + s_2 \begin{bmatrix} -b_{1r+2} \\ -b_{2r+2} \\ \vdots \\ -b_{rr+2} \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} + \cdots + s_{n-r} \begin{bmatrix} -b_{1n} \\ -b_{2n} \\ \vdots \\ -b_{rn} \\ 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}$$

여기서, s_1, \dots, s_{n-r} 은 임의의 실수으로

$$\mathbf{v}_1 = \begin{bmatrix} -b_{1r+1} \\ -b_{2r+1} \\ \vdots \\ -b_{rr+1} \\ 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \mathbf{v}_2 = \begin{bmatrix} -b_{1r+2} \\ -b_{2r+2} \\ \vdots \\ -b_{rr+2} \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \dots, \quad \mathbf{v}_{n-r} = \begin{bmatrix} -b_{1n} \\ -b_{2n} \\ \vdots \\ -b_{rn} \\ 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

도 연립방정식의 해이다. 따라서 위의 모든 $n-r$ 개의 벡터들의 일차결합 해는

$$\mathbf{x} = s_1\mathbf{v}_1 + s_2\mathbf{v}_2 + \dots + s_{n-r}\mathbf{v}_{n-r}$$

로 표현되므로 $\mathcal{S} = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{n-r}\}$ 는 $A\mathbf{x} = \mathbf{0}$ 의 해공간을 생성한다. 또, S 가 일차독립임을 쉽게 알 수 있다. 따라서 \mathcal{S} 는 이 해공간 $\{\mathbf{x} \in \mathbb{R}^n \mid A\mathbf{x} = \mathbf{0}\}$ 의 기저이고 이 해공간의 차원은 $n-r$ 이다.

정의. 열공간(column space) $\text{Col}(A)$, 행공간(row space), $\text{Row}(A)$

행렬 $A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$ 에 대하여 A 의 각 행으로 이루어진 m 개의 벡터

$$A_{(1)} = [a_{11} \ a_{12} \ \cdots \ a_{1n}], \ A_{(2)} = [a_{21} \ a_{22} \ \cdots \ a_{2n}], \ \dots, \ A_{(m)} = [a_{m1} \ a_{m2} \ \cdots \ a_{mn}]$$

과 A 의 각 열로 이루어진 n 개의 벡터

$$A^{(1)} = \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix}, \ A^{(2)} = \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{m2} \end{bmatrix}, \ \dots, \ A^{(n)} = \begin{bmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{bmatrix}$$

을 각각 A 의 행벡터(row vector), 열벡터(column vector)라고 한다. 이 행벡터 $A_{(1)}, \dots, A_{(m)}$ 들에 의해서 생성된 \mathbb{R}^n 의 부분공간 즉,

$$\langle A_{(1)}, \dots, A_{(m)} \rangle$$

을 A 의 행공간(row space), $\text{Row}(A)$ 로 나타내고, 열벡터 $A^{(1)}, \dots, A^{(n)}$ 에 의해 생성된 \mathbb{R}^m 의 부분공간 즉,

$$\langle A^{(1)}, \dots, A^{(n)} \rangle$$

을 A 의 열공간(column space)이라 하고, $\text{Col}(A)$ 로 나타낸다. 그리고 행공간의 차원을 A 의 행계수(row rank), 열공간의 차원을 A 의 열계수(column rank)라 하고, 각각 $r(A)$, $c(A)$ 로 나타낸다. 즉,

$$\dim \text{Row}(A) = r(A), \quad \dim \text{Col}(A) = c(A)$$

정리. 계수(rank), Rank–Nullity 정리

(1) 임의의 행렬 $A = [a_{ij}]_{m \times n}$ 에 대하여 A 의 행계수와 열계수는 같다.

- 이 계수를 행렬 A 의 계수(rank)라 하고 아래와 같이 쓴다.

$$r(A) = \text{rank}(A) = c(A)$$

(2) 임의의 행렬 $A = [a_{ij}]_{m \times n}$ 에 대하여 다음이 성립한다. [Rank–Nullity 정리]

$$\text{rank}(A) + \text{nullity}(A) = n \quad (\text{column 의 개수})$$

<http://matrix.skku.ac.kr/2018-album/LA-Sec-7-2-lab.html> (실습하세요!)

예제 3 행렬 A 의 rank와 nullity를 구하여라.

$$A = \begin{bmatrix} 1 & -2 & 1 & 1 & 2 \\ -1 & 3 & 0 & 2 & -1 \\ 0 & 1 & 1 & 3 & 4 \\ 1 & 2 & 5 & 13 & 5 \end{bmatrix}$$

[실습 : <http://sage.skku.edu/>]

```

A = matrix([[1, -2, 1, 1, 2], [-1, 3, 0, 2, -1], [0, 1, 1, 3, 4], [1, 2, 5, 13,
5]])
print("rank(A) =", A.rank())    # rank 구하기
print("nullity(A) =", A.right_nullity())    # nullity 구하기

```

rank(A) = 3

nullity(A) = 2

■ 주어진 행렬의 기본 공간들 사이의 관계

- (1) $\text{Row}(A^T) = \text{Col}(A)$, $\text{Col}(A^T) = \text{Row}(A)$,
- (2) $\text{Row}(A)^\perp = \text{Null}(A)$, $\text{Null}(A)^\perp = \text{Row}(A)$,
- (3) $\text{Col}(A)^\perp = \text{Null}(A^T)$, $\text{Null}(A^T)^\perp = \text{Col}(A)$

<http://matrix.skku.ac.kr/2018-album/LA-Sec-7-3-lab.html> (실습하세요!)

4.3 정사영과 최소제곱해(least square solution), QR 분해

참고 동영상: <http://youtu.be/Rv1rd3u-oYg> <https://youtu.be/BC9qeR0JWIs>

실습 사이트: <http://matrix.skku.ac.kr/knowl-knowls/cla-week-10-sec-7-5.html>

<http://matrix.skku.ac.kr/knowl-knowls/cla-week-10-sec-7-6.html>

QR 분해 <http://matrix.skku.ac.kr/2018-album/LS-QR-decom.html> (실습하세요!)

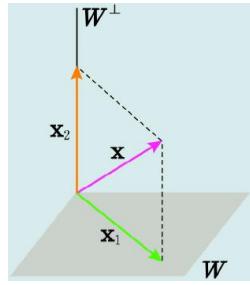
5강 동영상, 최소제곱해, QR분해 <https://youtu.be/5r2KghYFw2w> (40:49)

정리. W^\perp 는 W 에 직교(orthogonal)인 벡터들의 집합이다.

- (1) W 를 \mathbb{R}^n 의 부분공간이라 하면 \mathbb{R}^n 의 모든 벡터 \mathbf{x} 는

$$\mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2 \quad (\mathbf{x}_1 = \text{proj }_W \mathbf{x} \in W, \mathbf{x}_2 = \mathbf{x} - \mathbf{x}_1 \in W^\perp)$$

로 유일하게 표현된다. 여기서 W^\perp 는 W 에 직교(orthogonal)인 벡터들의 집합이다.



$$\mathbf{x}_1 = \text{proj}_{W} \mathbf{x} \in W, \quad \mathbf{x}_2 = \mathbf{x} - \mathbf{x}_1 = \text{proj}_{W^\perp} \mathbf{x} \in W^\perp$$

(2) W 를 \mathbb{R}^n 의 부분공간이라 하고, M 의 열벡터들이 W 의 기저(따라서 일차독립)를 이룰 때 모든 $\mathbf{x} \in \mathbb{R}^n$ 에 대하여

$$\text{proj}_W \mathbf{x} = M(M^T M)^{-1} M^T \mathbf{x}$$

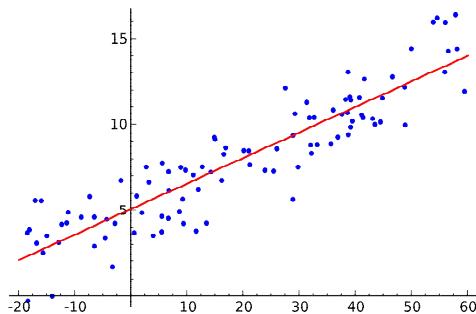
이다.

최소제곱해(least square solution)

다음과 같이 x 와 y 에 관한 2차원 데이터가 주어져 있다고 하자.

$$(x_1, y_1), \dots, (x_n, y_n)$$

이를 좌표평면에 나타내면 다음과 같다.



[그림 출처] https://en.wikipedia.org/wiki/Linear_least_squares

이제 이 데이터로부터 x 와 y 의 관계를 가장 잘 보여주는 선형모델(방정식) $y = a + bx$ 을 찾아보자. 이상적인 상황은 모든 데이터 (x_i, y_i) 에 대해서 모델 $y_i = a + bx_i$ 이 만족되는 것, 즉

$$\begin{aligned}y_1 &= a + bx_1 \\y_2 &= a + bx_2 \\\vdots \\y_n &= a + bx_n\end{aligned}$$

을 만족하는 경우이다. 그리고 이를 행렬로 나타내면 다음과 같다.

$$A = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} a \\ b \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \text{ 라 할 때, } A\mathbf{x} = \mathbf{b}$$

그러나 측정에서 생기는 오차의 영향을 줄이기 위하여, 대개 미지수의 개수보다 많은 데이터의 개수를 사용하므로, 방정식의 개수가 미지수의 개수보다 많은 (over-determined인) 선형연립방정식이 생긴다. 이런 경우, 일반적으로 선형연립방정식 $A\mathbf{x} = \mathbf{b}$ 를 만족하는 유일한 해는 물론 단 하나의 해도 없는 경우가 대부분이므로 대신 $A\mathbf{x} \approx \mathbf{b}$ 를 만족하는 근사해를 찾는다. 즉 오차 $\|A\mathbf{x} - \mathbf{b}\|$ 을 최소화하는 근사해를 찾는 이 문제를 최소제곱문제(least squares problem)라 한다.

■ 최소제곱문제(least squares problem):

Find \mathbf{x} such that $\min \|A\mathbf{x} - \mathbf{b}\|$

다시 위의 데이터로부터 선형모델을 찾는 문제를 생각해보자. \hat{y}_i 를 모델 $y = a + bx$ 에 x_i 를 대입하여 얻은 값이라고 하면, ($\hat{y}_i = a + bx_i$) 최소제곱문제는 결국 오차 $E = \sum_{i=1}^n (y_i - \hat{y}_i)^2$ 이 최소가 되는 a, b 를 구하는 것과 같다.

■ Normal equation (정규방정식)

최소제곱문제는 결국 \mathbf{b} 와 $A\mathbf{x}$ 사이의 거리(distance)를 가장 짧게 만드는 $\hat{\mathbf{x}}$ 를 찾는 것이다.

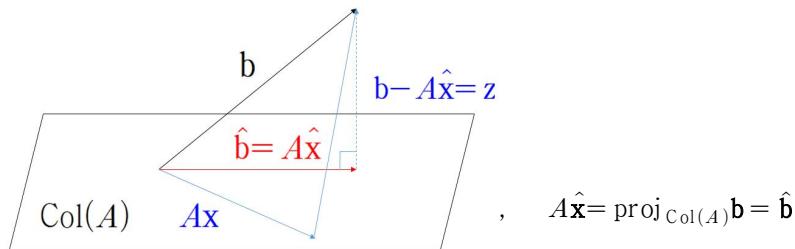
정의.

Normal equation(정규방정식)

$\hat{\mathbf{x}}$ 가 최소제곱문제 $\min \|A\mathbf{x} - \mathbf{b}\|$ 의 해가 될 필요충분조건은 $A^T A \hat{\mathbf{x}} = A^T \mathbf{b}$ 을 만족하는 것이다.

[이때 선형연립방정식 $A^T A \mathbf{x} = A^T \mathbf{b}$ 을 normal equation(정규방정식)이라 한다.]

(Sketch) 일단 선형연립방정식 $A\mathbf{x} = \mathbf{b}$ 의 해가 존재하지 않으므로 $\mathbf{b} \notin \text{Col}(A)$ 이 성립한다. \mathbf{b} 의 $\text{Col}(A)$ 위로의 정사영을 $\hat{\mathbf{b}}$ 이라 하면, $A\mathbf{x} = \hat{\mathbf{b}}$ 의 해가 존재한다. 이 해를 $\hat{\mathbf{x}}$ 이라 하면, $\hat{\mathbf{x}}$ 이 바로 최소제곱문제 $\min \|A\mathbf{x} - \mathbf{b}\|$ 의 최소제곱해(least squares solution)가 된다. 다음 그림에서



$\mathbf{z} = \mathbf{b} - A\hat{\mathbf{x}}$ 라 하면 $\mathbf{z} \perp \text{Col}(A)$ 이므로 \mathbf{z} 와 A 의 열벡터들이 각각 직교이므로 $A^T \mathbf{z} = 0$ 이 성립한다. 따라서 $A^T(\mathbf{b} - A\hat{\mathbf{x}}) = \mathbf{0}$, 즉 $A^T A \hat{\mathbf{x}} = A^T \mathbf{b}$ 이 성립한다.

A 의 열벡터들이 일차독립(이를 A 가 full column rank를 갖는다고 한다)이면, $A^T A$ 가 가역이고, A 의 열벡터들이 $\text{Col}(A)$ 의 기저가 된다. 따라서 최소제곱문제의 최소제곱해(least squares solution)는

$$\hat{\mathbf{x}} = (A^T A)^{-1} A^T \mathbf{b}$$

이 고, $A\hat{\mathbf{x}} = \text{proj}_{\text{Col}(A)} \mathbf{b} = [A(A^T A)^{-1} A^T] \mathbf{b} = \hat{\mathbf{b}}$ 를 만족한다.

예제 4 다음 네 점 $(0, 1), (1, 3), (2, 4), (3, 4)$ 를 지나는 least square line $y = a + bx$ 을 구하여라.

풀이 위의 네 점은 $y_1 = a + bx_1, y_2 = a + bx_2, y_3 = a + bx_3, y_4 = a + bx_4$ 의 해이다.

위의 연립방정식을 행렬을 이용하여 쓰면 $\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \\ 1 & x_4 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$ 가 된다. 따라서

$$A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ 3 \\ 4 \\ 4 \end{bmatrix} \text{ 이고, } A^T A \text{ 와 } (A^T A)^{-1} \text{ 를 계산하면,}$$

$$A^T A = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{bmatrix} = \begin{bmatrix} 4 & 6 \\ 6 & 14 \end{bmatrix}, \quad (A^T A)^{-1} = \begin{bmatrix} \frac{7}{10} & -\frac{3}{10} \\ -\frac{3}{10} & \frac{2}{10} \end{bmatrix} \text{ 이다.}$$

정규방정식을 이용하여 $\hat{\mathbf{x}}$ 를 구하면 다음과 같다.

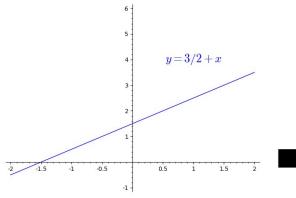
$$\hat{\mathbf{x}} = \begin{bmatrix} a \\ b \end{bmatrix} \equiv (A^T A)^{-1} A^T \mathbf{b} = \begin{bmatrix} \frac{7}{10} & -\frac{3}{10} \\ -\frac{3}{10} & \frac{2}{10} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 4 \\ 4 \end{bmatrix} = \begin{bmatrix} \frac{3}{2} \\ 1 \end{bmatrix} = \begin{bmatrix} 1.5 \\ 1 \end{bmatrix}$$

따라서 위의 점들에 가장 근접하는 최소제곱직선(least square line)은 $y = \frac{3}{2} + x$ 이다.

```
A = matrix([[1, 0], [1, 1], [1, 2], [1, 3]])
b = vector([1, 3, 4, 4])
print((A.transpose()*A).inverse()*A.transpose()*b)
```

(3/2, 1) # <http://matrix.skku.ac.kr/K-MOOC-LA/cla-week-10.html> (실습하세요!)

따라서 최소제곱해를 구해 찾은 least square line은 $y = \frac{3}{2} + x$ 이다.



- 💡 Normal equation(정규방정식)은 오차에 매우 민감하므로, 최소제곱문제 $\min \|Ax - b\|$ 를 풀 때는 특히 주의하여야 한다. 그래서 대부분의 경우는 A 의 QR 분해(QR-decompositon)를 이용하여 최소제곱문제를 해결한다. QR 분해는 다음 페이지에서 자세히 다룬다.

정의. 정규직교기저(orthonormal basis)

- (1) \mathbb{R}^n 의 벡터 $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k$ 에 대하여

$$S = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k\}$$

라 하자. 이때, S 의 서로 다른 임의의 두 벡터가 모두 직교하면 S 를 직교집합(orthogonal set)이라 한다. 특히, 직교집합 S 에 속하는 벡터가 모두 크기가 1인 경우 S 를 정규직교집합(orthonormal set)이라고 한다.

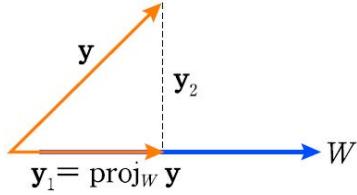
- (2) \mathbb{R}^n 의 기저 S 가 직교집합이면 직교기저(orthogonal basis), 정규직교집합이면 정규직교기저(orthonormal basis)라고 한다.

정리. Gram–Schmidt 정규직교화 기저의 존재성

집합 $S = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ 을 \mathbb{R}^n 의 임의의 기저라 하자. 그러면 S 로부터 얻어지는 정규직교기저가 존재한다.

[Gram–Schmidt 정규직교화 과정]

먼저 \mathbb{R}^n 의 기저 S 로부터 직교집합 $T = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n\}$ 을 다음과 같은 단계로 계산한다.



[단계 1] $y_1 = x_1$ 이라 한다.

[단계 2] y_1 에 의하여 생성되는 부분공간을 W_1 이라 하고

$$y_2 = x_2 - \text{proj}_{W_1} x_2 = x_2 - \frac{x_2 \cdot y_1}{\|y_1\|^2} y_1$$

로 한다.

[단계 3] y_1, y_2 에 의하여 생성되는 부분공간을 W_2 라 하고

$$y_3 = x_3 - \text{proj}_{W_2} x_3 = x_3 - \frac{x_3 \cdot y_1}{\|y_1\|^2} y_1 - \frac{x_3 \cdot y_2}{\|y_2\|^2} y_2 \text{로 한다.}$$

[단계 4] 에서부터 n 까지는 마찬가지 방법으로

$$y_k = x_k - \text{proj}_{W_{k-1}} x_k = x_k - \frac{x_k \cdot y_1}{\|y_1\|^2} y_1 - \frac{x_k \cdot y_2}{\|y_2\|^2} y_2 - \cdots - \frac{x_k \cdot y_{k-1}}{\|y_{k-1}\|^2} y_{k-1}$$

$(k = 4, 5, \dots, n)$

위의 단계로부터 얻어지는 $T = \{y_1, y_2, \dots, y_n\}$ 은 서로 직교인 직교집합이고, 각각의 크기를 1로 하면, 즉 $z_k = \frac{y_k}{\|y_k\|}$ ($k = 1, 2, \dots, n$)라 정의하면 집합 $\{z_1, z_2, \dots, z_n\}$ 은 \mathbb{R}^n 의 정규직교기저이다. ■

예제 5 $x_1 = (1, 1, 0), x_2 = (0, 1, 2), x_3 = (1, 2, 1)$ 일 때, Gram–Schmidt 정규직교화 과정을 이용하여 \mathbb{R}^3 의 기저 $S = \{x_1, x_2, x_3\}$ 로부터 \mathbb{R}^3 의 정규직교기저 $Z = \{z_1, z_2, z_3\}$ 를 구하여라.

풀이 먼저 직교화 과정을 이용하여 y_1, y_2, y_3 를 계산하면 (직교기저를 얻는다).

단계 1 : $y_1 = x_1 = (1, 1, 0)$

$$\text{단계 2} : \mathbf{y}_2 = \mathbf{x}_2 - \text{proj}_{W_1} \mathbf{x}_2 = \mathbf{x}_2 - \frac{\mathbf{x}_2 \cdot \mathbf{y}_1}{\|\mathbf{y}_1\|^2} \mathbf{y}_1$$

$$= (0, 1, 2) - \frac{1}{2}(1, 1, 0) = \left(-\frac{1}{2}, \frac{1}{2}, 2\right)$$

$$\text{단계 3} : \mathbf{y}_3 = \mathbf{x}_3 - \text{proj}_{W_2} \mathbf{x}_3$$

$$= \mathbf{x}_3 - \frac{\mathbf{x}_3 \cdot \mathbf{y}_1}{\|\mathbf{y}_1\|^2} \mathbf{y}_1 - \frac{\mathbf{x}_3 \cdot \mathbf{y}_2}{\|\mathbf{y}_2\|^2} \mathbf{y}_2$$

$$= (1, 2, 1) - \frac{3}{2}(1, 1, 0) - \frac{5}{9}\left(-\frac{1}{2}, \frac{1}{2}, 2\right) = \left(-\frac{2}{9}, \frac{2}{9}, -\frac{1}{9}\right)$$

그러므로 $\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3$ 를 각각 정규화하여, \mathbb{R}^3 의 정규직교기저 $Z = \{\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3\}$ 를 얻는다.

$$\mathbf{z}_1 = \frac{\mathbf{y}_1}{\|\mathbf{y}_1\|} = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0\right), \quad \mathbf{z}_2 = \frac{\mathbf{y}_2}{\|\mathbf{y}_2\|} = \left(-\frac{\sqrt{2}}{6}, \frac{\sqrt{2}}{6}, \frac{2\sqrt{2}}{3}\right),$$

$$\mathbf{z}_3 = \frac{\mathbf{y}_3}{\|\mathbf{y}_3\|} = \left(-\frac{2}{3}, \frac{2}{3}, -\frac{1}{3}\right).$$

집합 $Z = \left\{ \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0\right), \left(-\frac{\sqrt{2}}{6}, \frac{\sqrt{2}}{6}, \frac{2\sqrt{2}}{3}\right), \left(-\frac{2}{3}, \frac{2}{3}, -\frac{1}{3}\right) \right\}$ 는 \mathbb{R}^3 의 정규직교기저이다.

즉, $\{(1/2*\sqrt{2}), 1/2*\sqrt{2}, 0), (-1/3*\sqrt{1/2}), 1/3*\sqrt{1/2}, 4/3*\sqrt{1/2}), (-2/3, 2/3, -1/3)\}$ ■

[실습 : <http://sage.skku.edu/>]

```
x1 = vector([1, 1, 0])
x2 = vector([0, 1, 2])
x3 = vector([1, 2, 1])
A = matrix([x1, x2, x3]) # x1, x2, x3를 행벡터로 하는 행렬 생성

print("A =")
print(A)
print()
```

```
[G, mu] = A.gram_schmidt() # 직교화 과정: 행에 대하여 직교기저를 찾는다.
A==mu*G
print("G =")
print(G)
print()
N = matrix([G.row(i) / G.row(i).norm() for i in range(0, 3)])
# 정규화과정: 정규화한 직교기저를 행으로 하는 행렬
print("N =")
print(N)
```

A=

$$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

G=

$$\begin{bmatrix} 1 & 1 & 0 \\ -1/2 & 1/2 & 2 \\ -2/9 & 2/9 & -1/9 \end{bmatrix}$$

N=

$$\begin{bmatrix} 1/2*\sqrt{2} & 1/2*\sqrt{2} & 0 \\ -1/3*\sqrt{1/2} & 1/3*\sqrt{1/2} & 4/3*\sqrt{1/2} \\ -2/3 & 2/3 & -1/3 \end{bmatrix}$$
■

<http://matrix.skku.ac.kr/2018-album/LA-Sec-7-7-lab.html> (실습하세요!)

■ QR 분해 <http://matrix.skku.ac.kr/2018-album/LS-QR-decom.html> (실습!)

$m \times k$ 행렬 A 가 k 개의 일차독립인 열들을 가지면 여기에 Gram–Schmidt의 정규직교화과정을 적용하여 얻은 정규직교벡터들을 열로 하는 행렬 Q 를 만들어 행렬 $A = QR$ (여기서 R 은 상삼각행렬)로 분해가 된다. QR 분해는 최소제곱문제와 고윳값, 고유벡터를 구하는 문제를 해결하는데 많이 사용된다.

정리. QR 분해

행렬 A 가 $\text{rank } k$ 인 $m \times k$ 행렬이라면, $A = QR$ 로 분해 가능하다. 여기서 Q 는 A 의 열공간 $\text{Col}(A)$ 의 정규직교기저로 만들어진 $m \times k$ 행렬이고, R 은 가역인 크기 $k \times k$ 의 상삼각행렬이다.

이제 행렬 $A \in M_{m \times k}$ 이고 $\text{rank}(A) = k$ 인 경우 $A = [\mathbf{w}_1 : \mathbf{w}_2 : \dots : \mathbf{w}_k]_{m \times k}$ 라 하자. Gram–Schmidt 정규직교화 과정에 의하여 행렬 A 의 열공간 $\text{Col}(A)$ 의 정규직교기저 $\{\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_k\}$ 을 얻을 수 있다. 이제 A 와 $Q = [\mathbf{q}_1 : \mathbf{q}_2 : \dots : \mathbf{q}_k]_{m \times k}$ 사이의 관계를 알아보자.

$\{\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_k\}$ 가 $\text{Col}(A)$ 의 정규직교기저이므로

$$\begin{aligned}\mathbf{w}_1 &= (\mathbf{w}_1 \cdot \mathbf{q}_1)\mathbf{q}_1 + (\mathbf{w}_1 \cdot \mathbf{q}_2)\mathbf{q}_2 + \dots + (\mathbf{w}_1 \cdot \mathbf{q}_k)\mathbf{q}_k \\ \mathbf{w}_2 &= (\mathbf{w}_2 \cdot \mathbf{q}_1)\mathbf{q}_1 + (\mathbf{w}_2 \cdot \mathbf{q}_2)\mathbf{q}_2 + \dots + (\mathbf{w}_2 \cdot \mathbf{q}_k)\mathbf{q}_k \\ &\vdots \\ \mathbf{w}_k &= (\mathbf{w}_k \cdot \mathbf{q}_1)\mathbf{q}_1 + (\mathbf{w}_k \cdot \mathbf{q}_2)\mathbf{q}_2 + \dots + (\mathbf{w}_k \cdot \mathbf{q}_k)\mathbf{q}_k\end{aligned}$$

로 표현 가능하다. 또한 $\mathbf{w}_i \cdot \mathbf{q}_j = 0$ ($i < j$)이므로 위의 표현은 다음과 같이 간단하게 표현 가능하다. 즉,

$$\begin{aligned}\mathbf{w}_1 &= (\mathbf{w}_1 \cdot \mathbf{q}_1)\mathbf{q}_1 \\ \mathbf{w}_2 &= (\mathbf{w}_2 \cdot \mathbf{q}_1)\mathbf{q}_1 + (\mathbf{w}_2 \cdot \mathbf{q}_2)\mathbf{q}_2 \\ &\vdots \\ \mathbf{w}_k &= (\mathbf{w}_k \cdot \mathbf{q}_1)\mathbf{q}_1 + (\mathbf{w}_k \cdot \mathbf{q}_2)\mathbf{q}_2 + \dots + (\mathbf{w}_k \cdot \mathbf{q}_k)\mathbf{q}_k .\end{aligned}$$

이제 다음과 같이 상삼각행렬(upper triangular matrix) R 을 정의하고,

$$R = \begin{bmatrix} \mathbf{w}_1 \cdot \mathbf{q}_1 & \mathbf{w}_1 \cdot \mathbf{q}_2 & \mathbf{w}_1 \cdot \mathbf{q}_3 & \dots & \mathbf{w}_1 \cdot \mathbf{q}_k \\ 0 & \mathbf{w}_2 \cdot \mathbf{q}_2 & \mathbf{w}_2 \cdot \mathbf{q}_3 & \dots & \mathbf{w}_2 \cdot \mathbf{q}_k \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & \mathbf{w}_k \cdot \mathbf{q}_k & \mathbf{q}_k \end{bmatrix}$$

앞서 정의한 행렬 Q 와의 곱을 생각해보자. 행렬 곱 QR 의 j 번째 열벡터는 $QR^{(j)}$ 이므로 R 의 j 번째 열벡터의 성분을 계수로 하는 Q 의 열벡터들의 일차결합이다. 따라서

$$[\mathbf{w}_1 : \mathbf{w}_2 : \cdots : \mathbf{w}_k] = [\mathbf{q}_1 : \mathbf{q}_2 : \cdots : \mathbf{q}_k] \begin{bmatrix} \mathbf{w}_1 \cdot \mathbf{q}_1 & \mathbf{w}_2 \cdot \mathbf{q}_1 & \cdots & \mathbf{w}_k \cdot \mathbf{q}_1 \\ 0 & \mathbf{w}_2 \cdot \mathbf{q}_2 & \cdots & \mathbf{w}_k \cdot \mathbf{q}_2 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{w}_k \cdot \mathbf{q}_k \end{bmatrix} \quad (1)$$

즉, $A = QR$ 이다. 위에서 만들어진 k 차 정사각 삼각행렬 R 은 주대각성분이 모두 영이 아니므로, 가역행렬이다. 또한 $Q^T Q = I_k$ 이므로 Q 는 (column) 직교행렬이다. 따라서 우리는 QR 분해(QR decomposition) $\textcolor{red}{A = QR}$ 를 얻은 것이다.

예제 6 행렬 A 가 다음과 같이 주어졌을 때 QR -분해를 구하여라.

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

풀이. 행렬 A 가 full column rank를 가지고 있으므로 QR 분해가 존재함을 알 수 있다. Gram-Schmidt 정규직교화를 다음 벡터들에 적용해보자.

$$\mathbf{w}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \quad \mathbf{w}_2 = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}, \quad \mathbf{w}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

그러면 정규직교화 과정을 통해 직교행렬 Q 를 다음과 같이 구할 수 있다.

$$Q = \begin{bmatrix} \frac{1}{\sqrt{3}} & -\frac{2}{\sqrt{6}} & 0 \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{6}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{2}} \end{bmatrix} \quad (\text{직교벡터!! 열벡터의 내적들이 모두 zero})$$

그리면 앞의 (1)식으로부터 (삼각행렬) R 을 구하면,

$$R = Q^T A = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ -\frac{2}{\sqrt{6}} & \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{6}} \\ 0 & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} \sqrt{3} & \frac{2}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ 0 & \frac{2}{\sqrt{6}} & \frac{1}{\sqrt{6}} \\ 0 & 0 & \frac{1}{\sqrt{2}} \end{bmatrix} \quad (\text{삼각행렬})$$

따라서 다음과 같은 QR 분해를 얻을 수 있다.

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{3}} & -\frac{2}{\sqrt{6}} & 0 \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{6}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \sqrt{3} & \frac{2}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ 0 & \frac{2}{\sqrt{6}} & \frac{1}{\sqrt{6}} \\ 0 & 0 & \frac{1}{\sqrt{2}} \end{bmatrix} \quad (\text{QR 분해 } A = QR)$$

$A \qquad \qquad \qquad Q \qquad \qquad \qquad R$

👉 아래는 Sage가 제공하는 `gram_schmidt()` 함수를 수정하여 좀 더 직관적으로 그 과정을 확인하도록, `gs_orth()` 함수 코드를 만들어 QR 분해를 확인하였다. (자주 사용하는 함수는 함수 코드를 만들어 활용한다.)

```
# SageMath의 gram_schmidt() 함수를 보완하여 gs_orth()을 만들었음
def gs_orth(A):          # 함수 정의

    m, n = A.nrows(), A.ncols()    # 행렬의 크기
    r = A.rank()      # 행렬의 rank

    if m < n:    # 행렬의 크기 확인
        raise ValueError("The number of rows must be larger than the number of columns.")

    elif r < n:  # full column rank인지 확인
        raise ValueError("The matrix is not full column rank.")

    [G, mu] = A.transpose().gram_schmidt()    # gram_schmidt 함수 활용
    Q1 = matrix([G.row(i) / G.row(i).norm() for i in range(0, n)]) # Q의 transpose
    R1 = Q1*A
```

```

Q = simplify(Q1.transpose()) # 정규직교기저로 만들어진 행렬
R = simplify(R1)           # 상삼각행렬

return Q, R

A = matrix([[1, 0, 0], [1, 1, 0], [1, 1, 1]]) # 행렬 입력
Q, R = gs_orth(A) # QR 분해
print("Q =")
print(Q)
print()
print("R =")
print(R)
print()
print("Q*R =")
print(Q*R)

```

```

Q =
[      1/3*sqrt(3) -1/3*sqrt(2)*sqrt(3)          0]
[      1/3*sqrt(3)  1/6*sqrt(2)*sqrt(3)       -1/2*sqrt(2)]
[      1/3*sqrt(3)  1/6*sqrt(2)*sqrt(3)        1/2*sqrt(2)]

R =
[      sqrt(3)      2/3*sqrt(3)      1/3*sqrt(3)]
[          0      1/3*sqrt(2)*sqrt(3)  1/6*sqrt(2)*sqrt(3)]
[          0          0      1/2*sqrt(2)]

Q*R = A =
[1  0  0]
[1  1  0]
[1  1  1]      # (QR 분해  $A = QR$  임을 확인)

```

■ 최소제곱문제를 QR 분해로 풀기

앞서 학습했듯이, $A\mathbf{x} = \mathbf{b}$ 의 최소제곱해란 normal equation인 $A^T A \mathbf{x} = A^T \mathbf{b}$ 의 해이다. 여기서 $A^T A$ 가 가역행렬이라면 $\hat{\mathbf{x}} = (A^T A)^{-1} A^T \mathbf{b}$ 가 그 해가 된다. 그렇다면 $A^T A$ 의 역행렬을 구하는 것이 최소제곱해 문제를 해결하는 가장 중요한 열쇠가 된다. 이를 QR 분해를 이용하여 구해보자.

A 가 full column rank를 가지면, 다음과 같이 QR 분해된다.

$$A = QR$$

그러면

$$\begin{aligned} A^T A \mathbf{x} = A^T \mathbf{b} &\Leftrightarrow (R^T Q^T)(QR)\mathbf{x} = R^T Q^T \mathbf{b} \\ &\Leftrightarrow R^T R \mathbf{x} = R^T Q^T \mathbf{b} \quad (\text{양변에 } (R^T)^{-1} \text{를 취해주자.}) \\ &\Leftrightarrow R \mathbf{x} = Q^T \mathbf{b} \end{aligned}$$

이제 (R 의 가역 행렬이므로) 여기에 후진대입법(backward substitution)을 적용하면,

$$\therefore \mathbf{x} = R^{-1} Q^T \mathbf{b} \quad (\text{유일해})$$

가 된다. 이와 같이 QR 분해는 선형연립방정식의 (최소제곱)해를 역행렬을 구하지 않고 손쉽게 구할 수 있도록 해주므로 다양한 계산이 필요한 컴퓨터 알고리즘의 연구에서 가장 중요한 도구로 여겨진다. QR 분해의 이론에 대해 좀 더 자세한 내용은 아래 주소를 참고하라.

<http://matrix.skku.ac.kr/nla/ch7.htm>

<http://matrix.skku.ac.kr/sglee/03-Note/QR-Decomp.htm>

<http://matrix.skku.ac.kr/sglee/project/generalized-inverse/>

예제 7 다음 문제의 최소제곱해를 구하여라.

$$\begin{cases} x_1 + 3x_2 + 5x_3 = 2 \\ x_1 + x_2 = 3 \\ x_1 + x_2 + 2x_3 = -1 \\ x_1 + 3x_2 + 3x_3 = 2 \end{cases}$$

풀이 계수행렬 A 는 다음과 같이 $A = QR$ 분해된다.

$$A = \begin{bmatrix} 1 & 3 & 5 \\ 1 & 1 & 0 \\ 1 & 1 & 2 \\ 1 & 3 & 3 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \end{bmatrix} \begin{bmatrix} 2 & 4 & 5 \\ 0 & 2 & 3 \\ 0 & 0 & 2 \end{bmatrix}$$

이를 이용하면, 최소제곱해 $\hat{\mathbf{x}} = R^{-1}Q^T \mathbf{b}$ 를 얻는다.

$$\hat{\mathbf{x}} = R^{-1}Q^T \mathbf{b} = \begin{bmatrix} \frac{1}{2} & -1 & \frac{1}{4} \\ 0 & \frac{1}{2} & -\frac{3}{4} \\ 0 & 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \end{bmatrix} \begin{bmatrix} 2 \\ 3 \\ -1 \\ 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ -1 \end{bmatrix} \text{ 최소제곱해 } \blacksquare$$

```
# Sage의 gram_schmidt 함수를 보완하여 작성
def gs_orth(A):          # 함수 정의

    m, n = A.nrows(), A.ncols()    # 행렬의 크기
    r = A.rank()      # 행렬의 rank

    if m < n:    # 행렬의 크기 확인
        raise ValueError("The number of rows must be larger than the number of
columns.")
    elif r < n:  # full column rank인지 확인
        raise ValueError("The matrix is not full column rank.")

    [G, mu] = A.transpose().gram_schmidt()    # gram_schmidt 함수 활용
    Q1 = matrix([G.row(i) / G.row(i).norm() for i in range(0, n)]) # Q의 transpose
    R1 = Q1*A
    Q = simplify(Q1.transpose())  # 정규직교기저로 만들어진 행렬
    R = simplify(R1)            # 상삼각행렬

    return Q, R

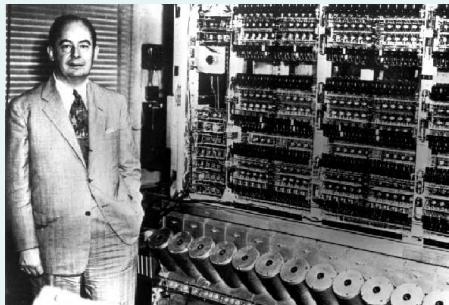
A = matrix([[1, 3, 5], [1, 1, 0], [1, 1, 2], [1, 3, 3]]) # 계수 행렬 입력
b = vector([2, 3, -1, 2]) # 상수항 벡터 입력
```

```
Q, R = gs_orth(A)      # QR 분해  
print(R.solve_right(Q.transpose()*b)) # 최소제곱해
```

(0, 2, -1) # 최소제곱해 ■

읽을거리

2차 세계대전 이후 폰 노이만(John von Neumann)과 골드스타인(Herman Goldstine)이 현대적인 디지털 컴퓨터의 발전을 이끌며 행렬에 대한 활발한 연구가 다시 꽂피우게 된다. 1947년 round-off errors를 분석하는 과정에서 condition number가 소개되었다. 20세기 stored-program 컴퓨터 개발을 이끈 사람은 튜링(Alan Turing)과 폰 노이만이었다.



폰 노이만

튜링은 1948년 LU-분해를 소개하였으며, QR-분해의 장점은 그 후 10년이 지난 후 폰 노이만이 컴퓨터를 이용해 최소제곱해를 구하면서 크게 인식되기 시작하였다. 특히 QR-분해는 연립방정식의 최적해를 구하거나 고윳값을 구하는 등 다양한 컴퓨터 알고리즘에 이용되면서 그 중요성이 LU-분해보다 훨씬 더 커졌다. 2차 세계대전 이후 폰 노이만(John von Neumann)과 골드스타인(Herman Goldstine)이 현대적인 디지털 컴퓨터의 발전을 이끌며 행렬에 대한 활발한 연구가 다시 꽂피우게 된다. 1947년 round-off errors를 분석하는 과정에서 condition number가 소개되었다. 20세기 stored-program 컴퓨터 개발을 이끈 사람은 튜링(Alan Turing)과 폰 노이만이었다. 튜링은 1948년 LU-분해를 소개하였으며, QR-분해의 장점은 그 후 10년이 지난 후 폰 노이만이 컴퓨터를 이용해 최소제곱해를 구하면서 크게 인식되기 시작하였다. 특히 QR-분해는 연립방정식의 최적해를 구하거나 고윳값을 구하는 등 다양한 컴퓨터 알고리즘에 이용되면서 그 중요성이 LU-분해보다 훨씬 더 커졌다.

5. 선형변환 (Linear Transformations)

6장 동영상, 선형변환 https://youtu.be/_t871V2CDSw (23:28)

5.1 선형변환

참고 동영상: <http://youtu.be/YF6-ENHfI6E> <http://youtu.be/Yr23NRSpSoM>

실습 사이트: <http://matrix.skku.ac.kr/knou-knowls/cla-week-8-Sec-6-1.html>

정의. 행렬변환(matrix transformation), 선형변환(linear transformation)

(1) 입력과 출력이 모두 벡터인 함수를 변환(Transformation)이라 한다. 그리고 \mathbb{R}^n 에서 \mathbb{R}^m 으로의 변환 $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ 에서 $w = T(x)$ 를 벡터 x 의 T 에 대한 이미지(image), x 를 벡터 w 의 원상(pre-image)이라 한다.



변환의 특수한 경우로, A 가 $m \times n$ 행렬이고, $T_A(\mathbf{x}) = A\mathbf{x}$, $\mathbf{x} \in \mathbb{R}^n$ 인 $T_A : \mathbb{R}^n \rightarrow \mathbb{R}^m$ 을 행렬변환(matrix transformation)이라 한다.

(2) \mathbb{R}^n 에서 \mathbb{R}^m 으로의 변환 $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ 가 임의의 벡터 $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ 와 임의의 스칼라 k 에 대하여 다음 두 조건을 만족하면 T 를 \mathbb{R}^n 에서 \mathbb{R}^m 로의 선형변환(linear transformation)이라고 한다.

$$(1) \quad T(\mathbf{u} + \mathbf{v}) = T(\mathbf{u}) + T(\mathbf{v})$$

$$(2) \quad T(k\mathbf{u}) = kT(\mathbf{u}) \quad (k \in \mathbb{R})$$

특히, \mathbb{R}^n 에서 \mathbb{R}^n 자신으로의 선형변환 $T : \mathbb{R}^n \rightarrow \mathbb{R}^n$ 를 \mathbb{R}^n 위의 선형연산자 (linear operator)라고 한다.

☞ \mathbb{R}^n 에서 \mathbb{R}^m 으로의 모든 선형변환 $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ 은 행렬변환 $T(\mathbf{x}) = A\mathbf{x}$ 으로 나타낼 수 있다.

- $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ 을 임의의 선형변환이라 할 때, \mathbb{R}^n 의 기본단위벡터 (표준기저) $\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_n$ 에 대하여 모든 $\mathbf{x} \in \mathbb{R}^n$ 는

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = x_1\mathbf{e}_1 + x_2\mathbf{e}_2 + \cdots + x_n\mathbf{e}_n$$

와 같이 나타낼 수 있고, $T(\mathbf{e}_1), T(\mathbf{e}_2), \dots, T(\mathbf{e}_n)$ 은 각각 $m \times 1$ 행렬이므로

$$T(\mathbf{e}_1) = \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix}, \quad T(\mathbf{e}_2) = \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{m2} \end{bmatrix}, \quad \dots, \quad T(\mathbf{e}_n) = \begin{bmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{bmatrix}$$

이라 할 수 있다. 따라서 모든 선형변환 $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ 은

$$T(\mathbf{x}) = x_1 T(\mathbf{e}_1) + x_2 T(\mathbf{e}_2) + \cdots + x_n T(\mathbf{e}_n)$$

$$= x_1 \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix} + x_2 \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{m2} \end{bmatrix} + \cdots + x_n \begin{bmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{bmatrix} = \begin{bmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \end{bmatrix} \quad (1)$$

의 형태로 표시할 수 있다. 여기서 $T(\mathbf{e}_1), T(\mathbf{e}_2), \dots, T(\mathbf{e}_n)$ 을 열벡터로 갖는 $m \times n$ 행렬을 A 라 하면

$$A = [T(\mathbf{e}_1) : T(\mathbf{e}_2) : \cdots : T(\mathbf{e}_n)] = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

이므로

$$T(\mathbf{x}) = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = A\mathbf{x}$$

이다. 위의 행렬 $A = [a_{ij}]_{m \times n}$ 를 선형변환 T 의 표준행렬(standard matrix)이라 하며 $[T]$ 라 표시한다. 따라서 (1)로 주어진 선형변환의 표준행렬은 기본단위벡터를 순서대로 대입하여 열을 구하여 쉽게 만든다.

정리. 표준행렬

$T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ 이면 T 의 표준행렬 $A = [T]$ 와 $\mathbf{x} \in \mathbb{R}^n$ 에 대하여 다음이 성립 한다.

$$T(\mathbf{x}) = A\mathbf{x}, \quad \forall \mathbf{x} \in \mathbb{R}^n$$

여기서 $A = [T(\mathbf{e}_1) : T(\mathbf{e}_2) : \dots : T(\mathbf{e}_n)]$ 이다.

예제 1. 선형변환 $T : \mathbb{R}^3 \rightarrow \mathbb{R}^4$ 가 $T \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} x_1 + 2x_2 \\ -x_1 - x_2 \\ x_3 \\ x_1 + x_3 \end{pmatrix}$ 일 때, T 의 표준행렬을 구하

여라. [실습: <http://sage.skku.edu/>]

```
var('x, y, z')
h(x, y, z) = [x + 2*y, -x - y, z, x + z]    # 함수의 정의
T = linear_transformation(QQ^3, QQ^4, h)      # 선형변환 정의, 여기서 스칼라는
                                                유리수
e1 = vector([1, 0, 0])    # R^3의 단위벡터 생성
e2 = vector([0, 1, 0])
e3 = vector([0, 0, 1])
column_matrix([T(e1), T(e2), T(e3)])    # T의 표준행렬
```

```
[ 1  2  0]
[-1 -1  0]
[ 0  0  1]
[ 1  0  1]
```

예제 2 선형변환 $T : \mathbb{R}^3 \rightarrow \mathbb{R}^4$ 가 $T\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} x_1 + 2x_2 \\ -x_1 - x_2 \\ x_3 \\ x_1 + x_3 \end{pmatrix}$ 일 때, T 의 표준행렬을 이용하여 구하여라.

하여

$T(\mathbf{x}) = A\mathbf{x}$ 으로 표시하여라. 그리고 $\mathbf{x}_0 = (2, -3, 3)$ 일 때, A 를 이용하여 $T(\mathbf{x}_0)$ 를 구하여라.

```
var('x, y, z')
h(x, y, z) = [x + 2*y, -x - y, z, x + z]    # 함수의 정의
T = linear_transformation(QQ^3, QQ^4, h)      # 선형변환 정의, 여기서 스칼라는
                                                유리수
A = T.matrix(side = 'right')    # 선형변환의 표준행렬을 구하는 Sage 명령어
x0 = random_vector(3)    # 임의의 벡터
print("A =")
print(A)
print()
print("T(x0) =", T(x0))    # 벡터 x0에 대한 image 구하기
print("A*x0 =", A*x0)    # 표준행렬과 벡터의 곱 구하기
```

```
A =
[ 1  2  0]
[-1 -1  0]
[ 0  0  1]
[ 1  0  1]
```

```
T(x0) = (-4, 1, 3, 5)
A*x0 = (-4, 1, 3, 5)
```

<http://matrix.skku.ac.kr/2018-album/LA-Sec-6-1-lab.html> (실습하세요!)

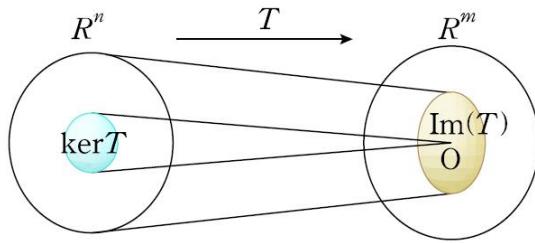
5.2 핵(kernel)과 치역(range)

참고 동영상: <http://youtu.be/9YciT9Bb2B0> <http://youtu.be/H-P4lDgruCc>

설습 사이트: <http://matrix.skku.ac.kr/knowls/cla-week-8-Sec-6-3.html>

정의. 핵(kernel), 단사, 전사, 전단사, 동형사상(isomorphism)

- (1) $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ 이 선형변환일 때, T 에 의한 상이 $\mathbf{0}$ 이 되는 \mathbb{R}^n 안의 벡터 전체의 집합을 T 의 핵(kernel)이라 하고 $\ker T$ 로 나타낸다. 즉
 $\ker T = \{\mathbf{v} \in \mathbb{R}^n \mid T(\mathbf{v}) = \mathbf{0}\}$



- (2) 변환 $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ 가 $T(\mathbf{u}) = T(\mathbf{v}) \Rightarrow \mathbf{u} = \mathbf{v}$ 를 만족하면 단사(one-to-one; injective)라 한다.
- (3) 선형변환 $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ 에 대하여, 임의의 $\mathbf{v} \in \mathbb{R}^n$ 의 상 $T(\mathbf{v})$ 전체의 집합을 T 의 치역(range)이라 하고 $\text{Im } T$ 로 나타낸다. 즉,
 $\text{Im } T = \{T(\mathbf{v}) \in \mathbb{R}^m \mid \mathbf{v} \in \mathbb{R}^n\} \subset \mathbb{R}^m$. 특히, $\text{Im } T = \mathbb{R}^m$, 즉 변환 $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ 가 임의의 $\mathbf{w} \in \mathbb{R}^m$ 에 대해 $T(\mathbf{v}) = \mathbf{w}$ 인 $\mathbf{v} \in \mathbb{R}^n$ 가 존재하면 전사(onto, surjective)라 한다.
- (4) 선형변환 T 가 단사이고 전사(전단사)이면 $n = m$ 이 되고, T 를 \mathbb{R}^n 에서 \mathbb{R}^n 으로의 동형사상(isomorphism)이라고 한다.

정리. 선형변환이 단사일 필요충분조건

$\mathbb{R}^n, \mathbb{R}^m$ 이 벡터공간이고 $T : \mathbb{R}^n \rightarrow \mathbb{R}^m$ 가 선형변환일 때, T 가 단사일 필요충분조건은 $\ker T = \{\mathbf{0}\}$ 이다.

예제 3 선형변환 $T : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ 를 $T(x, y, z) = (x + 2y - z, y + z, x + y - 2z)$ 로 정의하자. T 의 kernel을 구하고, T 가 단사인지 확인하여라.

```

var('x, y, z')
U = QQ^3    # 벡터공간 정의
h(x, y, z) = [x + 2*y - z, y + z, x + y - 2*z]    # 선형변환 함수 정의
T = linear_transformation(U, U, h)    # 선형변환 생성
print(T)
print()
TK = T.kernel()    # kernel을 구하기
print(TK)
print()
print("T is injective?", T.is_injective())    # 단사인지 확인

```

Vector space morphism represented by the matrix:

```

[ 1  0  1]
[ 2  1  1]
[-1  1 -2]

```

Domain: Vector space of dimension 3 over Rational Field

Codomain: Vector space of dimension 3 over Rational Field

Vector space of degree 3 and dimension 1 over Rational Field

Basis matrix:

```
[ 1 -1/3  1/3]    # kernel이 span([1, -1/3, 1/3])임을 뜻한다.
```

T is injective? False]

따라서 $\ker(T) \neq \{0\}$ 이므로 주어진 선형변환 T 는 단사가 아니다.

예제 4 $A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}$ 이라 하자. $T_A : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ 는 단사이지만, 전사는 아니다.

```
U = QQ^2    # 벡터공간 정의
V = QQ^3    # 벡터공간 정의
A = matrix(QQ, [[1, 0], [0, 1], [0, 0]])    # 변환행렬 정의
TA = linear_transformation(U, V, A, side='right')    # 선형변환 정의
print(TA)
print()
print(TA.kernel())    # kernel 구하기
print()
print("TA is injective?", TA.is_injective())    # 단사인지 확인
print()
print(TA.image())    # 치역 구하기
print()
print("TA is surjective ?", TA.is_surjective())    # 전사인지 확인
```

Vector space morphism represented by the matrix(행렬표현):

[1 0 0]

[0 1 0]

정의역(Domain: Vector space of dimension 2 over Rational Field)

공변역(Codomain: Vector space of dimension 3 over Rational Field)

Vector space of degree 2 and dimension 0 over Rational Field

Basis matrix:

[]

TA is injective? True (단사이다)

Vector space of degree 3 and dimension 2 over Rational Field

Basis matrix:

[1 0 0]

[0 1 0]

TA is surjective? False (전사는 아니다)

<http://matrix.skku.ac.kr/2018-album/LA-Sec-6-3-lab.html> (실습하세요!)



읽을거리

여인자 전개 – 도지슨(Charles Dodgson, 1832~1898,
필명은 Lewis Carroll)

소설 ‘이상한 나라의 앤리스’ 저자인 영국의 수학자 도지슨은 큰 크기의 행렬에 대한 행렬식을 작은 크기 행렬의 행렬식들의 합으로 표현할 수 있다는 여인자(Cofactor) 전개 방법을 처음으로 소개하였다. 라플라스는 이를 확장하여 어떤 행, 어떤 열에 대해서도 이 여인자 전개 방법이 적용된다는 것을 보인 것이다. 여인자 전개는 최근 컴퓨터 병렬 처리에 효과적인 도구로 새롭게 부활되었다. 그는 사진 찍는 것을 매우 좋아하며 여러 권의 잘 알려진 책을 쓴 유명한 작가이며 뛰어난 수학자였다.



<개교 150년 기념하는 2012년 1월 MIT 본관>

6. 고윳값, 고유벡터, 대각화(Diagonalization)

7강 동영상, 행렬의 대각화 <https://youtu.be/d8KE1QpKiDo> (37:11)

6.1 고윳값과 고유벡터

참고 동영상: <http://youtu.be/OImrmmWXuvU> <http://youtu.be/96Brbkx1cQ4>

실습 사이트: <http://matrix.skku.ac.kr/knowls/CLA-Week-6-Sec-4-5.html>

정의. 고윳값(eigenvalue), 고유벡터(eigenvector)

A 를 n 차의 정사각행렬이라 하자. $\mathbf{0}$ 아닌 벡터 $\mathbf{x} \in \mathbb{R}^n$ 가 적당한 스칼라 λ 에 대하여 다음을 만족하면 λ 를 A 의 고윳값(eigenvalue)이라 하고, \mathbf{x} 를 λ 에 대응하는 A 의 고유벡터(eigenvector)라고 한다.

$$A\mathbf{x} = \lambda\mathbf{x}$$

- $\mathbf{x} \in \mathbb{R}^n$ 가 고윳값 λ 에 대응하는 A 의 고유벡터이면 영 아닌 임의의 스칼라 k 에 대하여 $k\mathbf{x}$ 도 λ 에 대응하는 A 의 고유벡터가 된다.

$$A\mathbf{x} = \lambda\mathbf{x} \Rightarrow A(k\mathbf{x}) = k(A\mathbf{x}) = k(\lambda\mathbf{x}) = \lambda(k\mathbf{x})$$

예제 1 행렬 $A = \begin{bmatrix} 1 & 2 & 2 \\ 1 & 2 & -1 \\ 3 & -3 & 0 \end{bmatrix}$ 의 고윳값과 고유벡터를 구하여라.

[실습 : <http://sage.skku.edu/>]

```
A = matrix([[1, 2, 2], [1, 2, -1], [3, -3, 0]])
print(A.eigenvalues()) # A의 고윳값 구하기
print(A.eigenvectors_right()) # A의 고유벡터 구하기: (고윳값, [고유벡터], 중복도)
```

```
[-3, 3, 3]
[(-3, [(1, -1/2, -3/2)], 1), (3, [(1, 0, 1), (0, 1, -1)], 2)]
```

따라서 행렬 A 의 고윳값의 $-3, 3$ 이다. 그리고 -3 에 대응하는 고유벡터는 $\left(r, -\frac{1}{2}r, -\frac{3}{2}r\right)$ ($r \in \mathbb{R} \setminus \{0\}$)이고, 3 에 대응하는 고유벡터는 $(t, 0, t) + (0, s, -s)$

$(t, s \in \mathbb{R} \setminus \{0\})$)이다.

<http://matrix.skku.ac.kr/2018-album/LA-Sec-4-5-lab.html> (실습하세요!)

Note: A 의 서로 다른 고윳값들에 대응하는 고유벡터들은 모두 직교집합입니다.

6.2 닮음 행렬(similar matrix)과 행렬의 대각화(Matrix Diagonalization)

참고 동영상: <http://youtu.be/xirjNZ40kRk> <http://youtu.be/MnfLcBZsV-I>

실습 사이트: <http://matrix.skku.ac.kr/knowls/cla-week-11-sec-8-2.html>

정의. 닮은(similar) 행렬

정사각행렬 A, B 에 대하여 다음을 만족하는 가역행렬 P 가 존재할 때 B 는 A 와 닮은(similar) 행렬이라고 한다.

$$B = P^{-1} A P$$

이때, $B \sim A$ 라 쓴다.

(모든 행렬은 주대각선 성분이 고윳값인 삼각행렬과 닮음이고, 삼각행렬의 특성방정식은 쉽게 구할 수 있으며) 닮음행렬들은 행렬식이 같기 때문에, 특성방정식, 고윳값이 같다는 것을 아주 쉽게 보일 수 있다. 따라서 주어진 행렬문제는 좀 더 단순한 모양의 닮음행렬을 찾아서 쉽게 처리할 수 있다.

정의. 대각화가능한(diagonalizable) 행렬

A 가 어떤 대각선행렬과 닮은 행렬일 때, 즉 적당한 가역행렬 P 가 존재하여 $P^{-1}AP$ 가 대각선행렬일 때 A 를 대각화가능한(diagonalizable) 행렬이라 하며, 이 때 행렬 P 를 A 를 대각화하는(diagonalizing) 행렬이라고 한다.

정리. 대각화가능할 필요충분조건

n 차의 정사각행렬 A 가 대각화가능할 필요충분조건은 A 가 n 개의 일차독립인 고유벡터를 갖는 것이다. 이때, A 는 자신의 고윳값 $\lambda_1, \dots, \lambda_n$ 을 주대각선성분으로

갖는 대각선행렬 D 와 닮은 행렬이다.

■ A 를 대각화하는 행렬 P 를 구하는 과정

1단계: A 의 n 개의 일차독립인 고유벡터 $\mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \dots, \mathbf{p}^{(n)}$ 을 구한다.

2단계: $\mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \dots, \mathbf{p}^{(n)}$ 을 열벡터로 갖는 행렬 P 를 만든다.

3단계: 이 P 가 A 를 대각화하는 행렬이고 $P^{-1}AP$ 는 A 의 대응하는 고윳값 $\lambda_1, \dots, \lambda_n$ 을 순서대로 주 대각선 성분으로 갖는 대각선행렬 D 이다.

$$D = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$$

예제 2 행렬 $A = \begin{bmatrix} 0 & 0 & -2 \\ 1 & 2 & 1 \\ 1 & 0 & 3 \end{bmatrix}$ 가 대각화가능함을 보이고, 이때 A 를 대각화하는 행렬 P 와 대각선행렬 D 를 구하여라.

[실습 : <http://sage.skku.edu/>]

```
A = matrix(QQ, [[0, 0, -2], [1, 2, 1], [1, 0, 3]])
print(A.eigenvectors_right()) # 고유벡터 구하기: (고윳값, [ 고유벡터 ], 중복도)
print(A.is_diagonalizable()) # 대각화 가능 여부 확인
```

```
[(1, [(1, -1/2, -1/2)], 1), (2, [(1, 0, -1), (0, 1, 0)], 2)]
True
```

따라서 행렬 A 는 n 개의 일차독립인 고유벡터를 갖는다. 그러므로 행렬 A 는 대각화가능이다.

```
x1 = vector([1, -1/2, -1/2]) # 첫 번째 고유벡터 정의
x2 = vector([1, 0, -1])       # 두 번째 고유벡터 정의
x3 = vector([0, 1, 0])        # 세 번째 고유벡터 정의
P = column_matrix([x1, x2, x3]) # 3개의 고유벡터를 순서대로 열벡터로 하는 행렬
print("P =")
print(P)
```

```

print()
print("D =")
print(P^-1*A*P) # 대각화 가능하면, 고윳값을 대각원소로 하는 대각행렬 생성됨

```

```

P=
[ 1   1   0]
[-1/2  0   1]
[-1/2 -1   0]

```

```

D=
[1 0 0]
[0 2 0]
[0 0 2]

```

<http://matrix.skku.ac.kr/2018-album/LA-Sec-8-2-lab.html>

6.3 직교대각화 (orthogonally diagonalizing)

참고 동영상: <http://youtu.be/jimlkBGAZfQ> <http://youtu.be/B--ABwoKAN4>

실습 사이트: <http://matrix.skku.ac.kr/knou-knowls/cla-week-11-sec-8-3.html>

정의. **직교행렬(real orthogonal matrix, $A^T A = I$, $A^{-1} = A^T$)**

정사각행렬 A 에 대하여 $A^{-1} = A^T$ 이면 A 를 **직교행렬(real orthogonal matrix)**이라고 한다.

정리. **직교행렬 정리**

행렬 A 가 직교행렬이면 다음을 만족한다.

- (1) 행렬 A 의 행벡터들은 서로 직교이며, 정규벡터이다. (행들이 정규직교벡터이다)
- (2) 행렬 A 의 열벡터들은 서로 직교이며, 정규벡터이다. (열들이 정규직교벡터이다)
- (3) A 는 가역행렬이다. ($A^{-1} = A^T$, $A^T A = I$)
- (4) $\|Ax\| = \|\mathbf{x}\|$ 를 만족한다. (즉, 길이를 보존한다)

$$\langle Ax, Ax \rangle = (Ax)^T Ax = \mathbf{x}^T A^T Ax = \langle \mathbf{x}, \mathbf{x} \rangle$$

- 직교행렬의 역행렬은 단지 전치행렬을 쓰기만 해도 구할 수 있다. ($A^{-1} = A^T$)

정의. 직교대각화

- (1) 만일 A 와 C 가 같은 크기의 정사각행렬이라 할 때, $C = P^TAP$ 인 직교행렬 P 가 존재하면, C 는 A 에 직교닮음(orthogonally similar)이라고 한다.
- (2) 정사각행렬 A 에 대하여 A 를 대각화하는 직교행렬 P 가 존재할 때 A 는 직교 대각화가능(orthogonally diagonalizable)하다고 하며 P 는 A 를 직교대각화하는 (orthogonally diagonalizing) 행렬이라고 한다.

정리. 직교대각화가능할 필요충분조건

n 차 정사각행렬 A 가 직교대각화가능할 필요충분조건은 A 가 대칭행렬인 것이다.

예제 3 대칭행렬 $A = \begin{bmatrix} -1 & -1 & 1 \\ -1 & 2 & 4 \\ 1 & 4 & 2 \end{bmatrix}$ 를 직교대각화하는 행렬 P 를 구하여라.
 $(P^TAP = D)$

풀이]. A 의 특성다항식은 $|\lambda I_3 - A| = \lambda(\lambda + 3)(\lambda - 6) = 0$ 이므로 A 의 고윳값은 $\lambda_1 = -3$, $\lambda_2 = 0$, $\lambda_3 = 6$ 이고, 대칭행렬의 서로 다른 고윳값에 대응하는 고유벡터는 모두 직교집합(o.g.)이고 각각 다음과 같다.

$$\mathbf{x}_1 = \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}, \quad \mathbf{x}_3 = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

직교집합인 $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3 (= \mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3)$ 를 간단히 정규화(o.n.)하면 (Find \mathbb{R}^3 의 정규직교기저 $Z = \{\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3\}$)

$$\left\{ \begin{bmatrix} -\frac{1}{\sqrt{3}} \\ -\frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \end{bmatrix}, \begin{bmatrix} \frac{2}{\sqrt{6}} \\ -\frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} \end{bmatrix}, \begin{bmatrix} 0 \\ \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} \right\} \therefore P = \begin{bmatrix} -\frac{1}{\sqrt{3}} & \frac{2}{\sqrt{6}} & 0 \\ -\frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{2}} \end{bmatrix} \quad (\textcolor{blue}{P^TAP = \begin{bmatrix} -3 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 6 \end{bmatrix}}) \blacksquare$$

예제 4 행렬 $A = \begin{bmatrix} 0 & 3 & 3 \\ 3 & 0 & 3 \\ 3 & 3 & 0 \end{bmatrix}$ 의 고윳값은 $\lambda_1 = \lambda_2 = -3$ (중복도 2), $\lambda_3 = 6$ 이었다.

$\lambda_1 = -3$ 에 대응하는 두 개의 일차독립인 고유벡터는 (일반적으로는 직교가 아닐 수 있다)

$$\mathbf{x}_1 = \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

이고 Gram-Schmidt 정규직교화 과정을 이용하면

$$\mathbf{y}_1 = \mathbf{x}_1 = \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{y}_2 = \mathbf{x}_2 - \frac{\mathbf{x}_2 \cdot \mathbf{y}_1}{\|\mathbf{y}_1\|^2} \mathbf{y}_1 = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} - \frac{1}{2} \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -\frac{1}{2} \\ \frac{1}{2} \\ 1 \end{bmatrix}$$

$$\Rightarrow \mathbf{z}_1 = \frac{\mathbf{y}_1}{\|\mathbf{y}_1\|} = \begin{bmatrix} -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \\ 0 \end{bmatrix}, \quad \mathbf{z}_2 = \frac{\mathbf{y}_2}{\|\mathbf{y}_2\|} = \begin{bmatrix} -\frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} \\ \frac{2}{\sqrt{6}} \end{bmatrix}$$

$\lambda_3 = 6$ 에 대응하는 고유벡터는 $\mathbf{x}_3 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$ 이고 (서로 다른 고윳값에 대응하는 고유벡터들과는 이미 직교이므로) 정규화만하면 된다. $\mathbf{z}_3 = \begin{bmatrix} \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \end{bmatrix}$. 따라서 \mathbb{R}^3 의 정규직교

기저 $Z = \{\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3\}$ 을 열벡터로 하는 행렬 P 를 정의하면 된다.

즉, $P^T A P = \begin{bmatrix} -3 & 0 & 0 \\ 0 & -3 & 0 \\ 0 & 0 & 6 \end{bmatrix}$ 하는 직교행렬은 $P = [\mathbf{z}_1 : \mathbf{z}_2 : \mathbf{z}_3] = \begin{bmatrix} -\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{3}} \\ 0 & \frac{2}{\sqrt{6}} & \frac{1}{\sqrt{3}} \end{bmatrix}$ 이다. ■

예제 5 행렬 $A = \begin{bmatrix} 4 & 2 & 2 & 0 \\ 2 & 4 & 2 & 0 \\ 2 & 2 & 4 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}$ 가 직교대각화가능함을 보이고, 이때 A 를 직교대각화 하는 직교행렬 P 와 대각선행렬 D 를 구하여라.

```
A = matrix([[4, 2, 2, 0], [2, 4, 2, 0], [2, 2, 4, 0], [0, 0, 0, 4]]) # 행렬 입력
print(A.eigenvectors_right())
```

```
[(8, [(1, 1, 1, 0)], 1), (4, [(0, 0, 0, 1)], 1), (2, [(1, 0, -1, 0), (0, 1, -1, 0)], 2)]
```

* 고윳값은 $\lambda_1 = 8$, $\lambda_2 = 4$, $\lambda_3 = \lambda_4 = 2$ (중복도 2) 이다.

대칭행렬의 서로 다른 고윳값에 대응하는 고유벡터는 모두 직교집합이므로, $\lambda = 8$ 에 대응하는

고유벡터 $\mathbf{x}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$, $\lambda = 4$ 에 대응하는 고유벡터 $\mathbf{x}_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$ 는 이미 직교이다. 그러나 $\lambda = 2$

에 대응하는 두 개의 일차독립인 고유벡터 $\mathbf{x}_3 = \begin{bmatrix} 1 \\ 0 \\ -1 \\ 0 \end{bmatrix}$, $\mathbf{x}_4 = \begin{bmatrix} 0 \\ 1 \\ -1 \\ 0 \end{bmatrix}$ 는 직교가 아닐 수 있다.

여기에서 Gram-Schmidt 정규직교화 과정을 이용하면 다음을 얻는다.

```
x1 = vector([1, 1, 1, 0])
x2 = vector([0, 0, 0, 1])
x3 = vector([1, 0, -1, 0])
x4 = vector([0, 1, -1, 0])
B = matrix([x3, x4])           # x3, x4를 행벡터로 하는 행렬 생성
[G, mu] = B.gram_schmidt()    # 행에 대하여 직교기저를 찾아준다. B==mu*G
print(G)
```

```
[ 1  0 -1  0]
[-1/2  1 -1/2  0]
```

따라서 행렬 A 를 직교대각화 하는 직교행렬을 P 라 하면 A 와 직교닮음인 대각행렬 D 는 다음과 같다.

[실습 : <http://sage.skku.edu/>]

```

y3 = vector([1, 0, -1, 0])
y4 = vector([-1/2, 1, -1/2, 0])
C = column_matrix([x1, x2, y3, y4])
P = column_matrix([C.column(i) / C.column(i).norm() for i in range(0, 4)])
print("P =")
print(P)
print()
print(P.transpose()*A*P)  # 직교 닮음인 대각행렬

```

```

P =
[ 1/3*sqrt(3)      0      1/2*sqrt(2)   -1/3*sqrt(3/2)]
[ 1/3*sqrt(3)      0       0      2/3*sqrt(3/2)]
[ 1/3*sqrt(3)      0     -1/2*sqrt(2)   -1/3*sqrt(3/2)]
[    0            1       0           0        ]

```



```

[8 0 0 0]
[0 4 0 0]
[0 0 2 0]
[0 0 0 2]

```

<http://matrix.skku.ac.kr/2018-album/LA-Sec-8-3-lab.html> (실습하세요!)

■ 고윳값 분해(eigen-decomposition)

정사각행렬 A 가 대각화 가능하다면 대각선행렬 D 와 가역행렬 P 가 존재하여 다음이 성립한다.

$$P^{-1}AP = D \Leftrightarrow A = PDP^{-1}$$

특히 A 가 (실수) 대칭행렬일 때는 위를 만족하는 대각선행렬 D 와 직교행렬 P ($\Rightarrow P^T P = I$, $P^{-1} = P^T$)가 존재한다. 만일 λ_i 와 \mathbf{u}_i , $i = 1, \dots, n$ 를 각각 A 의 n 개의 ($\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$) 고윳값과 대응하는 n 개의 (정규직교) 고유벡터라 하면, $P = [\mathbf{u}_1 : \mathbf{u}_2 : \dots : \mathbf{u}_n]$ 는 직교행렬 ($\Rightarrow P^T P = I$)이 되고,

$$D = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix} = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n) \quad \text{는 대각선 행렬이며, 다음이 성립한다.}$$

$$P^T A P = D \Leftrightarrow A = P D P^T.$$

이때 D 의 주대각선 성분은 A 의 고윳값이고, P 의 열벡터는 그에 대응하는 n 개의 정규직교 고유 벡터 (따라서 \mathbb{R}^n 의 o.n. basis가 된다)이다. 따라서 아래가 성립한다.

$$A = P D P^T = \lambda_1 \mathbf{u}_1 \mathbf{u}_1^T + \lambda_2 \mathbf{u}_2 \mathbf{u}_2^T + \cdots + \lambda_n \mathbf{u}_n \mathbf{u}_n^T \quad (\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n).$$

이 분해는 고윳값 λ_i 와 그에 대응하는 고유벡터 \mathbf{u}_i 만 이용하므로 행렬 A 의 고윳값 분해 (eigen-decomposition)라 하며, Spectral decomposition이라고도 부른다. 여기서 모든 $\mathbf{u}_i \mathbf{u}_i^T$ 가 rank가 1 행렬이고 A 가 rank 1 행렬들의 합이므로 행렬 A 의 rank 1 decomposition이라고도 한다.

(o) 정리는 복소수 성분 행렬인 경우에도 유사하게 성립한다. Normal matrix!)

정리.

$$\det(CD) = \det(D)\det(C) \text{이고 } \text{tr}(CD) = \text{tr}(DC), \quad P^{-1}AP = D \Leftrightarrow A = PDP^{-1}$$

n 차의 두 정사각행렬 A, B 가 닮음이면 다음이 성립한다.

$$(1) \quad \det(A) = \det(B) \quad (\det(B) = \det(P^{-1}AP) = \det(P^{-1})\det(A)\det(P) = \det(A))$$

$$(2) \quad \text{tr}(A) = \text{tr}(B) \quad (\text{tr}(B) = \text{tr}(P^{-1}AP) = \text{tr}(PP^{-1}A) = \text{tr}(A))$$

■ n 차 정사각행렬 A 의 고윳값 분해가 존재할 때, 다음 계산을 쉽게 할 수 있다.

$$(1) \quad \det(A) = \det(PDP^{-1}) = (\det P)(\det D)(\det P^{-1})$$

$$= (\det P)(\det P^{-1})(\det D) = (\det PP^{-1})(\det D) = \det(D) = \lambda_1 \lambda_2 \cdots \lambda_n$$

$$(2) \quad \text{tr}(A) = \text{tr}(PDP^{-1}) = \text{tr}(DP^{-1}P) = \text{tr}(D) = \sum_{i=1}^n \lambda_i$$

$$(3) A^k = (PDP^{-1})(PDP^{-1}) \cdots (PDP^{-1}) = PD(P^{-1}P)D(P^{-1} \cdots P)DP^{-1}$$

$$= PD^k P^{-1} = P \operatorname{diag}(\lambda_1^k, \dots, \lambda_n^k) P^{-1}$$

$$(4) A^{-1} = (PDP^{-1})^{-1} = PD^{-1}P^{-1} = P \operatorname{diag}(\lambda_1^{-1}, \dots, \lambda_n^{-1}) P^{-1} \quad (\text{역행렬을 구하는 새 방법})$$

7. SVD (특이값 분해, 특잇값 분해, singular value decomposition)

8강 동영상, singular value decomposition <https://youtu.be/e0IoDqJLB8U> (23:03)

7.1 특이값 분해 (singular value decomposition)

참고 동영상: <https://youtu.be/ejCge6Zjf1M>

실습 사이트: <http://matrix.skku.ac.kr/knowls/cla-week-12-sec-8-6.html>

- 고윳값 분해는 대각화 가능한 정사각행렬인 경우에만 정의되는 개념이다. 따라서 일반적인 행렬의 경우에 대해서도 고윳값 분해와 유사한 행렬 분해가 존재하는지 생각해 볼 수 있다.

임의의 행렬 A 에 대해서 $A^T A$ 와 $A A^T$ 는 대칭행렬이므로 직교 대각화가 가능하다. 이를 이용하면 다음의 중요한 행렬분해인 특이값 분해(singular value decomposition, SVD)를 얻을 수 있다.

정리. 특이값 분해(SVD) 정리 [Key Idea 1]

- (1) 행렬 A 를 $m \times n$ 의 실수 행렬이라 하자. 그러면 다음과 같은 직교(orthogonal) 행렬 U , V 와 대각선행렬 Σ 가 존재한다.

$$U^T A V = \begin{bmatrix} \Sigma_1 & O \\ O & O \end{bmatrix} = \Sigma \quad (U \in M_{m \times m}, \Sigma \in M_{m \times n}, V \in M_{n \times n}) \quad (1)$$

여기서 Σ_1 은 주대각선성분이 모두 (단조감소의 순서로 배열된) 양수이고, 가역인 대각선행렬, O 은 영행렬이다. 즉,

$$A = U\Sigma V^T = [\mathbf{u}_1 \mathbf{u}_2 \cdots \mathbf{u}_k \mathbf{u}_{k+1} \cdots \mathbf{u}_m] \begin{bmatrix} \sigma_1 & & 0 & | & 0 & \cdots & 0 \\ \sigma_2 & \ddots & & | & 0 & \cdots & 0 \\ \vdots & & \vdots & | & \vdots & & \vdots \\ 0 & & \sigma_k & | & 0 & \cdots & 0 \\ \hline 0 & 0 & \cdots & 0 & | & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & | & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & | & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_n^T \end{bmatrix}$$

(단 $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k > 0$, 단조감소의 순서)

위의 행렬 Σ 의 대각선성분들을 행렬 A 의 특이값(singular value)들이라 하고, U 의 열들을 A 의 left singular vector, V 의 열들을 A 의 right singular vector라고 한다.

(2) V 와 U 는 각각 대칭행렬 $\mathbf{A}^T \mathbf{A}$ 와 AA^T 를 직교대각화하는 직교행렬이다. 즉 $m \times n$ ($m \geq n$) 크기의 행렬 $A = U\Sigma V^T$ 를 특이값 분해라 하고 r 을 행렬 A 의 계수(rank)라 하자. 그러면 다음이 성립한다. ($U \in M_{m \times m}$, $\Sigma \in M_{m \times n}$, $V \in M_{n \times n}$) (보통 A 의 앞에 \mathbf{A}^T 를 곱하여 $\mathbf{A}^T \mathbf{A}$ 의 고유벡터, 즉 A 의 오른쪽에 대응하는 A 의 right singular vector를 먼저 구한다)

$$V^T(\mathbf{A}^T \mathbf{A}) V = \text{diag}(\sigma_1^2, \sigma_2^2, \dots, \sigma_r^2, 0, \dots, 0)_{n \times n} \quad (\text{※ 행렬의 크기 } n \times n \text{에 주의!})$$

$$U^T(AA^T)U = \text{diag}(\sigma_1^2, \sigma_2^2, \dots, \sigma_r^2, 0, \dots, 0)_{m \times m} \quad (\text{※ 행렬의 크기 } m \times m \text{에 주의!})$$

<http://matrix.skku.ac.kr/2018-album/SVD.html> 을 실습!

- 고윳값 분해의 경우와 마찬가지로 A 의 SVD가 주어지면 다음과 같이 표현이 가능하다.

$$A = U\Sigma V^T = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^T + \cdots + \sigma_k \mathbf{u}_k \mathbf{v}_k^T \quad < A \text{의 특이값분해(SVD)} >$$

예제 1 행렬 $A = \begin{bmatrix} \sqrt{3} & 2 \\ 0 & \sqrt{3} \end{bmatrix}$ 의 특이값분해(SVD)를 구하여라.

[실습 : <http://sage.skku.edu/>]

```

A = matrix([[sqrt(3), 2], [0, sqrt(3)]])
B = A.transpose()*A
eig = B.eigenvalues()
sv = [sqrt(i) for i in eig]           # 특이값 구하기
print(B.eigenvectors_right())        #  $A^T A$ 의 고유벡터 구하기,  $A$ 의 right singular
vector들

```

```
[9, [(1, sqrt(3)), 1), (1, [(1, -1/3*sqrt(3))], 1)]
```

```

G = matrix([[1, sqrt(3)], [1, -1/3*sqrt(3)]])
Vh = matrix([1/G.row(j).norm()*G.row(j) for j in range(0,2)])  # V의 전치행렬
Vh = Vh.simplify()      # V의 전치행렬 표현
print(Vh)
print()
U = matrix([A*Vh.row(j)/sv[j] for j in range(0,2)]).transpose()  # U
print(U)                # U의 열들을 A의 left singular vector들
print()
S = diagonal_matrix(sv)
print(S)
print()
print(U*S*Vh)

```

```
[ 1/2 1/2*sqrt(3)]
[1/2*sqrt(3) -1/2]
```

```
[ 1/2*sqrt(3) 1/2]
[ 1/2 -1/2*sqrt(3)]
```

```
[3 0]
[0 1]
```

```
[sqrt(3) 2]
[ 0 sqrt(3)]
```

7.2 일반화된 역행렬 (pseudo-inverse, Moore–Penrose Generalized Inverse)

참고 동영상: <http://youtu.be/7-qG-A8nXmo>

실습 사이트: <http://matrix.skku.ac.kr/knowls/cla-week-12-sec-8-6.html>

- 일반화된 역행렬 (pseudo-inverse) : 최소제곱해 연구에 중요하다.
- A 가 크기 $n \times n$ 인 가역행렬인 경우 특이값 분해에 의해

$$A = U\Sigma V^T \quad (2)$$

로 표현할 수 있다. 여기에서, U , Σ , V 는 모두 $n \times n$ 인 가역행렬들이고, 특히 U , V 는 직교행렬들이다. 그러므로 A 의 역행렬은 다음과 같이 표현된다.

$$A^{-1} = V\Sigma^{-1}U^T \quad (3)$$

이제 행렬 A 가 정사각행렬이 아니거나, 비가역인 정사각행렬인 경우에는 (3)식은 적용할 수 없다. 그러나 (2)의 가운데 행렬을 $\Sigma = \begin{bmatrix} \Sigma_1 & O \\ O & O \end{bmatrix}$ (여기서 Σ_1 은 가역행렬)로 생각하면 모든 행렬 A 에 대한 행렬곱 $V\Sigma'U^T$ 을 다음과 같이 정의할 수 있다.

$$A^\dagger = V\Sigma'U^T = V\begin{bmatrix} \Sigma_1^{-1} & O \\ O & O \end{bmatrix}U^T$$

정의. pseudo-inverse (Moore–Penrose Generalized Inverse)

행렬 A 가 크기 $m \times n$ 인 경우 크기 $n \times m$ 인 행렬 $A^\dagger = V\Sigma'U^T$ 를 행렬 A 의 pseudo-inverse라 한다. 여기서 U , V 는 직교행렬이고, Σ' 은 다음과 같은 행렬이다.

$$\Sigma' = \begin{bmatrix} \Sigma_1^{-1} & O \\ O & O \end{bmatrix} \quad (\text{여기서 } \Sigma_1 \text{은 가역행렬})$$

정리. full column rank를 갖는 $m \times n$ 행렬의 pseudo-inverse

[pseudo-inverse (Moore–Penrose Generalized Inverse의 특수한 경우)]

행렬 A 가 (n 개의 열이 모두 일차독립인) full column rank를 갖는 $m \times n$ 행렬이

면, $A\mathbf{x} = \mathbf{b}$ 의 양변에 A^T 를 곱해준 $(A^T A)\mathbf{x} = A^T \mathbf{b}$ 를 정규방정식(normal equation)이라하고, 이 정규방정식은 언제나 유일해 $\mathbf{x} = (A^T A)^{-1} A^T \mathbf{b}$ 를 갖는다. 이 때 $A^\dagger = (A^T A)^{-1} A^T$ 을 A 의 pseudo-inverse (Moore-Penrose Generalized Inverse의 특수한 경우)라고 한다.

예제 2 (rank 2인) 행렬 $A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$ 의 Pseudo-inverse $A^\dagger = (A^T A)^{-1} A^T$ 를 구하여라.

[실습 : <http://sage.skku.edu/>]

```
A = matrix([[1, 1], [0, 1], [1, 0]])
print("A =")
print(A)
print()
print("Pseudo-inverse of A =")
print((A.transpose()*A).inverse()*A.transpose()) # Pseudo-inverse 구하기
```

```
A=
[1 1]
[0 1]
[1 0]
```

```
Pseudo-inverse of A=
[ 1/3 -1/3  2/3]
[ 1/3  2/3 -1/3]
```

정리. 최소제곱해(least square solution)

A 가 $m \times n$ 행렬이고, \mathbf{b} 는 \mathbb{R}^n 의 임의의 벡터이면, $\mathbf{x} = A^\dagger \mathbf{b}$ 는 (최소의 에러를 갖는) $A\mathbf{x} = \mathbf{b}$ 의 최소제곱해이다.

예제 3 다음은 컴퓨터 과학 전공 학생 105명의 고등학교 성적(GPA)과 대학교 성적(GPA)의 데이터를 가지고 최소제곱직선을 구하는 과정이다. 이를 통해 어떤 학생의 고등학교 성적을 알고 있다면, 이 학생의 대학교 성적을 예측해볼 수 있다. 자료의 출처는 다음과 같다.

[출처] <http://onlinestatbook.com/2/regression/intro.html>

고등학교 성적을 x , 대학교 성적을 y , 선형모델(최소제곱직선)을 $y = c + dx$ 라 하자. 그리고 데이터를 이용하여 이를 행렬로 나타내자. 그러면 우리가 풀고자 하는 문제는

$$A = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} c \\ d \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \text{ 라 할 때 } \min \| A\mathbf{x} - \mathbf{b} \|$$

이다. 이때 A 의 열벡터들이 일차독립(이를 A 가 full column rank를 갖는다고 한다)이면, $A\mathbf{x} = \mathbf{b}$ 의 정규방정식(normal equation) $A^T A\mathbf{x} = A^T \mathbf{b}$ 으로 부터 $\hat{\mathbf{x}} = (A^T A)^{-1} A^T \mathbf{b}$ 을 얻는다.

Sage 코드를 이용하여 확인하면 <http://sage.skku.edu/>

```
x1 = vector([1 for i in range(105)]) # 첫 번째 열 생성
x2 = vector([3.45, 2.78, 2.52, 3.67, 3.24, 2.1, 2.82, 2.36, 2.42, 3.51, 3.48, 2.14,
            2.59, 3.46, 3.51, 3.68, 3.91, 3.72, 2.15, 2.48, 3.09, 2.71, 2.46, 3.32,
            3.61, 3.82, 2.64, 2.19, 3.34, 3.48, 3.56, 3.81, 3.92, 4, 2.52, 2.71,
            3.15, 3.22, 2.29, 2.03, 3.14, 3.52, 2.91, 2.83, 2.65, 2.41, 2.54, 2.66,
            3.21, 3.34, 3.68, 2.84, 2.74, 2.71, 2.24, 2.48, 3.14, 2.83, 3.44, 2.89,
            2.67, 3.24, 3.29, 3.87, 3.94, 3.42, 3.52, 2.24, 3.29, 3.41, 3.56, 3.61,
            3.28, 3.21, 3.48, 3.62, 2.92, 2.81, 3.11, 3.28, 2.7, 2.62, 3.72, 3.42,
            3.51, 3.28, 3.42, 3.9, 3.12, 2.83, 2.09, 3.17, 3.28, 3.02, 3.42, 3.06,
            2.76, 3.19, 2.23, 2.48, 3.76, 3.49, 3.07, 2.19, 3.46]) # 두 번째 열 생성
A = column_matrix([x1, x2]) # 계수행렬
b = vector([3.52, 2.91, 2.4, 3.47, 3.47, 2.37, 2.4, 2.24, 3.02, 3.32, 3.59, 2.54,
            3.19, 3.71, 3.58, 3.4, 3.73, 3.49, 2.25, 2.37, 3.29, 3.19, 3.28, 3.37,
            3.61, 3.81, 2.4, 2.21, 3.58, 3.51, 3.62, 3.6, 3.65, 3.76, 2.27, 2.35,
            3.17, 3.47, 3, 2.74, 3.37, 3.54, 3.28, 3.39, 3.28, 3.19, 2.52, 3.08,
            3.01, 3.42, 3.6, 2.4, 2.83, 2.38, 3.21, 2.24, 3.4, 3.07, 3.52, 3.47,
            3.08, 3.38, 3.41, 3.64, 3.71, 3.01, 3.37, 2.34, 3.29, 3.4, 3.38, 3.28,
```

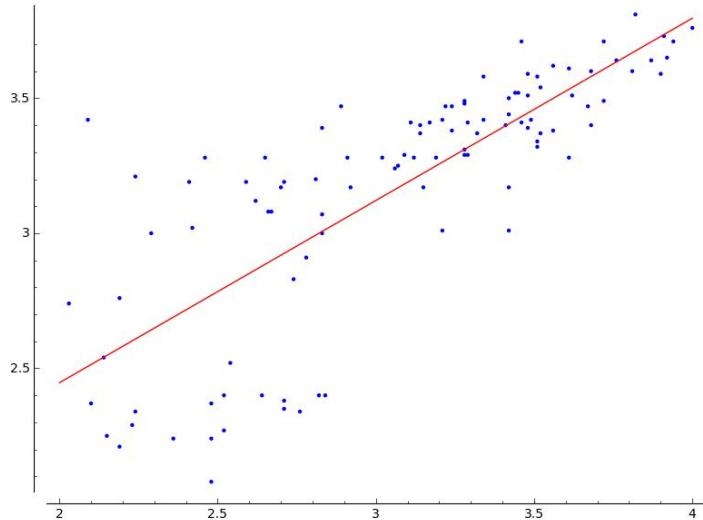
```

3.31, 3.42, 3.39, 3.51, 3.17, 3.2, 3.41, 3.29, 3.17, 3.12, 3.71, 3.5,
3.34, 3.48, 3.44, 3.59, 3.28, 3, 3.42, 3.41, 3.49, 3.28, 3.17, 3.24,
2.34, 3.28, 2.29, 2.08, 3.64, 3.42, 3.25, 2.76, 3.41]) # 상수항 벡터
c, d = n((A.transpose()*A).inverse()*A.transpose()*b, digits = 5) # 최소제곱해
print(c, d)

p1 = point([(x2[i], b[i]) for i in range(105)]) # 데이터를 좌표평면에 그리기
p2 = plot(c + d*x, (x, 2, 4), color = 'red') # 최소제곱직선 그리기
p1 + p2

```

1.0968 0.67483 # $y = c + dx$, $c=1.0968$ $d=0.67483$



따라서 $y = 1.0968 + 0.67483x$ 임을 알 수 있다. 데이터와 선형모델(최소제곱직선) $y = c + dx$ 을 그리면 위와 같다.

<http://matrix.skku.ac.kr/sglee/project/generalized-inverse/> (일반화된 역행렬의 성질들!)

<http://matrix.skku.ac.kr/2018-album/LS-QR-decom.html> (최소제곱해)

8. 이차형식(quadratic form)

9강 동영상, 이차형식 <https://youtu.be/rCNBWT0r5mA> (36:26)

8.1 이차형식

참고 동영상: <http://youtu.be/vWzHWEhAd-k> http://youtu.be/lznsULrqJ_0

실습 사이트: <http://matrix.skku.ac.kr/knowls/cla-week-12-sec-8-4.html>

- 이차형식(quadratic form)은 각 항이 2차인 다항식으로서 수학, 물리학, 경제학, 통계학, 이미지 처리기법 등 다양한 분야에서 사용된다.

정의.

이차곡선의 방정식의 행렬표현

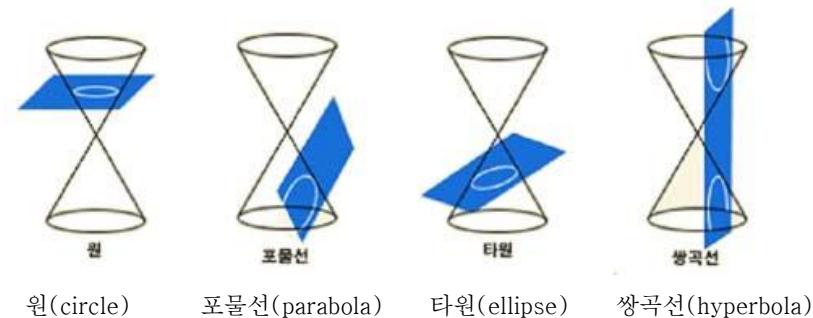
두 변수 x, y 를 갖는 이차곡선의 방정식

$$ax^2 + 2bxy + cy^2 + dx + ey + f = 0 \quad (1)$$

을 행렬로 표현하면 다음과 같다.

$$[x, y] \begin{bmatrix} a & b \\ b & c \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + [d, e] \begin{bmatrix} x \\ y \end{bmatrix} + f = 0 \quad (2)$$

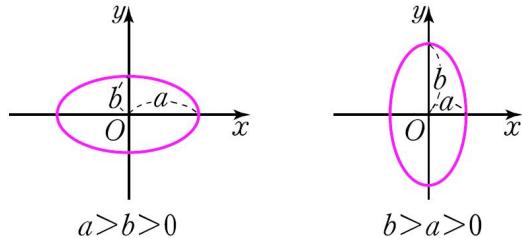
- 두 변수 x, y 를 갖는 이차방정식 (1)의 그래프는 기하학적으로 원뿔곡선(conic section)의 형태를 나타낸다. 원뿔곡선이란 용어는 평면이 원뿔과 교차될 때 생기는 곡선에서 유래된 말이다. 방정식 (1)을 만족하는 점 $(x, y) \in \mathbb{R}^2$ 가 없을 때는 식 (1)을 허 원뿔곡선(imaginary conic)의 방정식이라 한다. 또한, 방정식 (1)의 그래프가 한 점, 한 직선, 또는 한 쌍의 직선으로 이루어지거나 존재하지 않을 때, 이 그래프를 퇴화 원뿔곡선(degenerate conic)이라고 한다. 이 경우는 단순하므로 우리가 좀더 관심을 갖는 경우는 퇴화되지 않는 정상적인 경우이다. 정상적인 원뿔곡선(nondegenerate conic section)의 그래프는 타원, 쌍곡선 또는 포물선이 된다.



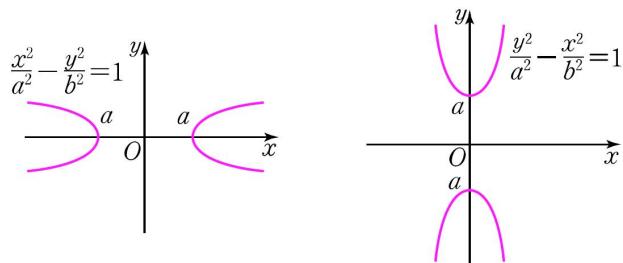
원뿔곡선의 방정식이 다음의 식 (3), (4), (5)로 표현되면 이 원뿔곡선은 표준위치에 있다고 한다.

■ 표준위치(standard position)에 있는 원뿔곡선

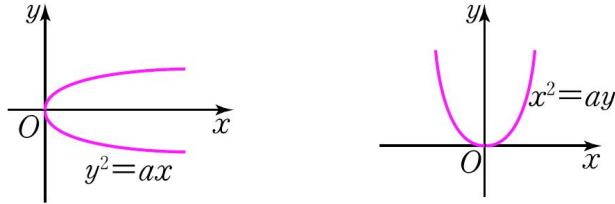
$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \quad (\text{타원}) \quad (3)$$



$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1 \quad \text{또는} \quad \frac{y^2}{a^2} - \frac{x^2}{b^2} = 1 \quad (\text{쌍곡선}) \quad (4)$$



$$y^2 = ax \quad \text{또는} \quad x^2 = ay \quad (\text{포물선}, \quad a > 0) \quad (5)$$



- 이차방정식에서 x^2 항과 x 항, y^2 항과 y 항을 갖는 이차방정식의 그래프는 표준위치로부터 평행 이동된 원뿔곡선이다.

예제 1 방정식 $3x^2 - 2y^2 - 18x + 4y + 19 = 0$ 은 완전제곱꼴(Completing the square)로 만들면

$$3(x-3)^2 - 2(y-1)^2 = 6 \quad (6)$$

이므로 $x' = x - 3$, $y' = y - 1$ 로 치환하면 새로운 $x'y'$ -좌표계에서 다음과 같이 나타난다.

$$\frac{(x')^2}{2} - \frac{(y')^2}{3} = 1$$

이 식은 $x'y'$ -좌표계에서 표준위치에 있는 쌍곡선의 방정식이다. 따라서 식 (6)의 그래프는 $x'y'$ -좌표계에서 표준위치에 있는 쌍곡선을 x -축으로 3만큼, y -축으로 1만큼 평행 이동한 그래프이다.

정의. \mathbb{R}^2 상의 이차형식(quadratic form)

$$[x \ y] \begin{bmatrix} a & b \\ b & c \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = ax^2 + 2bxy + cy^2 \quad (7)$$

부분을 \mathbb{R}^2 상의 방정식 (1) $ax^2 + 2bxy + cy^2 + dx + ey + f = 0$ 에 대한 이차형식 (quadratic form)이라 한다.

- 2차항들만으로 이루어진 $2x^2 + 6xy + y^2$, $x^2 + y^2$ 은 이차형식이고, $3x^2 - 6xy + y^2 - 3x + 1$ 은 $-3x$ 와 1이 2차항이 아니므로 이차형식이 아니다.

💡 일반적인 이차형식은 행렬을 도입하여 행렬 곱의 형태인 $\mathbf{x}^T A \mathbf{x}$ 꼴로 표현할 수 있다.

$$3x^2 + 7y^2 - 2xy = [x \ y] \begin{bmatrix} 3 & -1 \\ -1 & 7 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \text{ 또는 } 3x^2 + 7y^2 - 2xy = [x \ y] \begin{bmatrix} 3 & -2 \\ 0 & 7 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

■ 앞으로는 대칭행렬 A 를 얻기 위하여 b 를 둘로 나누어, 다음과 같이 나타내도록 한다.

$$ax^2 + by^2 + cxy = [x \ y] \begin{bmatrix} a & \frac{c}{2} \\ \frac{c}{2} & b \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix},$$

같은 방법으로 $ax^2 + by^2 + cz^2 + dxy + exz + fyz = [x \ y \ z] \begin{bmatrix} a & \frac{d}{2} & \frac{e}{2} \\ \frac{d}{2} & b & \frac{f}{2} \\ \frac{e}{2} & \frac{f}{2} & c \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

이와 같이 A 를 대칭행렬로 택하는 이유는 대칭행렬 A 는 항상 직교대각화 가능하기 때문이다.

정의. \mathbb{R}^n 상의 이차형식

$A = [a_{ij}]$ 가 n 차의 대칭행렬이고, n 개의 변수 x_1, x_2, \dots, x_n 을 성분으로 갖는 \mathbb{R}^n 의 벡터 $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$ 에 대하여 이차다항식 $q(\mathbf{x}) = \langle A\mathbf{x}, \mathbf{x} \rangle = \mathbf{x}^T A \mathbf{x} = \sum_{i,j=1}^n a_{ij} x_i x_j$ 을 \mathbb{R}^n 상의 이차형식이라 한다.

■ 위의 정의에서 A 를 n 차의 대칭행렬로 가정한 이유는 다음과 같다. 임의의 행렬 $A \in M_n$ 가 이차형식의 행렬곱 형태를 만드는 행렬이라고 하자. 이 행렬 A 는 대칭행렬 B 와 반대칭(skew symmetric) 행렬 C 의 합으로 표시된다 ($A = B + C = \frac{A+A^T}{2} + \frac{A-A^T}{2}$). 또한 $\mathbf{x}^T C \mathbf{x}$ 는 1×1 이고 $C^T = -C$ 므로

$$\mathbf{x}^T C \mathbf{x} = (\mathbf{x}^T C \mathbf{x})^T = \mathbf{x}^T C^T \mathbf{x} = -\mathbf{x}^T C \mathbf{x}$$

이다. 따라서 $2\mathbf{x}^T C \mathbf{x} = 0$ 이므로, 다음이 성립한다.

$$\begin{aligned} q(\mathbf{x}) &= (\langle A\mathbf{x}, \mathbf{x} \rangle) = \mathbf{x}^T A \mathbf{x} = \mathbf{x}^T (B + C) \mathbf{x} \\ &= \mathbf{x}^T B \mathbf{x} + \mathbf{x}^T C \mathbf{x} = \mathbf{x}^T B \mathbf{x} + 0 = \mathbf{x}^T B \mathbf{x} \end{aligned}$$

즉, 행렬 A 는 대칭행렬 B 와 반대칭(skew symmetric) 행렬 C 의 합으로 표시되고, $A = B + C$ 일 때 항상 $q(\mathbf{x}) = \mathbf{x}^T A \mathbf{x} = \mathbf{x}^T B \mathbf{x}$ 이므로, 이차형식의 값은 주어진 행렬의 대칭행렬 부분에만 의존하므로, 일반성을 잃지 않고 정의에서부터 A 를 대칭행렬이라고 한 것이다. 이차형식에 관한 연구의 많은 부분은 “대칭행렬 A 는 언제나 직교대각화가 가능하다”는 사실과 밀접한 관계가 있다.

💡 이차형식에서 xy 항을 교차항이라 한다. 대칭행렬의 직교대각화를 이용하면 교차항을 제거할 수 있다.

■ 이차형식 $q(\mathbf{x}) = \mathbf{x}^T A \mathbf{x} = ax^2 + 2bxy + cy^2$ 에서 행렬 $A = \begin{bmatrix} a & b \\ b & c \end{bmatrix}$ 가 대칭행렬이므로 고윳값 λ_1, λ_2 에 대응하는 A 의 정규직교인 고유벡터 $\mathbf{v}_1, \mathbf{v}_2$ 를 찾을 수 있고, $P = [\mathbf{v}_1 \ \mathbf{v}_2]$ 라 하면 A 는 P 에 의하여 직교대각화 가능하다. 즉, $P^T A P = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$ 이다. 이때, 고유벡터 \mathbf{v}_1 과 \mathbf{v}_2 는 λ_1 과 λ_2 의 역할을 바꾸어서 교환할 수 있으므로 일반성을 잃지 않고 $\det(P) = 1$ 이라 할 수 있다. 따라서 직교행렬 P 는 $\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$ 꼴의 \mathbb{R}^2 의 회전(rotation)행렬이다. 이러한 행렬 P 에 의하여 얻어진 새로운 좌표계를 $x'y'$ -좌표계라 하고 $\mathbf{x}' = \begin{bmatrix} x' \\ y' \end{bmatrix}$ 이라 하자. 그러면 $\mathbf{x} = P\mathbf{x}'$ 이고

$$\begin{aligned} q(\mathbf{x}) &= \mathbf{x}^T A \mathbf{x} = (P\mathbf{x}')^T A (P\mathbf{x}') = (\mathbf{x}')^T (P^T A P) \mathbf{x}' \\ &= [x' \ y'] \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} = \lambda_1 (x')^2 + \lambda_2 (y')^2 \end{aligned}$$

이므로 이차형식 q 는 새로운 좌표계에서는 교차항이 없이 표현된다. 따라서 다음 정리를 얻는다.

정리. \mathbb{R}^2 의 주축정리 (principal axis theorem)

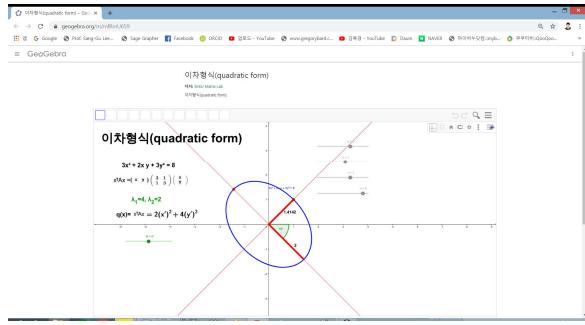
대칭행렬 $A = [a_{ij}]_{2 \times 2}$ 의 고윳값을 λ_1, λ_2 라 할 때, 좌표축의 회전에 의하여 이차형식 $q(\mathbf{x}) = \mathbf{x}^T A \mathbf{x}$ 는 새로운 $x'y'$ -좌표계에서

$$q(\mathbf{x}) = \lambda_1(x')^2 + \lambda_2(y')^2 \quad (8)$$

으로 표현될 수 있다. 이 회전은 (정규직교화법을 이용하여) A 를 대각화하는 (행렬식이 1인) 직교행렬을 P 를 선택하면, 새 $x'y'$ -좌표계는 $\mathbf{x} = P\mathbf{x}'$ 이라는 치환에 의하여 얻어진다.

 이차형식의 대각화를 이용한 3차원 곡면의 예.

<https://www.geogebra.org/m/mfRmU659>



■ 이차형식 (7)을

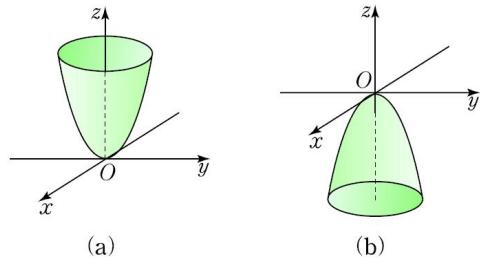
$$z = ax^2 + 2bxy + cy^2 \quad (13)$$

라 하고 이것을 대각화하면 회전된 $x'y'z$ -좌표계에서는

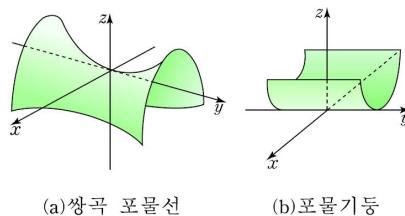
$$z = \lambda_1(x')^2 + \lambda_2(y')^2 \quad (14)$$

으로 변환되므로 \mathbb{R}^3 상에서 식 (13)의 그래프를 쉽게 알 수 있다.

식 (14)에서 (이차형식에 대응하는 대칭행렬 $A = [a_{ij}]_{2 \times 2}$ 의 고윳값인) λ_1, λ_2 가 모두 양이라면, 이 그래프는 아래 그림 (a)와 같이 위쪽이 열린 포물면(paraboloid)이다. 또한, λ_1, λ_2 가 모두 음이라면 그림 (b)와 같이 아래쪽이 열린 포물면이다. 이러한 포물면의 수평절단면은 타원(ellipse)이므로 타원포물면(elliptic paraboloid)이라고 한다.



- 또한 식 (14)에서 λ_1, λ_2 가 모두 영이 아니고 서로 다른 부호이면, 이 그래프는 아래 그림 (a)와 같이 안장 모양의 쌍곡포물면(hyperbolic paraboloid)이 된다. λ_1, λ_2 중 하나가 영이라면, 그래프는 그림 (b)와 같은 포물기둥(parabolic cylinder)이 된다.



- 이제 이차형식 그래프의 형태를 구분하는 이차형식의 부호를 정의하자. 이는 Part II에서 다면수함수의 극값을 구할 때 사용된다.

정의. 양의 정부호(positive definite), 음의 정부호, 부정부호(indefinite)

행렬 $A \in M_n$ 가 대칭행렬일 때, 이차형식 $q(\mathbf{x}) = \mathbf{x}^T A \mathbf{x}$ 가 임의의 $\mathbf{x} \neq 0$ 에 대하여 $q(\mathbf{x}) > 0$ 이면 양의 정부호(positive definite), $q(\mathbf{x}) < 0$ 이면 음의 정부호(negative definite)라 한다. 또한, 어떤 \mathbf{x} 에 대하여는 $q(\mathbf{x}) > 0$ 이고, 어떤 \mathbf{x} 에 대하여는 $q(\mathbf{x}) < 0$ 이면 부정부호(indefinite)라 한다.

정리. 이차형식의 정부호, 부정부호(indefinite)

행렬 $A \in M_n$ 가 대칭행렬일 때, A 의 이차형식 $q(\mathbf{x}) = \mathbf{x}^T A \mathbf{x}$ 는 다음을 만족한다.

- (1) A 의 고윳값들이 모두 양이라면 $q(\mathbf{x})$ 는 양의 정부호(positive definite)이다.
- (2) A 의 고윳값들이 모두 음이라면 $q(\mathbf{x})$ 는 음의 정부호(negative definite)이다.
- (3) A 가 양의 고윳값과 음의 고윳값을 모두 가지면 $q(\mathbf{x})$ 는 부정부호(indefinite)이다.

예제 2 다음 이차형식이 양의 정부호(positive definite)임을 보여라.

$$q(x, y, z) = 5x^2 + 6y^2 + 7z^2 + 4xy + 4yz$$

풀이. $q(x, y, z)$ 의 행렬 $A = \begin{bmatrix} 5 & 2 & 0 \\ 2 & 6 & 2 \\ 0 & 2 & 7 \end{bmatrix}$ 의 특성방정식은

$$|A - \lambda I| = (2 - \lambda)(6 - \lambda)(9 - \lambda) = 0$$

이므로 A 의 고윳값은 $\lambda_1 = 2, \lambda_2 = 6, \lambda_3 = 9$ 이다. 따라서 정리에 의해 이차형식 $q(x, y, z)$ 은 양의 정부호이다.

```
A = matrix([[5, 2, 0], [2, 6, 2], [0, 2, 7]])
print(A.eigenvalues())
```

[9, 6, 3]

예제 3 다음 이차형식이 부정부호(indefinite)임을 보여라.

$$q(x, y, z) = 3x^2 + 2xy + 2xz + 4yz$$

```
A = matrix([[3, 1, 1], [1, 0, 2], [1, 2, 0]])
print(A.eigenvalues())
```

[4, 1, -2]

따라서 이차형식 q 는 부정부호이다.

- 이차형식 $q(\mathbf{x})$ 의 행렬 A 의 각 고윳값이 $\lambda_i \geq 0$ 이면 양의 준정부호(positive semidefinite), $\lambda_i \leq 0$ 이면 음의 준정부호(negative semidefinite)라 한다. 예를 들면 이차형식 $g_4(x, y) = 2x^2$ 은 양의 준정부호이다.
- 이제, 행렬의 고윳값을 계산하지 않고도 두 변수를 갖는 정상적인 이차형식의 정부호를 결정하는 방법을 알아본다.

대칭행렬 $A = [a_{ij}] \in M_n$ 의 $k \times k$ ($k = 1, 2, \dots, n$) 소행렬식(minor)을 각각

$$\Delta_1 = a_{11}, \quad \Delta_2 = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}, \quad \Delta_3 = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}, \dots, \quad \Delta_n = |A|$$

이라할 때, $\Delta_1, \Delta_2, \dots, \Delta_n$ 을 A 의 선행 주 소행렬식(leading principal minors)이라고 한다. 아래는 주 부분행렬을 보여준다.

$$\begin{array}{cccc} \left[\begin{array}{cccc} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{array} \right] & \left[\begin{array}{cccc} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{array} \right] & \left[\begin{array}{cccc} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{array} \right] & \left[\begin{array}{cccc} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{array} \right] \\ \Delta_1 & \Delta_2 & \Delta_3 & \Delta_4 \end{array}$$

정리.

대칭행렬 $A \in M_n$ 의 이차형식 $q(\mathbf{x}) = \mathbf{x}^T A \mathbf{x}$ 에 대하여 다음이 성립한다.

- (1) q 가 양의 정부호(p.d.)일 필요충분조건은 모든 $k = 1, 2, \dots, n$ 에 대하여 $\Delta_k > 0$ 이다.
- (2) q 가 음의 정부호(n.d.)일 필요충분조건은 모든 $k = 1, 2, \dots, n$ 에 대하여 $(-1)^k \Delta_k > 0$ 이다.

예제 4 다음 이차형식이 양의 정부호(p.d.)임을 보여라.

$$q(x, y, z) = 2x^2 + 2y^2 + 9z^2 - 2xy - 6xz + 8yz$$

풀이]. 이차형식 q 의 행렬은 $A = \begin{bmatrix} 2 & -1 & -3 \\ -1 & 2 & 4 \\ -3 & 4 & 9 \end{bmatrix}$

이고, $\Delta_1 = 2, \Delta_2 = 3, \Delta_3 = 1$ 이므로 정리에 의하여 이차형식 q 는 양의 정부호(p.d.)이다.

```
A = matrix([[2, -1, -3], [-1, 2, 4], [-3, 4, 9]]) # 행렬 입력
m, n = A.nrows(), A.ncols() # 행렬의 크기
if m != n:
    raise ValueError("The matrix must be square.") # 정사각행렬이 아니면 에러

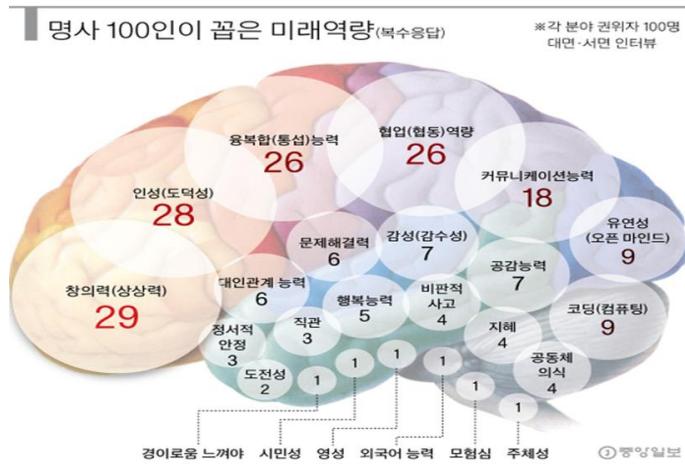
for i in range(1, n + 1):
    A_principal_minor = A.submatrix(0, 0, i, i).det() # 선행 주 소행렬식
    print("The ", i, "th principal minor is ", A_principal_minor)
```

The 1 th principal minor is 2
 The 2 th principal minor is 3
 The 3 th principal minor is 1 # 양의 정부호(p.d.)

위 내용은 Part II에서 다변수함수의 극값을 구할 때 사용된다. ■

Part 1 시험: <https://youtu.be/RwTb4ltUwA0>

[End of Part 1]



[선형대수학 지식을 더 알고 싶은 학생은 저자의 아래 동영상 강의를 참고하면 된다.]

Linear Algebra (선형대수학)

- 디지털 교과서: <http://matrix.skku.ac.kr/LA-K/>
[http://matrix.skku.ac.kr/2015-Album/Big-Book-LinearAlgebra-Eng-2015.pdf \(파일\)](http://matrix.skku.ac.kr/2015-Album/Big-Book-LinearAlgebra-Eng-2015.pdf)

- Video Lectures

Chapter 1. Vectors

- 1.1 벡터 and 1.2 내적 <http://youtu.be/aeLVQoPQMpE>
- 1.3 벡터방정식 <http://youtu.be/4UGACWYwOgA>

Chapter 2. Linear system of equations

- 2.1 선형연립방정식 <http://youtu.be/CiLn1F2pmvY>
- 2.2 Gauss-Jordan 소거법 <http://youtu.be/jnC66zvqHJI>

Chapter 3. Matrix and Matrix Algebra

- 3.1 행렬연산 <http://youtu.be/DmtMvQR7cwA>
- 3.2, 3.3 역행렬과 기본행렬 <http://youtu.be/GCKM2VIU7bw>
- 3.4 부분공간 http://youtu.be/HFq_-8B47xM
- 3.5 해공간 3.6 특수행렬 http://youtu.be/daIxHJBHL_g

Chapter 4. Determinant

- 4.1 행렬식 <http://youtu.be/DM-q2ZuQtI0>
- 4.2 여인자 전개와 역행렬 <http://youtu.be/XPCD0ZYoH5I>
- 4.3 크래머의 법칙 4.4. Appl, 4.5 고유값, 고유벡터
<http://youtu.be/OImrmmWXuvU>

Chapter 6. Linear Transformations

- 6.1 선형변환 <http://youtu.be/YF6-ENHfI6E>
- 6.2 선형변환의 기하학적 의미 <http://youtu.be/cgySDj-OVlM>
- 6.3 핵과 치역 <http://youtu.be/9YciT9Bb2B0>
- 6.4 선형변환의 합성과 역행렬 <http://youtu.be/EOlq4LouGao>

Chapter 7. Dimension and Subspaces

- 7.1 기저와 차원 <http://youtu.be/or9c97J3Uk0>
- 7.2 주요 부분공간들 <https://youtu.be/BC9qeR0JWIs>
- 7.3 Rank Nullity Theorem http://youtu.be/ez7_JYRGsb4
- 7.4 계수정리 <http://youtu.be/P4cmhZ3X7LY>
- 7.5 정사영정리 <http://youtu.be/GlcA4l8SmlM>
- 7.6* 최소제곱해 <https://youtu.be/BC9qeR0JWIs>
- 7.7 Gram-Schmidt의 정규직교화과정 <http://youtu.be/gt4-EuXvx1Y>
- 7.8* QR-분해, Householder transformations <https://youtu.be/crMXPi2lgGs>
- 7.9 좌표벡터 <http://youtu.be/M4peLF7Xur0>

Chapter 8. Diagonalization

- 8.1 선형변환의 행렬표현 <http://youtu.be/gn5ve1tXD7k>, <http://youtu.be/jfMcPoso6g4>
- 8.2 닮음과 행렬의 대각화 <http://youtu.be/xirjNZ40kRk>
- 8.3 직교대각화 <http://youtu.be/jimlkBGAZfQ>
- 8.4 이차형식 <http://youtu.be/vWzHWEhAd-k>
- 8.5* Appl of Quadratic Function <http://youtu.be/cOW9qT64e0g>
- 8.6 Singular Value Decomposition <https://youtu.be/ejCge6Zjf1M>
- 8.7 and 8.8 복소고유값, 복소고유벡터, 정규행렬 http://youtu.be/8_uNVj_OIAk

Chapter 9. General Vector Spaces

- 9.1 and 9-2 일반벡터공간, 내적공간 <http://youtu.be/m9ru-F7EvNg>
- 9.3 동형사상 <http://youtu.be/frOcceYb2fc>

Chapter 10. Jordan Canonical Form

- 10.1 Jordan 표준형 <http://youtu.be/NBLZPcWRHYI>
- 10.3 Jordan Canonical Form with Sage <http://youtu.be/LxY6RcNTEE0>

○ All Solutions

<http://matrix.skku.ac.kr/2010-Album/2010-MT-all-Solution-v1-sglee/2010-MT-all-Solution-v1-sglee.html>

○ Midterm Exam <http://youtu.be/R3F3VNGH8Oo>

○ K-MOOC Lab <http://matrix.skku.ac.kr/K-MOOC-LA/>

[출처] <http://matrix.skku.ac.kr/2019-album/>

[명령어 모음] <http://matrix.skku.ac.kr/Lab-Book/Sage-Lab-Manual-1.htm>

[강의동영상] <http://matrix.skku.ac.kr/2019-album/>

[교재] <http://matrix.skku.ac.kr/Cal-Book1/>

[연습문제] <http://matrix.skku.ac.kr/Cal-Book/>

1. 일변수함수와 미적분 <http://matrix.skku.ac.kr/PBL/>

2. 다변수함수와 미적분 <http://matrix.skku.ac.kr/PBL2/>



[부록 1] 미적분학의 상호연관성(Calculus Map)

[부록 2] 미적분학용 Sage/Python 코드

[부록 3] 미적분학 공식과 표(Table)

1. 일변수함수와 미적분

10강 동영상, 극한과 도함수 <https://youtu.be/rsltpfMbtBQ> (25:55)

1.1 함수

참고 동영상: <http://youtu.be/cl8GqIWIRD0>

실습 사이트: <http://matrix.skku.ac.kr/Cal-Book1/Ch1/>

<http://matrix.skku.ac.kr/Cal-Book/part1/CS-Sec-1-3.htm> (그래프 그리기 기본, 크롬)

예제 1 함수 $y = \frac{x^3}{8}$ 와 $y = -\frac{1}{x}$ 의 그래프를 그려라.

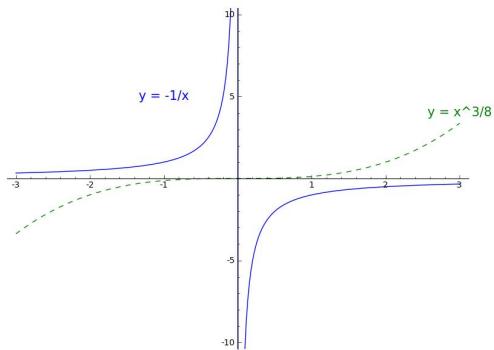
[실습: <http://sage.skku.edu/> 또는 <https://sagecell.sagemath.org/>]

```
var('x') # Define variable "x"
A = plot(x^3/8, x, -3, 3, linestyle = "--", color = 'green')
```

```

# graph the first function
A = plot(-1/x, x, -3, 3) # graph the second function
B = plot(-1/x, x, -3, 3)
C = text("y = x^3/8 ", (3, 4), color = 'green', fontsize = 15)
D = text("y = -1/x", (-1, 5), fontsize = 15 )
# add the text on given position
# and set the font of the text
show(A + B + C + D, ymax = 10, ymin = -10) # display graphs and texts one time

```

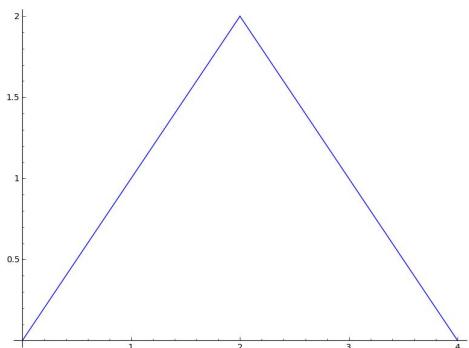


예제 2 함수 $f(x) = \begin{cases} x, & 0 \leq x \leq 2 \\ 4-x, & 2 < x \leq 4 \end{cases}$ 의 그래프를 그려라.

```

var('x')
f1(x) = x
f2(x) = 4 - x
# graph f1 on (0, 2), and f2 on (2, 4)
plot(piecewise([(0, 2), f1], ((2, 4), f2))), (x, 0, 4))

```



1.2 극한 (limit)

참고 동영상: <http://youtu.be/VBCeAllP1M0>

실습 사이트: <http://matrix.skku.ac.kr/Cal-Book1/Ch2/>

<http://matrix.skku.ac.kr/Cal-Book/part1/CS-Sec-2-1-Sol.html>

정의. 수렴, $f(x)$ 의 극한(limit)

임의의 양수(positive) ε 에 대하여, 만일 $0 < |x - a| < \delta$ 이면 $|f(x) - b| < \varepsilon$ 되게 하는 적당한 양수(positive) δ 가 존재하면, x 가 a 에 접근할 때 $f(x)$ 는 b 에 수렴(converge)한다고 하고, b 를 (x 가 a 에 접근할 때) $f(x)$ 의 극한(limit)이라고 부르며, $\lim_{x \rightarrow a} f(x) = b$ 라고 쓴다. 수렴하지 않으면 발산(diverse)한다고 한다.

예) <http://matrix.skku.ac.kr/Cal-Book/part1/CS-Sec-2-1-Sol.html>]

* 위의 [극한 정의, $\lim_{x \rightarrow a} f(x) = b$]에 대한 기호 표기는 다음과 같다.

$\forall \epsilon \exists \delta > 0 \text{ such that } \forall x \in S, 0 < |x - a| < \delta \text{ implies } |f(x) - L| < \epsilon$

* (x 가 a 에 접근할 때) 좌극한과 우극한이 존재하고 $f(a)$ 가 존재하여 이 셋의 값이 모두 같으면, 함수 $f(x)$ 는 a 에서 연속(continuous)이라고 한다. 앞에서 언급한 엡실론-델타(Epsilon-Delta) 개념은 생물학에서 혈미경과 같은 기능을, 수학 특히 함수에게 주어, 극한에 대한 애매함을 대부분 해결해 주면서 19세기 현대수학 발전을 획기적으로 리드하였다.

<http://matrix.skku.ac.kr/cal-lab/SKKU-Cell-Epsilon-Delta.html>

예제 3 다음 극한(limit)을 구하여라.

$$\lim_{x \rightarrow 2} \frac{\sin x}{x+1} \text{ (극한)} \quad \lim_{x \rightarrow 2^-} \frac{x^2 - 4}{|x - 2|} \text{ (좌극한)}, \quad \lim_{x \rightarrow 2^+} \frac{x^2 - 4}{|x - 2|} \text{ (우극한)}$$

[실습: <http://sage.skku.edu/> 또는 <https://sagecell.sagemath.org/>]

```
var( 'x' )
g(x) = sin(x)/(x +1)
print(limit(g(x), x = 2))          # x = 2일 때 극한
f(x) = (x^2 - 4)/abs(x - 2)
print(limit(f(x), x = 2, dir = '-')) # x = 2일 때 좌극한
print(limit(f(x), x = 2, dir = '+')) # x = 2일 때 우극한
```

$$\frac{1}{3}\sin(2) \quad \# x = 2\text{일 때 극한} \quad \frac{\sin 2}{2+1} = \frac{1}{3} \sin(2)$$

■

예제 4 극한 $\lim_{x \rightarrow \frac{\pi}{2}} \frac{\sin x}{|\cos x|}$ 와 $\lim_{x \rightarrow \frac{\pi}{4}} \frac{x^2}{\sin x}$ 을 구하여라.

```
var( 'x, y' )
f(x) = sin(x)/abs(cos(x))
g(y) = x^2/sin(y)
A = limit(f(x), x = pi/2) # x = pi/2일 때 극한
B = limit(g(y), y = pi/4) # x = pi/4일 때 극한
print(A)
print()
print(B)
```

$$+\infty \quad \# \text{극한이 존재하지 않는다. 발산(diverse)한다.}$$

■

$$\sqrt{2}x^2 \quad \# \text{극한이 존재한다. 수렴(converge)한다.}$$

1.3 도함수(derivative)와 미분(differentiation)

참고 동영상: <http://youtu.be/A-vDsF9u1Ts> (redo)

실습 사이트: <http://matrix.skku.ac.kr/Cal-Book1/Ch3/>

<http://matrix.skku.ac.kr/Cal-Book/part1/CS-Sec-3-1-Sol.html>

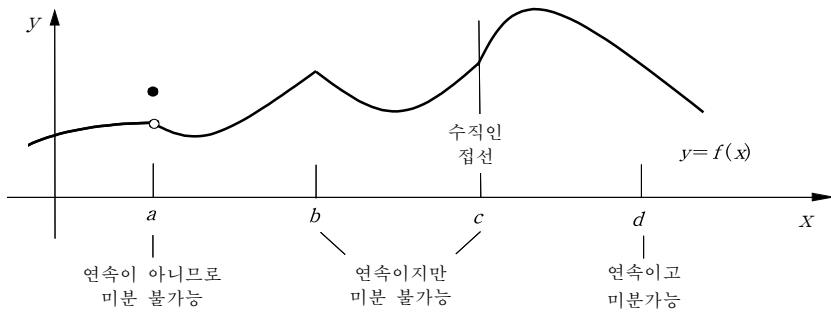
정의.

미분가능(differentiable)

함수 $f(x)$ 가 a 를 포함하는 어떤 구간에서 정의(well defined)되어 있고, 극한값

$$\lim_{h \rightarrow 0} \frac{f(a+h) - f(a)}{h} \text{ 이 } \begin{array}{l} \text{존재하면 } \\ \text{f(x) 는 } \end{array} \begin{array}{l} x=a \text{ 에서 } \\ \text{미분가능} \end{array}$$

(differentiable)이라고 하며 이 극한값을 $f(x)$ 의 a 에서의 미분계수(differential coefficient)라 부르고 $f'(a)$ 로 나타낸다. $f(x)$ 는 $x=a$ 에서 미분가능이면 $x=a$ 에서 연속이다 (그러나 역은 성립하지 않는다). 즉 $f(x)$ 가 $x=a$ 에서 연속이라도 반드시 미분가능인 것은 아니다. 다음 그림을 참조하자.



$y=f(x)$ 가 어떤 구간의 각 점 x 에서 미분가능일 때 $f(x)$ 는 이 구간에서 미분가능이라고 한다. 이 경우 각 점 x 에 그 점에서의 미분계수를 대응시킴으로써 정해지는 함수를 $f(x)$ 의 도함수(derivative)라 하고 다음 기호들로 나타낸다.

$$f'(x), y', \frac{dy}{dx}, \frac{d}{dx} f(x), Dy$$

함수 $f(x)$ 의 도함수를 구하는 것을 $f(x)$ 를 미분(differentiate)한다고 하고, 함수 $y=f(x)$ 의 도함수는 식 $f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h)-f(x)}{h}$ 으로 구한다. 따라서 어떤 함수를 미분하면 그 함수가 도함수이고, 거기에 상수를 대입하면 그 점에서의 미분계수가 나오는 것이다. 개인과 전체의 개념으로 설명하면, 한 개인의 기울기를 구하는 것이 미분계수이고, 그런 개인이 많아져서 전체가 되면 도함수라고 생각할 수도 있다.

$y = f(x)$ 의 도함수 $y' = f'(x)$ 가 다시 미분가능이면 그 도함수 (y')'을 생각할 수 있다. 이것을 $y = f(x)$ 의 제 2계도함수(2nd derivative)라 하고, y'' , $f''(x)$, $\frac{d^2y}{dx^2}$ 등으로 나타낸다. 이 제2계도함수가 또 다시 미분가능이면 제 3계 도함수(3rd derivative)를 생각할 수 있게 된다. 이와 같이 $y = f(x)$ 를 계속하여 n 번 미분하면 제 n 계도함수가 정의된다. 제 n 계도함수(n th derivative)는 다음 기호로 나타낸다.

$$y^{(n)}, f^{(n)}(x), \frac{d^n y}{dx^n}$$

$f^{(n)}(x)$ 가 존재하는 경우 $f(x)$ 는 n 번 미분가능이라고 한다. 미분계수의 응용인 속도와 가속도는 각각 1계도함수와 2계도함수의 예이다.

예제 5 $y = x^2 + 4e^x$ 의 도함수(derivative) $\frac{dy}{dx}$ 와 2계도함수 $\frac{d^2y}{dx^2}$ 와 3계도함수를 구 하여라.

실습: <http://sage.skku.edu/> 또는 <https://sagecell.sagemath.org/>

```
var('x')
f(x) = x^2 + 4*e^x
print(diff(f(x), x))    # dy/dx
print(diff (diff(f(x), x), x))  # 2계도함수
print(diff (diff (diff(f(x), x), x), x)) # 3계도함수
```

```
2*x + 4*e^x
2 + 4*e^x          # 2계도함수
4*e^x              # 3계도함수
```



$f(x)$ 가 $x=a$ 에서 미분가능인 것은 곡선

$y=f(x)$ 상의 점 $A(a, f(a))$ 에서 이 곡선에 대한

접선(tangent line)을 그을 수 있음을 의미한다. 이때

$(a, f(a))$ 에서의 기울기는 $f'(a) = \frac{y-f(a)}{x-a}$ 이므로

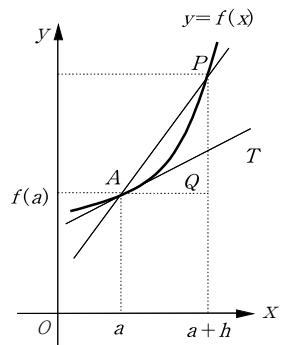
접선의 방정식은 다음과 같이 구한다.

$$y - f(a) = f'(a)(x - a) \quad \text{또는} \quad y = f(a) + f'(a)(x - a)$$

그리고 곡선 $y=f(x)$ 위의 점 $A(a, f(a))$ 에 있어

서 이 곡선의 접선에 수직인 직선을 A 에 있어서의 법선(normal line)이라 한다.

(이 법선(normal line)의 기울기를 m 이라고 하면 $f'(a)m = -1$ 이다.)

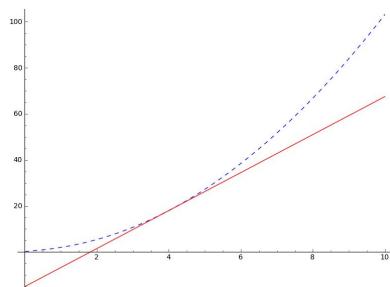


예제 6 점 $(4, 18)$ 에서 곡선 $y=x^2 + \sqrt{x}$ 에 대한 접선의 방정식을 구하여라.

[실습: <http://sage.skku.edu/> 또는 <https://sagecell.sagemath.org/>]

```
var('x') # [실습: http://sage.skku.edu/ 또는 https://sagecell.sagemath.org/]
f(x) = x^2 + sqrt(x)
df(x) = diff(f(x), x) # 도함수 구하기
y(x) = df(4)*(x - 4) + 18 # 기울기(df(4))를 구하여 접선의 방정식 구하기
print(y(x))
p1 = plot(f(x), x, 0, 10, linestyle = "--", color = 'blue') # 함수 f(x) 그리기
p2 = plot(y(x), x, 0, 10, color = 'red') # 접선 y(x) 그리기
show(p1 + p2) # 함수 f(x)와 접선 y(x)를 동시에 보여주기
```

$$33/4x - 15 \quad \# \text{접선의 방정식} \quad y = f(a) + f'(a)(x-a) = -15 + \frac{33}{4}x \blacksquare$$



$$\text{접선 방정식 } y = f(a) + f'(a)(x-a) = -15 + \frac{33}{4}x \blacksquare$$

1.4 미분의 응용 (Applications)

참고 동영상: <http://youtu.be/mXVU8OqIHJY>

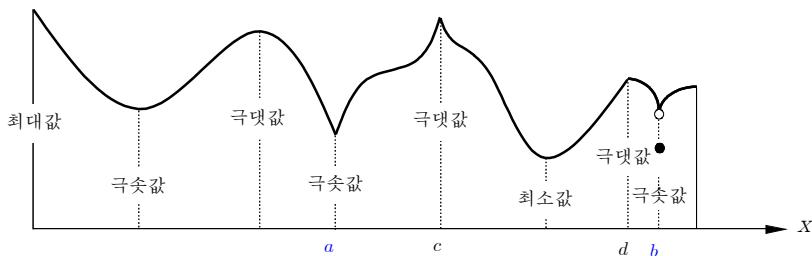
실습 사이트: <http://matrix.skku.ac.kr/Cal-Book1/Ch4/>

<http://matrix.skku.ac.kr/Cal-Book/part1/CS-Sec-4-1-Sol.html>

11강 동영상, 미분의 응용 <https://youtu.be/O4lN5zEZnMA> (13:58)

정리. 최댓값 최솟값의 정리

$f(x)$ 가 폐구간 $[a, b]$ 에서 연속이면 이 구간에서 $f(x)$ 가 최댓값을 취하는 점 및 최솟값을 취하는 점이 존재한다.



따라서 아래 예제와 같이 그래프를 그리고, 1차도함수의 방정식의 근인 극값에서의 함수 값과 구간의 양 끝점에서의 함수값을 비교하여, 최댓값과 최솟값을 쉽게 구할 수 있다.

예제 7 $[-3, 3]$ 에서 연속함수 $f(x) = x^3 - x$ 의 최댓값과 최솟값을 구하여라.

[실습: <http://sage.skku.edu/> 또는 <https://sagecell.sagemath.org/>]

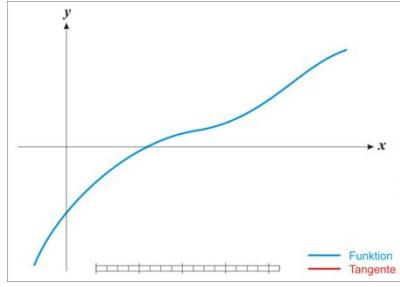
```
var('x')          # [실습: http://sage.skku.edu/ 또는 https://sagecell.sagemath.org/]
f(x) = x^3 - x
p1=plot (f(x), x, -4, 4)
show (p1)
solve(diff(f(x)) == 0, x)    # look for the flection points of df, -1/3*sqrt(3),
1/3*sqrt(3)
ma = max(f(-1/3*sqrt(3)), f(1/3*sqrt(3)), f(-3), f(3))    # find maximum value
mi = min(f(-1/3*sqrt(3)), f(1/3*sqrt(3)), f(-3), f(3))    # find minimum value
print('최댓값 =', ma)
print('최솟값 =', mi)
```

최댓값 = 24 # $[-3, 3]$ 에서 연속함수 $f(x) = x^3 - x$ 의 최댓값, 24

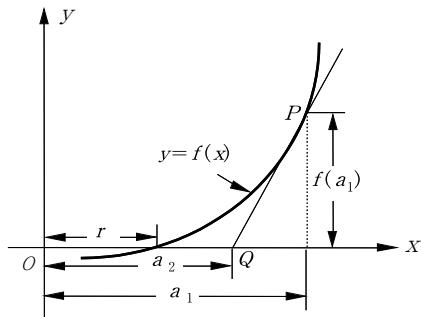
최솟값 = -24 # $[-3, 3]$ 에서 연속함수 $f(x) = x^3 - x$ 의 최솟값, -24 ■

- Newton's Method (뉴턴 방법) : 실수값 함수의 근(근사값)을 그래프와 도함수를 이용하여 쉽게 구하는 방법. <https://darkpgmr.tistory.com/58>
<https://www.quora.com/In-which-cases-is-Newtons-method-not-applicable>

주어진 함수의 그래프를 그린 후, Newton의 방법에 의하면 그 함수가 x 축과 만나는 근(무리수 값의 근)의 근사값을 소숫점 이하 몇 번째 자리까지든지 원하는 대로 정확하게 구할 수 있다.



[wiki 크리에이티브 커먼즈 그림]



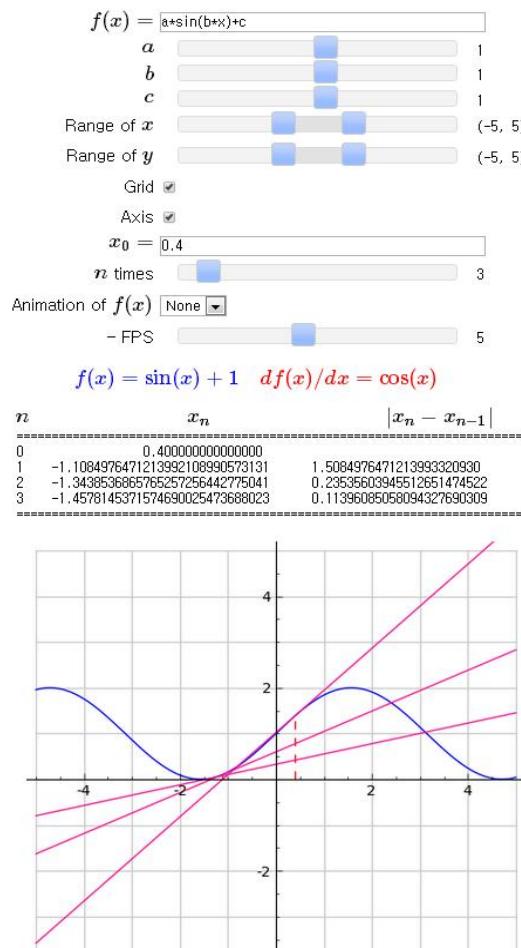
최적화문제를 푸는 계산방법은 대개 반복법(iterative method)으로, 초기 근사해로부터 시작하여 특정한 반복단계를 거쳐 이전보다 나은 근사해들을 생성한다. 위의 그림에서 곡선 $y = f(x)$ 를 생각하자. $x = a_1$ 을 방정식 $f(x) = 0$ 의 근 r 에 대한 첫 번째 근사값이라 하자. 그럼 $y - f(a_1) = f'(a_1)(x - a_1)$ 은 점 P 에서의 접선 $y = f(x)$ 에 대한 접선의 방정식이다. 그래프를 보고 a_1 을 r 에 적당히 가까운 값으로 선택하면, 이 접선은 보통 x 축과 점 Q 에서 만나는데, 그 점의 x 좌표 a_2 는 a_1 보다 r 에 더 가까운 근사값이 된다. 따라서 위식에 $x = a_2$, $y = 0$ 를 대입하면, $a_2 = a_1 - \frac{f(a_1)}{f'(a_1)}$ 을 얻는다. 다시 a_2 를 첫 번째 근사값으로 생각하고 다음 접선을 이용하면 $a_3 = a_2 - \frac{f(a_2)}{f'(a_2)}$ 를 얻는다. 이 과정을 반복하면 공식 $a_{n+1} = a_n - \frac{f(a_n)}{f'(a_n)}$, $n = 1, 2, 3, \dots$ 을 얻게 된다. ⓠ Newton의 공식을 반복하여 사용하면, 원하는 근을 소숫점 이하 몇 번째 자리까지라도 필요한 오차 이내로 정확하게 구할 수 있다.

『주의』 첫 번째 근사값 a_1 을 찾으려는 근 r 에 충분히 가깝게 택하지 못하면

Newton의 공식은 불합리한 결과를 초래할 수도 있다. 이를 보완하는 한 방법은 근에 상당히 가까운 근사값에 이를 때까지는 그래프를 먼저 그리거나, 이등분법(bisection method)을 사용하여, 근에 충분히 가까운 점에 온 후 Newton의 방법으로 전환하는 것이다.

이 과정을 아래 도구를 이용하여 실습하도록 하자.

http://matrix.skku.ac.kr/Mobile-Sage-G/sage-grapher-newton_method.html



미적분학 스토리텔링 <http://matrix.skku.ac.kr/Calculus-Story/index.htm> ■

1.5 적분 (Integral)

참고 동영상: <http://youtu.be/GIm3Oz58Ti8>

실습 사이트: <http://matrix.skku.ac.kr/Cal-Book1/Ch5/>

<http://matrix.skku.ac.kr/Cal-Book/part1/CS-Sec-5-2-Sol.html>

12강 동영상, 적분 <https://youtu.be/62OxYG7VMsE> (8:41)

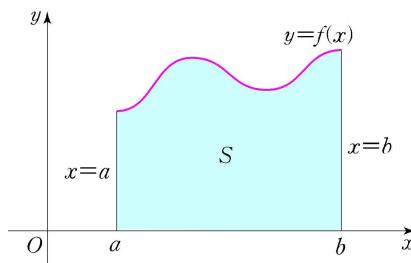
어떤 구간에서 정의된 함수 $f(x)$ 에 대하여 이 구간의 모든 x 에 관하여 $F'(x) = f(x)$ 를 만족하는 함수 $F(x)$ 가 존재할 때 $F(x)$ 를 $f(x)$ 의 원시함수 또는 부정적분(indefinite integral)이라고 한다. $f(x)$ 가 주어졌을 때 그 부정적분 $F(x)$ 를 구하는 것을 $f(x)$ 를 적분한다고 한다.

$f(x)$ 의 부정적분을 $\int f(x)dx$ 로 나타내고 \int 를 적분기호, $f(x)$ 를 피적분함수, dx 를 적분변수라고 한다. $F(x)$ 가 $f(x)$ 의 한 부정적분이면

$$\int f(x)dx = F(x) + C \quad (C \text{는 임의의 상수}) \quad (3-2)$$

이다. 식 (3-2)의 우변의 C 를 적분상수라고 한다.

[a, b]에서 연속인 함수 $f(x)$ 는 [a, b]에서 적분가능이라고 하며, $f(x)$ 의 a 로부터 b 까지의 정적분(definite integral)을 $S = \int_a^b f(x) dx$ 로 정의한다.

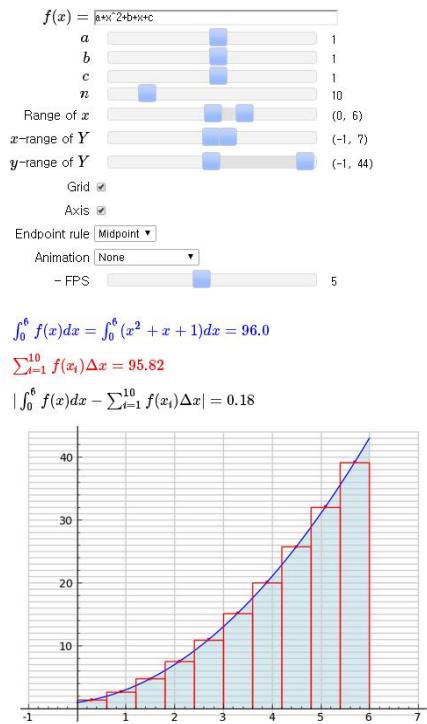


$$S = \{(x, y) | a \leq x \leq b, 0 \leq y \leq f(x)\}$$

<http://matrix.skku.ac.kr/cal-lab/Area-Sum.html>

■ 면적(Area), 리만 합(Riemann sum)과 적분값(Integral)

http://matrix.skku.ac.kr/Mobile-Sage-G/sage-grapher-riemann_sum.html (실습)

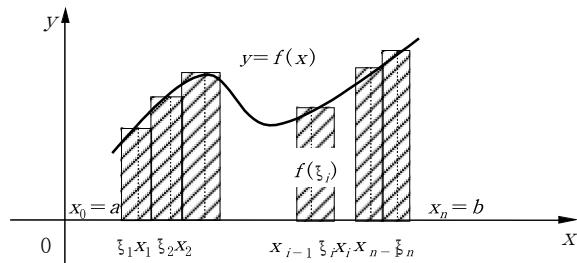


정리. 적분에 관한 평균값의 정리

$f(x)$ 가 $[a, b]$ 에서 연속이면

$$\int_a^b f(x)dx = (b-a)f(\xi)$$

인 ξ 가 a 와 b 사이에 적어도 하나 존재한다.



[미적분학의 기본정리]는 정적분과 부정적분(도함수), 미분과 적분 사이의 관계를 잘 보여준다.

정리. 미적분학의 기본정리 (적분과 미분의 연결고리)

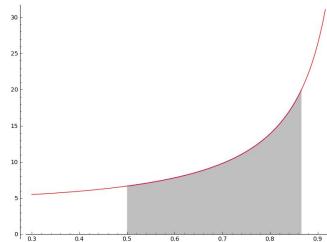
$f(x)$ 가 $[a, b]$ 에서 연속이고 $F(x)$ 를 $f(x)$ 의 임의의 한 부정적분, $F'(x) = f(x)$ 이라 하면 다음식이 성립한다.

$$\int_a^b f(x) dx = F(b) - F(a)$$

예제 8 적분 $\int_{1/2}^{\sqrt{3}/2} \frac{5}{1-x^2} dx$ 을 구하여라.

[실습: <http://sage.skku.edu/> 또는 <https://sagecell.sagemath.org/>]

```
f(x) = 5/(1 - x^2)          # [실습: http://sage.skku.edu/ 또는 https://sagecell.sagemath.org/]
P1 = plot(f(x), 0.5, sqrt(3)/2, fill = "axis")  # fill the area between f(x) and x-axis
P2 = plot(f(x), 0.3, sqrt(3)/2 + 0.05, color = 'red')  # graph f(x)
show(P1 + P2)  # show P1 and P2
print(integral(f(x), x, 1/2, sqrt(3)/2).simplify_full())  # integrate f(x) on  $\left(\frac{1}{2}, \frac{\sqrt{3}}{2}\right)$ 
print((integral(f(x), x, 1/2, sqrt(3)/2).simplify_full()).n(digits=5))
```



$-5/2\log(-\sqrt{3} + 2) + 5/2\log(\sqrt{3} + 2) - 5/2\log(3)$

■



[미적분학 Big Picture] <http://matrix.skku.ac.kr/Calculus-Map/>

2. 다변수함수와 미적분

13장 동영상, 다변수함수 https://youtu.be/XQW8_8k9GjE (10:12)

2.1 벡터와 공간기하

참고 동영상: http://youtu.be/_s_2T1VVob8 <http://youtu.be/BFgh6irMqsc>

http://youtu.be/lxuGE_Erthg

실습 사이트: <http://matrix.skku.ac.kr/Cal-Book1/Ch11/>

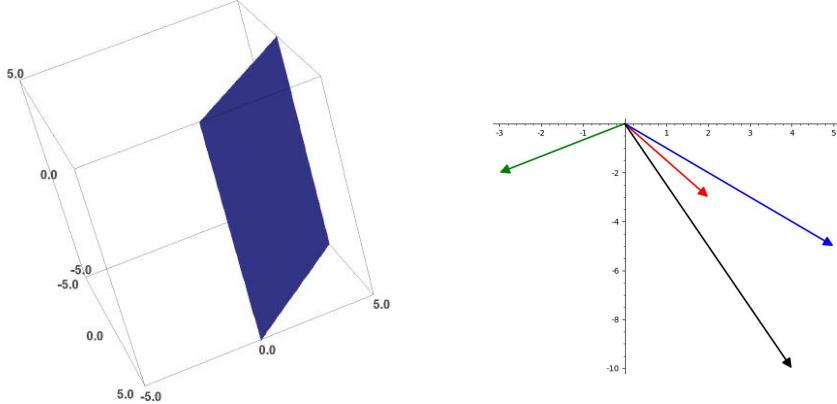
<http://matrix.skku.ac.kr/Cal-Book/part2/CS-Sec-11-5-Sol.html>

<http://matrix.skku.ac.kr/Cal-Book/part2/CS-Sec-11-2-Sol.html>

예제 1 \mathbb{R}^3 상에 곡면(평면) $x+y=5$ 을 그려라. (아래 좌측 그림)

[3차원 그래프 실습: <https://sagecell.sagemath.org/>]

```
var('x, y, z')
implicit_plot3d(x + y == 5, (x, -5, 5), (y, -5, 5), (z, -5, 5)) # graph x+y=5
on in  $\mathbb{R}^3$ 
```



예제 2 벡터 $\mathbf{a} = \langle 2, -3 \rangle$, $\mathbf{b} = \langle -3, -2 \rangle$, $\mathbf{c} = \langle 5, -5 \rangle$ 를 평면에 그리고 그 합을 구하여라.

```

a = vector([2, -3])    # define a vector a
b = vector([-3, -2])   # define a vector b
c = vector([5, -5])    # define a vector c
print(a + b + c)      # find the vector sum of a, b, c
d=a + b + c
plot(a, color= 'red' ) + plot(b, color = 'green') + plot(c, color = 'blue') +
plot(d, color= 'black' )  # graph 4 vectors

```

(4, -10) # 3 벡터의 합 $d = a + b + c = (4, -10)$ (위 우측 그림) ■

■ 외적(Cross Product) <http://matrix.skku.ac.kr/cal-lab/cal-11-4-Exs-6.html>

$$\mathbf{a} \times \mathbf{b} = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{vmatrix} = \begin{vmatrix} a_2 & a_3 \\ b_2 & b_3 \end{vmatrix} \mathbf{i} - \begin{vmatrix} a_1 & a_3 \\ b_1 & b_3 \end{vmatrix} \mathbf{j} + \begin{vmatrix} a_1 & a_2 \\ b_1 & b_2 \end{vmatrix} \mathbf{k}$$

■ [C-S 부등식!! $|\mathbf{x} \cdot \mathbf{y}| \leq \|\mathbf{x}\| \|\mathbf{y}\|$] 벡터 \mathbf{x}, \mathbf{y} 에 대하여

$$\mathbf{x} \cdot \mathbf{y} = \|\mathbf{x}\| \|\mathbf{y}\| \cos \theta, \quad 0 \leq \theta \leq \pi \quad (-1 \leq \frac{\|\mathbf{x}\| \|\mathbf{y}\|}{\mathbf{x} \cdot \mathbf{y}} \leq 1)$$

인 θ 가 존재하며, 이 θ 를 \mathbf{x} 와 \mathbf{y} 가 이루는 각(angle, 사잇각)이라 한다.

예제 3 벡터 $\mathbf{a} = \langle 6, -3, 4 \rangle$ 와 $\mathbf{b} = \langle 2, 0, -3 \rangle$ 의 내적, 외적, 사잇각을 구하여라.

```

a = vector([6, -3, 4])    # https://sagecell.sagemath.org/
b = vector([2, 0, -3])
s = b.dot_product(a)      # find the dot product of a and b
c = b.cross_product(a)   # find the cross product of a and b
theta = arccos(a.dot_product(b)/(a.norm()*b.norm()))
# find the angle between a and b
print('내적(inner product, dot product)은 ', s)
print('외적(cross product)은', c)
print('벡터 a와 b 의 사잇각(angle)은', theta)

```

내적(inner product, dot product)은 0
외적(cross product)은 $(-9, -26, -6)$
벡터 a 와 b 의 사잇각(angle)은 $1/2\pi$



2.2 벡터 함수

참고 동영상: <http://youtu.be/0pvywjBjsQw>

실습 사이트: <http://matrix.skku.ac.kr/Cal-Book1/Ch12/>

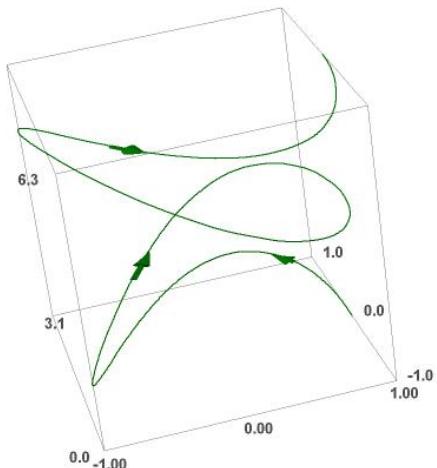
<http://matrix.skku.ac.kr/Cal-Book/part2/CS-Sec-12-1-Sol.html>

<http://matrix.skku.ac.kr/Cal-Book/part2/CS-Sec-12-2-Sol.html>

예제 4 벡터 방정식 $\mathbf{r}(t) = \langle \cos 2t, \sin 3t, t \rangle$ 으로 주어지는 곡선을 그리고 t 가 증가하는 방향을 화살표로 나타내어라.

[실습: <https://sagecell.sagemath.org/>]

```
var('t')                               # https://sagecell.sagemath.org/
r = vector([cos(2*t), sin(3*t), t])   # Define r(t)
C = parametric_plot3d(r, (t, 0, 2*pi), color = 'green', thickness = 2)      #
Sketch curve
t1 = pi/10    # location of arrows
t2 = 2*pi/3
t3 = 5*pi/3
Ar1 = arrow3d(r(t = t1), r(t = t1+0.1), color = 'green')    # Draw arrows
Ar2 = arrow3d(r(t = t2), r(t = t2+0.1), color = 'green')
Ar3 = arrow3d(r(t = t3), r(t = t3+0.1), color = 'green')
C + Ar1 + Ar2 + Ar3
```



예제 5 벡터함수 $\mathbf{r}(t) = \langle \cos t, t \rangle$ 에 대하여 $\mathbf{r}'(t)$ 을 구하고, $t = \frac{\pi}{4}$ 일 때의 위치벡터 $\mathbf{r}(t)$ 와 접선벡터 $\mathbf{r}'(t)$ 를 그려라.

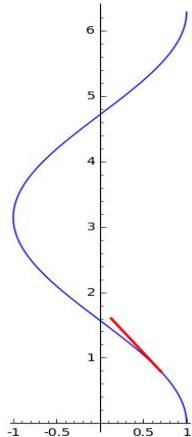
```

var('t')
x(t) = cos(t)
y(t) = t
r(t) = (x(t), y(t))
dr(t) = (diff(x(t), t), diff(y(t), t))
print("r'(t) =", dr(t))    # r'(t)
t0 = pi/4
p1 = r(t0)    # Find the startpoint of tangent vector at t = π/4
p2 = dr(t0)/abs(dr(t0)) + r(t0)    # Find the endpoint of tangent vector at
t = π/4
p = parametric_plot((x(t), y(t)), (t, 0, 2*pi))    # Sketch curve
utv = line([p1, p2], color = 'red', thickness = 2)    # Sketch tangent vector
show(p + utv)

```

$\mathbf{r}'(t) = (-\sin(t), 1)$ # 접선벡터 $\mathbf{r}'(t)$





예제 6 $-5 \leq t \leq 5$ 에서 곡선 $\mathbf{r}(t) = \langle 4t, 3\cos t, 3\sin t \rangle$ 의 길이(length)를 구하여라.

```

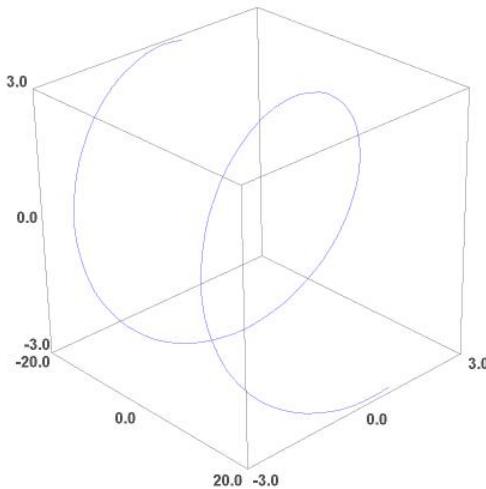
var('t')                                     # https://sagecell.sagemath.org/
r(t) = (4*t, 3*cos(t), 3*sin(t))      # Define r(t)
dr = diff(r(t), t)
s(t) = dr.norm()
print(integral(s(t), t, -5, 5))    # length of the curve
parametric_plot3d(r(t), (t, -5, 5))   # Sketch curve

```

50

곡선의 길이는 $\int_{-5}^5 \| \mathbf{r}'(t) \| dt = 50$

■



■

2.3 편도함수(Partial Derivative)와 그래디언트(gradient)

참고 동영상: <http://youtu.be/LR89Ct3cEDY>

실습 사이트: <http://matrix.skku.ac.kr/Cal-Book1/Ch13/>

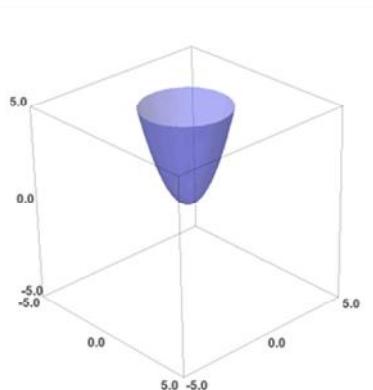
<http://matrix.skku.ac.kr/Cal-Book/part2/CS-Sec-13-3-Sol.html>

14강 동영상, 편도함수 <https://youtu.be/cvsBYZT4SZg> (25:26)

■ x, y 를 독립적으로 변화하는 두 변수라 하고 z 를 제 3의 변수라 하자. x, y 의 값이 각각 정해지면 여기에 대응하여 z 의 값이 정해질 때 z 를 두 변수 x 와 y 의 함수라 하고 이것을 $z = f(x, y)$ 로 표시한다. 같은 방법으로 더 많은 변수의 함수도 정의 할 수 있다. 이와 같이 이변수 이상의 함수를 일반적으로 다변수함수라 한다. 이 장에서는 주로 이변수함수만을 취급하지만 이 함수의 성질은 그대로 다변수함수까지 확장할 수 있다.

■ 공간에 있어서 x, y 및 $z = f(x, y)$ 를 좌표로 하는 점을 생각하고 x 와 y 를 움직이면 그들 점은 일반적으로 하나의 곡면을 나타내는데 이 곡면을 함수 $z = f(x, y)$ 의 그 래프라 부른다.

[2변수 함수의 그래프–곡면]



정의.

편도함수(Partial Derivative)

$z = f(x, y)$ 가 x 와 y 의 함수라 하자. f 의 x 에 관한 편도함수는 다음과 같이 정의된다.

$$\frac{\partial z}{\partial x} = \frac{\partial f}{\partial x} = z_x = f_x(x, y) = \lim_{h \rightarrow 0} \frac{f(x+h, y) - f(x, y)}{h}$$

마찬가지로 f 의 y 에 관한 편도함수(Partial Derivative)는 다음과 같이 정의된다.

$$\frac{\partial z}{\partial y} = \frac{\partial f}{\partial y} = z_y = f_y(x, y) = \lim_{h \rightarrow 0} \frac{f(x, y+h) - f(x, y)}{h}$$

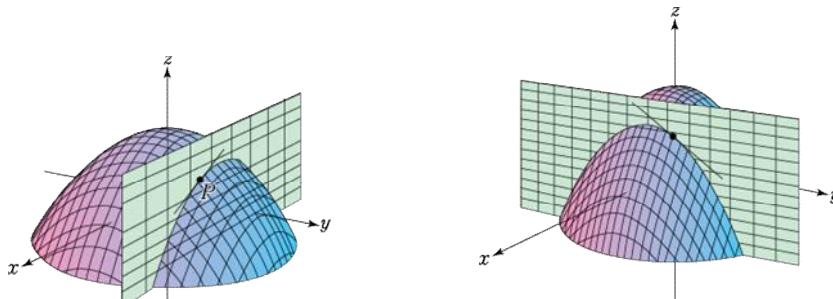
- $z = f(x, y)$ 는 아래 그림에서 보듯이 곡면의 방정식이다. 곡면의 임의의 한 점 $P(x_1, y_1, z_1)$ 을 지나고 xz 평면과 평행인 평면으로 곡면을 자르면 한 곡선 APB 를 얻는다. 한 점이 이 곡선을 따라 움직이면 그 z 좌표는 x 에 따라서 변하지만 y 는 변하지 않는다. 그러므로 점 P 에서의 곡선 APB 의 접선의 기울기는 이 점에서의 x 에 관한 z 의 변화율이다.

마찬가지로 점 P 에서의 곡선 CPD 의 접선의 기울기는 이 점에서의 y 에 관한 z 의 변화율이다. 따라서 다음을 알 수 있다.

$$\left[\frac{\partial z}{\partial x} \right]_{\substack{x=x_1 \\ y=y_1}} = f_x(x_1, y_1) = \tan \alpha = P \text{에서의 } APB \text{의 기울기} \quad (\text{왼쪽그림})$$

$$\left[\frac{\partial z}{\partial y} \right]_{\substack{x=x_1 \\ y=y_1}} = f_y(x_1, y_1) = \tan \beta = P \text{에서의 } CPD \text{의 기울기} \quad (\text{오른쪽그림})$$

[편도함수의 시각적 의미]



$z = f(x, y)$ 와 평면 $y = y_1$ 의
교선(곡선 APB)의 점
 P 에서의 기울기

$z = f(x, y)$ 와 평면 $x = x_1$ 의
교선(곡선 CPD)의 점 P 에서의
기울기

예제 7 함수 $f(x, y) = 24xy - 7x^2y^2$ 의 x, y 에 관한 편도함수를 각각 구하여라.

```
var('x, y')
f(x, y) = 24*x*y - 7*x^2*y^2
f_x = diff(f(x, y), x, 1)    # 편도함수  $f_x(x, y)$ 
f_y = diff(f(x, y), y, 1)    # partial derivatives  $f_y(x, y)$ 
print("fx(x, y) =", f_x)
print("fy(x, y) =", f_y)
```

$fx(x, y) = -14*x*y^2 + 24*y$ # 편도함수 $f_x(x, y)$

$fy(x, y) = -14*x^2*y + 24*x$ ■

- 함수 $z = f(x, y)$ 의 편도함수는 일반적으로 x 와 y 의 함수가 되므로 이것을 다시 편미분할 수 있다. 그 결과를 나타내기 위하여 다음의 기호를 사용하여

$$\frac{\partial}{\partial x} \left(\frac{\partial z}{\partial x} \right) = \frac{\partial^2 z}{\partial x^2} = z_{xx} = \frac{\partial^2 f}{\partial x^2} = f_{xx}(x, y)$$

$$\frac{\partial}{\partial x} \left(\frac{\partial z}{\partial y} \right) = \frac{\partial^2 z}{\partial x \partial y} = z_{yx} = \frac{\partial^2 f}{\partial x \partial y} = f_{yx}(x, y)$$

$$\frac{\partial}{\partial y} \left(\frac{\partial z}{\partial x} \right) = \frac{\partial^2 z}{\partial y \partial x} = z_{xy} = \frac{\partial^2 f}{\partial y \partial x} = f_{xy}(x, y)$$

$$\frac{\partial}{\partial y} \left(\frac{\partial z}{\partial y} \right) = \frac{\partial^2 z}{\partial y^2} = z_{yy} = \frac{\partial^2 f}{\partial y^2} = f_{yy}(x, y)$$

등으로 표시하고 이들을 2계 편도함수(second-order partial derivative)라 한다. 마찬가지로 n 계 편도함수를 정의할 수 있다. [Note: 2계 편도함수는 해시안(Hessian) 행렬에 활용된다]

정리. $f_{xy} = f_{yx}$ 되는 조건

f_{xy} 와 f_{yx} 가 존재하고 모두 연속이면 $f_{xy} = f_{yx}$ 이다.

- 이 정리는 모든 고계 편도함수에 대하여 항상 성립한다. 앞으로는 이런 형태의 함수만 다룬다.

예제 8 함수 $f(x, y) = \sin(x - y) + e^{-xy}$ 의 2계 편도함수(2nd-order partial derivative) 4개를 모두 구하여라.

<https://sagecell.sagemath.org/>

```
var('x, y') # https://sagecell.sagemath.org/
```

```
f = sin(x - y) + exp(-x*y)
fxx = diff(f, x, 2)
fyy = diff(f, y, 2)
fxy = diff(f, x, y)
fyx = diff(f, y, x)
print("fxx =", fxx)
print("fyy =", fyy)
print("fxy =", fxy)
print("fyx =", fyx)
```

```
fxx = y^2*e^(-x*y) - sin(x - y)
fyy = x^2*e^(-x*y) - sin(x - y)
fxy = x*y*e^(-x*y) - e^(-x*y) + sin(x - y)
fyx = x*y*e^(-x*y) - e^(-x*y) + sin(x - y)
```

■

■ 미적분학에서, 연쇄 법칙(連鎖法則, Chain rule)은 합성함수의 도함수에 대한 공식이다. 그 공식이 마치 사슬이 이어져 있는 것 같다 해서, 합성함수의 미분 공식을 연쇄법칙이라고 한다.. 연쇄 법칙을 적분에 거꾸로 적용한 것이 치환 적분이다.

정리. 연쇄법칙(Chain Rule)

(1) $z = f(x, y)$ 가 x, y 에 관해서 편미분가능이며 편도함수 $f_x(x, y), f_y(x, y)$ 가 연속이고, x, y 가 또 다른 변수 t 의 미분가능인 함수이면, z 는 t 에 관해서 미분가능이며

$$\frac{dz}{dt} = \frac{\partial z}{\partial x} \frac{dx}{dt} + \frac{\partial z}{\partial y} \frac{dy}{dt}$$

이다.

(2) $z = f(x, y)$, $y = g(x)$ 일 때의 연쇄법칙은

$$\frac{dz}{dx} = \frac{\partial z}{\partial x} + \frac{\partial z}{\partial y} \frac{dy}{dx}$$

가 된다.

(3) $z = f(x, y)$ 가 연속인 편도함수를 가지며, $x = \phi(u, v)$, $y = \psi(u, v)$ 가 편미분 가능이면,

$$\frac{\partial z}{\partial u} = \frac{\partial z}{\partial x} \frac{\partial x}{\partial u} + \frac{\partial z}{\partial y} \frac{\partial y}{\partial u}, \quad \frac{\partial z}{\partial v} = \frac{\partial z}{\partial x} \frac{\partial x}{\partial v} + \frac{\partial z}{\partial y} \frac{\partial y}{\partial v}$$

이다.

예제 9 $z = x^2y$, $x = u^2 + v^2$, $y = 2uv$ 일 때 $\frac{\partial z}{\partial u}$ 를 구하여라.

<https://sagecell.sagemath.org/>

```
var('u, v, s, t') # https://sagecell.sagemath.org/
x = u^2 + v^2
y = 2*u*v
z = s^2*t
print("dz/du =", diff(z(s = x, t = y), u).simplify_full())
print("dz/dv =", diff(z(s = x, t = y), v).simplify_full())
# simplify_full()이 어느 정도 정리를 해주지만 완벽하지는 않다.
```

$$\begin{aligned} dz/du &= 10*u^4*v + 12*u^2*v^3 + 2*v^5 \\ dz/dv &= 2*u^5 + 12*u^3*v^2 + 10*u*v^4 \end{aligned}$$

■

정의. 방향도함수(directional derivative), 그레디언트(gradient), 헤시안(Hessian)

(1) $\mathbf{u} = (u_1, u_2)$ 를 단위벡터(크기가 1인 벡터)라 하자. 그러면 점 (a, b) 에서 \mathbf{u} 방향으로의 f 의 방향도함수(directional derivative)는 다음과 같이 정의된다.

$$D_{\mathbf{u}} f(a, b) = \lim_{t \rightarrow 0} \frac{f(a + tu_1, b + tu_2) - f(a, b)}{t}$$

- $\mathbf{u} = (1, 0)$ 일 때 $D_{\mathbf{u}}(a, b) = f_x(a, b)$ 이고, $\mathbf{v} = (0, 1)$ 일 때 $D_{\mathbf{v}}(a, b) = f_y(a, b)$ 이다.
- $\gamma(t)$ 가 점 (a, b) 를 지나고, $\mathbf{u} = (u_1, u_2)$ 에 평행한 직선이라 하자. 즉

$$\gamma(t) = (a, b) + t(u_1, u_2)$$

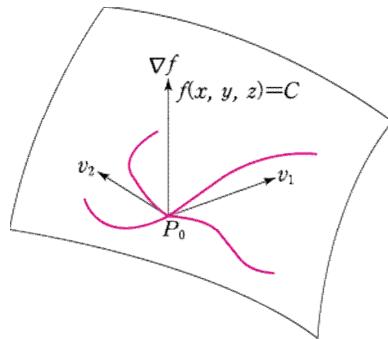
그러면 다음이 성립한다.

$$D_{\mathbf{u}} f(a, b) = \left. \frac{d}{dt} f(\gamma(t)) \right|_{t=0}.$$

(2) $f(x, y)$ 의 그래디언트(gradient)는 f 의 편도함수를 성분으로 갖는 벡터로 다음과 같이 정의된다.

$$\text{grad } f = \nabla f(x, y) = \left\langle \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right\rangle$$

[그래디언트(gradient)의 시작적 의미]



∇f 는 미분가능함수 f 의 P_0 점에서의 접선벡터와 직교인 벡터. 접선벡터는 표면 위에 있다.

(3) $f(x, y)$ 의 헤시안(Hessian)은 f 의 2계 편도함수를 성분으로 갖는 행렬로 다음과 같이 정의된다.

$$\nabla^2 f(x, y) = \begin{bmatrix} f_{xx} & f_{xy} \\ f_{yx} & f_{yy} \end{bmatrix}$$

- f 의 이계 편도함수가 연속인 경우에는 $f_{xy} = f_{yx}$ 이므로 $\nabla^2 f(x, y)$ 이 대칭행렬이 된다.

- 같은 방법으로 독립변수가 3개 이상인 다변수함수 f 에 대하여도 f 의 그레디언트와 헤시안이 정의된다. 즉 n 변수 함수 $f(x_1, x_2, \dots, x_n)$ 에 대하여 f 의 그레디언트와 헤시안은 다음과 같다.

$f(x_1, x_2, \dots, x_n)$ 의 **그레디언트(gradient)**: $\text{grad } f = \nabla f(x_1, x_2, \dots, x_n) = \left\langle \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_n} \right\rangle$

$$f(x_1, x_2, \dots, x_n) \text{의 } \text{헤시안(Hessian)}: \nabla^2 f(x_1, x_2, \dots, x_n) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \dots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \dots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

정리. 방향도함수(directional derivative)

f 가 점 (a, b) 에서 미분가능하고 $\mathbf{u} = \langle u_1, u_2 \rangle$ 가 단위벡터라 하면 다음이 성립한다.

$$D_{\mathbf{u}} f(a, b) = f_x(a, b)u_1 + f_y(a, b)u_2 = \nabla f(a, b) \cdot \mathbf{u}, \quad \|\mathbf{u}\| = 1$$

예제 10 점 $(1, 2)$ 에서 x 축의 양의 방향과 이루는 각(angle)이 $\pi/6$ 인 단위벡터

$$\mathbf{u} = (u_1, u_2) = (\cos(\pi/6), \sin(\pi/6))$$

방향으로의 함수 $f(x, y) = x^3 - 3xy + 4y^2$ 의 방향도함수를 구하여라.

[Find $D_{\mathbf{u}} f(1, 2) = f_x(1, 2)u_1 + f_y(1, 2)u_2 = \nabla f(1, 2) \cdot \mathbf{u} = \nabla f(1, 2) \cdot (u_1, u_2) = \nabla f(1, 2) \cdot (\cos(\pi/6), \sin(\pi/6))$]

<https://sagecell.sagemath.org/>

```
var('x, y') # https://sagecell.sagemath.org/
f(x, y) = x^3 - 3*x*y + 4*y^2
u = vector([cos(pi/6), sin(pi/6)])
gradf = f.gradient() # 함수 f의 그레디언트
print("Duf(1,2) =", gradf(1, 2).dot_product(u))
```

$$\text{Duf}(1,2) = -3/2*\sqrt{3} + 13/2 \quad \# D_{\mathbf{u}} f(1, 2) \text{ where } \mathbf{u} = (u_1, u_2) = (\cos(\pi/6), \sin(\pi/6)) \blacksquare$$

- 내적의 성질을 이용하면 다음을 알 수 있다.

$$D_{\mathbf{u}} f(a, b) = \nabla f(a, b) \cdot \mathbf{u} = \|\nabla f(a, b)\| \cdot \|\mathbf{u}\| \cos \theta = \|\nabla f(a, b)\| \cos \theta, \quad \|\mathbf{u}\| = 1$$

여기서 θ 는 $\nabla f(a, b)$ 과 \mathbf{u} 의 사잇각이다. 따라서 f 의 그래디언트 $\nabla f(a, b)$ 방향이 점 (a, b) 에서 f 가 가장 가파르게 증가하는 방향이고, 음의 그래디언트 $-\nabla f(a, b)$ 방향이 점 (a, b) 에서 f 가 가장 가파르게 감소하는 방향이다. 이는 이후에 학습하게 되는 경사하강법(gradient descent method)의 Key idea를 제공합니다.

- 다면수 함수 f 에 대한 Taylor 정리는 다음과 같다. 이는 한 점 \mathbf{x}_0 근방에서 f 의 선형근사(linear approximation) 및 이차근사식(quadratic approximation)을 구성하는데 사용된다.

(미분가능한) 다변수 함수에 대한 Taylor 근사를 표현하는 훌륭한 방법은 우선 열벡터 $\mathbf{x} = (x_1, x_2) = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ 와 행벡터 $\mathbf{x}^T = [x_1 \ x_2]$ 을 사용하여 1계(1st order)

Taylor 근사가 $f(\mathbf{x}) = f(\mathbf{a}) + (\mathbf{x} - \mathbf{a})^T \nabla f(\mathbf{b})$ 로 표현된다는 것을 이용한다. 여기서 두 번째 항이 행렬곱으로 표현된 것이다. 다음 n 변수 함수에 대한 Taylor 근사에서 2계항(second order term)까지 행렬표현으로 나타내면 아래와 같다.

$$f(\mathbf{x}) = f(\mathbf{a}) + (\mathbf{x} - \mathbf{a})^T \nabla f(\mathbf{a}) + \frac{1}{2} (\mathbf{x} - \mathbf{a})^T H(\mathbf{b})(\mathbf{x} - \mathbf{a})$$

여기서 H 는 $f(x_1, x_2, \dots, x_n)$ 의 대칭행렬인 헤시안(Hessian) 행렬 H 이다.

$$\text{헤시안(Hessian) 행렬 : } \nabla^2 f = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix} = H$$

위의 식에서 $\mathbf{x} = \mathbf{a} + \mathbf{d}$, $\mathbf{b} = \mathbf{a} + t\mathbf{d}$, $t \in (0, 1)$ 로 두면 다음을 얻는다.

정리. 다변수 함수에 대한 Taylor 정리의 행렬 표현

다면수 함수 f 가 연속인 편도함수를 가지면, 적당한 $t \in (0, 1)$ 가 존재하여 다음이 성립한다.

$$f(\mathbf{a} + \mathbf{d}) = f(\mathbf{a}) + \mathbf{d}^T \nabla f(\mathbf{a} + t\mathbf{d})$$

만일 연속인 2계 편도함수를 가지면, 적당한 $t \in (0, 1)$ 가 존재하여 다음이 성립한다.

$$f(\mathbf{a} + \mathbf{d}) = f(\mathbf{a}) + \mathbf{d}^T \nabla f(\mathbf{a}) + \frac{1}{2} \mathbf{d}^T H(\mathbf{b}) \mathbf{d} \quad \text{또는}$$

$$f(\mathbf{a} + \mathbf{d}) = f(\mathbf{a}) + \mathbf{d}^T \nabla f(\mathbf{a}) + \frac{1}{2} \mathbf{d}^T \nabla^2 f(\mathbf{a} + t\mathbf{d}) \mathbf{d}$$

- (위 식에서 $\mathbf{x} = \mathbf{a} + \mathbf{d}$, $\mathbf{a} = \mathbf{x}_0$, $\mathbf{d} = \mathbf{x} - \mathbf{a} = \mathbf{x} - \mathbf{x}_0$, $\mathbf{x}_0 \approx \mathbf{a} + t\mathbf{d}$ 일 때) 점 \mathbf{x}_0 근방에서 $f(\mathbf{x})$ 에 가까운 1차식인 선형근사식(벡터형식 표현)은 다음과 같다.

$$f(\mathbf{x}) \approx f(\mathbf{x}_0) + (\mathbf{x} - \mathbf{x}_0)^T \nabla f(\mathbf{x}_0)$$

- 마찬가지로 점 \mathbf{x}_0 근방에서 $f(\mathbf{x})$ 에 가까운 2차식인 이차근사식은 다음과 같다.

$$f(\mathbf{x}) \approx f(\mathbf{x}_0) + (\mathbf{x} - \mathbf{x}_0)^T \nabla f(\mathbf{x}_0) + \frac{1}{2} (\mathbf{x} - \mathbf{x}_0)^T \nabla^2 f(\mathbf{x}_0) (\mathbf{x} - \mathbf{x}_0) \quad \blacksquare$$

2.4 함수의 극대(Local Maximum), 극소

참고 동영상: <http://youtu.be/oDZUkOEszOQ>

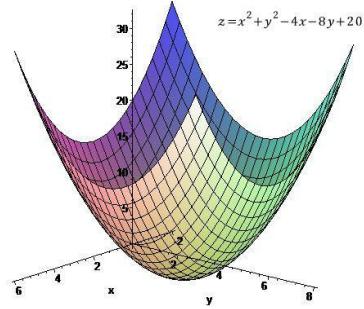
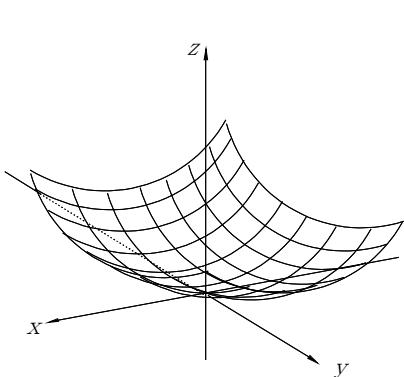
실습 사이트: <http://matrix.skku.ac.kr/Cal-Book1/Ch13/>

<http://matrix.skku.ac.kr/Cal-Book/part2/CS-Sec-13-8-Sol.html>

15강 동영상, 극대, 극소 <https://youtu.be/nR9it9cBjDk> (21:25)

정의. 극대, 극소, 최대, 최소

- (1) (a, b) 근방의 모든 (x, y) 에 대하여 $f(a, b) \geq f(x, y)$ 이 성립하면, $f(x, y)$ 는 (a, b) 에서 극대가 된다고 하며, $f(a, b)$ 를 극댓값(Local Maximum)이라 한다.
- (2) (a, b) 근방의 모든 (x, y) 에 대하여 $f(a, b) \leq f(x, y)$ 이 성립하면, $f(x, y)$ 는 (a, b) 에서 극소가 된다고 하며, $f(a, b)$ 를 극솟값(Local Minimum)이라 한다.
- (3) $f(x, y)$ 의 정의역의 모든 (x, y) 에 대하여 $f(a, b) \geq f(x, y)$ ($f(a, b) \leq f(x, y)$) 이 성립하면 $f(x, y)$ 는 (a, b) 에서 최대(최소)가 된다고 하며, $f(a, b)$ 를 최댓값(최솟값)이라 한다.



[이 변수 함수 극솟값의 시각적 이해]

$$f(x, y) = x^2 + y^2 - xy - 2x - 2y + 4$$

- 함수 $f(x, y)$ 가 (a, b) 에서 극값(극댓값 또는 극솟값)을 갖는다고 가정하자. 그러면 $y = b$ 일 때 일변수 함수 $f(x, b)$ 는 $x = a$ 에서 극값을 갖게 된다. 따라서

$f_x(a, b) = 0$ 이 성립한다. 마찬가지로 $x = a$ 일 때 일변수 함수 $f(a, y)$ 는 $y = b$ 에서 극값을 갖게 된다. 따라서 $f_y(a, b) = 0$ 이 성립한다. 이로부터 다음 정리를 얻는다.

정리. Fermat의 임계점 정리(Fermat's theorem on critical points)

f 가 (a, b) 에서 극대 또는 극소가 되고, f 의 편도함수가 존재하면 다음이 성립한다.

$$f_x(a, b) = 0, \quad f_y(a, b) = 0 \quad [\Leftrightarrow \nabla f(a, b) = \mathbf{0}]$$

정의. 임계점(critical point)

$\nabla f(a, b) = \mathbf{0}$ 을 만족하는 점 (a, b) 를 f 의 임계점(critical point)이라 한다.

예제 11 함수 $f(x, y) = 3x^2 + 2y^2 - 6x + 8y$ 의 임계점을 모두 구하여라.

풀이. $f_x = 6x - 6$ and $f_y = 4y + 8$ 이므로 $\nabla f(x, y) = \mathbf{0}$ 을 만족하는 점은 $(1, -2)$ 이다.

<https://sagecell.sagemath.org/>

```
var('x, y') # https://sagecell.sagemath.org/
f(x, y) = 3*x^2 + 2*y^2 - 6*x + 8*y
gradf = f.gradient() # 함수 f의 그래디언트
solve([gradf[0] == 0, gradf[1] == 0], x, y)

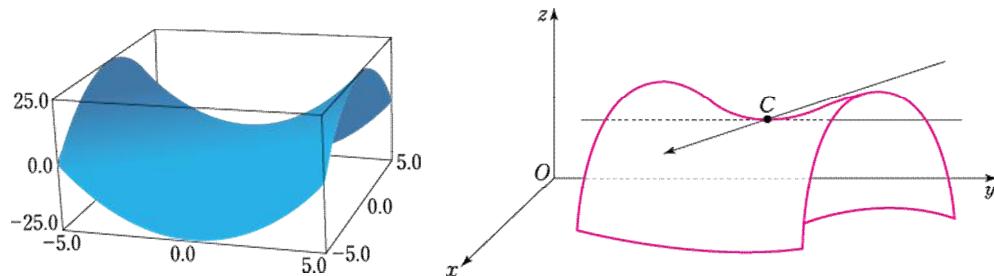
[[x == 1, y == -2]]
```

예제 12 함수 $f(x, y) = 4xye^{-x^2-y^2}$ 의 임계점을 모두 구하여라.

```
var('x, y')
f(x, y) = 4*x*y*exp(-x^2 - y^2)
gradf = f.gradient() # 함수 f의 그래디언트
solve([gradf[0] == 0, gradf[1] == 0], x, y)
```

$[[x = 0, y = 0], [x = -1/2\sqrt{2}, y = -1/2\sqrt{2}], [x = 1/2\sqrt{2}, y = -1/2\sqrt{2}], [x = -1/2\sqrt{2}, y = 1/2\sqrt{2}], [x = 1/2\sqrt{2}, y = 1/2\sqrt{2}]]$
 # 5개의 임계점 ■

- 임계점이라 해서 반드시 극대 또는 극소가 되는 것은 아니다. 예를 들어, 다음 그림과 같이 곡면이 말의 안장과 같은 모양을 하고 있는 경우, 임계점이지만 극 대도 극소도 되지 않는 경우가 있기 때문이다. 이러한 점을 안장점 (saddle point)이라 부른다.



[안장점 (saddle point)의 시각적 이해]

- Part I에서 정의한 양의 정부호 등의 성질이 극대, 극소 문제를 푸는데 어떻게 적용되는지를 알아보자. 먼저, \mathbb{R}^n 에서 두 번 편미분가능한 실변수 함수 $f : \mathbb{R}^n \rightarrow \mathbb{R}$ 를 생각하자. $f(\mathbf{a})$ 가 극솟값 또는 극댓값이 되려면, 앞서 학습한 [Fermat의 임계점 정리]에 의해 우선 $\mathbf{x} = \mathbf{a}$ 는 임계점이어야 한다. 즉, f 의 편도함수는 \mathbf{a} 에서 모두 0이어야 한다.

이제 $f(\mathbf{a})$ 가 극솟값 또는 극댓값이 되기 위한 충분조건을 얻으려면 임계점 $\mathbf{x} = \mathbf{a}$ 에서 f 의 2계 편도함수로 구성된 해시안을 생각해보아야 한다. 왜냐하면 점 \mathbf{a} 근방에서 f 의 이차근사식은

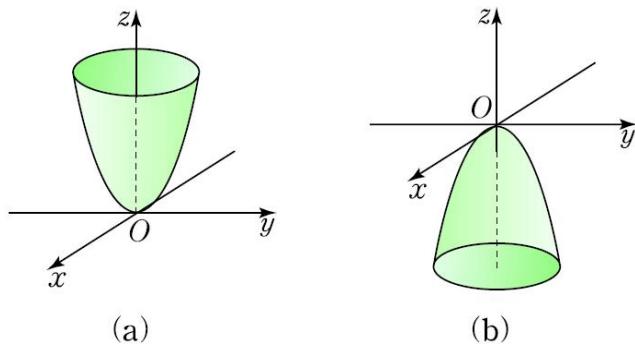
[이변수함수 일 때는 $f(x,y) = f(a,b) + [f_x(a,b)(x-a) + f_y(a,b)(y-b)]$

$$+ \frac{1}{2!} [f_{xx}(a,b)(x-a)^2 + 2f_{xy}(a,b)(x-a)(y-b) + f_{yy}(y-b)^2] + \dots]$$

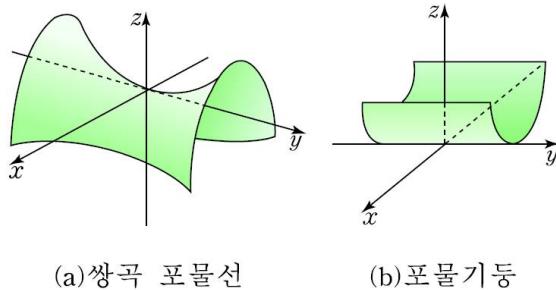
n 변수함수 일 때는 $f(\mathbf{x}) \approx f(\mathbf{a}) + \nabla f(\mathbf{a})^T(\mathbf{x}-\mathbf{a}) + \frac{1}{2}(\mathbf{x}-\mathbf{a})^T \nabla^2 f(\mathbf{a})(\mathbf{x}-\mathbf{a})$

이므로, 임계점 $\mathbf{x} = \mathbf{a}$ (즉 $\nabla f(\mathbf{a}) = \mathbf{0}$)에서 함수 f 의 극대, 극소를 결정하는 것은 $(\mathbf{x} - \mathbf{a})^T \nabla^2 f(\mathbf{a})(\mathbf{x} - \mathbf{a})$ 이기 때문이다. 따라서 $\mathbf{d} = \mathbf{x} - \mathbf{a}$ 로 치환하고 헤시안 (Hessian) 행렬 $H = \nabla^2 f(\mathbf{a})$ 로 놓은 후, 이차형식인 $q(\mathbf{d}) = \mathbf{d}^T H \mathbf{d}$ 의 극솟값/극댓값 분석을 하면 된다.

- 만일 H 의 고윳값이 모두 양이라면, 함수 f 는 $\mathbf{x} = \mathbf{a}$ 근방에서 아래 그림 (a)와 같이 위쪽이 열린 포물면(paraboloid)에 가까우므로 $f(\mathbf{a})$ 는 극솟값이 된다. 또한 H 의 고윳값이 모두 음이라면 함수 f 는 $\mathbf{x} = \mathbf{a}$ 근방에서 그림 (b)와 같이 아래쪽이 열린 포물면에 가까우므로 $f(\mathbf{a})$ 는 극댓값이 된다.



- 또한 만일 H 의 고윳값이 λ_1, λ_2 가 모두 영이 아니고 서로 다른 부호이면, 안장 모양의 쌍곡포물면(hyperbolic paraboloid)에 가까우므로 $\mathbf{x} = \mathbf{a}$ 는 안장점이 된다. 고윳값 중 하나가 0이면 함수 f 의 모습은 포물기둥에 가까워진다. 이를 정리로 나타내면 다음과 같다.



정리. 극대, 극소, 안장점

함수 $f : \mathbb{R}^n \rightarrow \mathbb{R}$ 가 임계점 $\mathbf{x} = \mathbf{a}$ 에서 연속인 2계 편도함수를 갖고, 이 점에서 f 의 이차형식을 $q(\mathbf{d}) = \mathbf{d}^T H \mathbf{d}$ 라고 할 때, 다음이 성립한다.

- (1) q 가 양의 정부호이면 $f(\mathbf{a})$ 는 극솟값이다.
- (2) q 가 음의 정부호이면 $f(\mathbf{a})$ 는 극댓값이다.
- (3) q 가 부정부호이면, $f(\mathbf{a})$ 는 극댓값도 아니고 극솟값도 아니다.

이때, $\mathbf{x} = \mathbf{a}$ 를 안장점(saddle point)이라고 한다.

■ 특히 $n=2$, 즉 f 가 이변수 함수일 때는 $H = \begin{bmatrix} f_{xx} & f_{xy} \\ f_{yx} & f_{yy} \end{bmatrix}$ 이므로 선행 주 소행렬식(leading principal minor)은 다음과 같다.

$$\Delta_1 = f_{xx}, \quad \Delta_2 = \begin{vmatrix} f_{xx} & f_{xy} \\ f_{yx} & f_{yy} \end{vmatrix} = f_{xx}f_{yy} - \{f_{xy}\}^2 \quad (\text{Recall 이차형식})$$

이를 활용하면 다음과 같이 미적분학에서 학습한 이변수함수의 극대극소판정법을 얻는다.

정리. 극소, 극대 판정법

점 (a, b) 의 근방에서 2변수 함수 f 가 연속인 2계 편도함수를 갖고, $f_x(a, b) = f_y(a, b) = 0$ 라 하자.

- (1) $f_{xx}(a, b)f_{yy}(a, b) - \{f_{xy}(a, b)\}^2 > 0$ 이고, $f_{xx}(a, b) > 0$ 이면, $f(a, b)$ 는 극솟값이다.
- (2) $f_{xx}(a, b)f_{yy}(a, b) - \{f_{xy}(a, b)\}^2 > 0$ 이고, $f_{xx}(a, b) < 0$ 이면, $f(a, b)$ 는 극댓값이다.
- (3) $f_{xx}(a, b)f_{yy}(a, b) - \{f_{xy}(a, b)\}^2 < 0$ 이면, $f(a, b)$ 는 극댓값도 극솟값도 아니다. 점 (a, b) 는 f 의 안장점이다.

이것은 1장의 마지막 부분, 이차형식의 양의 정부호, 음의 정부호, 부정부호의 개념과 일치한다. 따라서 선행 주 소행렬식(leading principal minor) 또는 고윳값만 보고 그 부호만을 이용하여, 같은 ‘이변수함수의 극대극소판정법’에 도달하게 된다.

같은 식으로 3변수 이상을 갖는 다변수함수에 대하여도, 선행 주 소행렬식(leading principal minor)의 부호만 확인하여 극대극소를 판정하는 것이 가능하므로 자연스레 다변수함수의 극대극소판정법을 얻는다.

예제 13 3변수 함수 $f(x, y, z) = x^2 + xz - 3\cos y + z^2$ 의 극값을 구하여라.

풀이. 우선 임계점을 모두 구한다. $\frac{\partial f}{\partial x} = 2x + z$, $\frac{\partial f}{\partial y} = 3 \sin y$, $\frac{\partial f}{\partial z} = x + 2z$, $\frac{\partial^2 f}{\partial y^2} = 3 \cos y$

이므로 f 의 임계점은 n 이 정수일 때 $\mathbf{a} = (0, n\pi, 0)$ 이다. 각각의 임계점에서 해시안을 구한다.

(i) 임계점 $\mathbf{a} = (0, 2k\pi, 0)$ 에서, f 의 해시안(Hessian) 행렬은

$$H_1 = \nabla^2 f(x, y, z) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial x \partial z} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} & \frac{\partial^2 f}{\partial y \partial z} \\ \frac{\partial^2 f}{\partial z \partial x} & \frac{\partial^2 f}{\partial z \partial y} & \frac{\partial^2 f}{\partial z^2} \end{bmatrix} = \begin{bmatrix} 2 & 0 & 1 \\ 0 & 3 & 0 \\ 1 & 0 & 2 \end{bmatrix}$$

이고 $\Delta_1 = 2 > 0$, $\Delta_2 = 6 > 0$, $\Delta_3 = 9 > 0$ 이므로 f 는 $\mathbf{x} = \mathbf{a}$ 에서 양의 정부호이다. 따라서 $f(0, 2k\pi, 0) = -3$ 은 f 의 극솟값이다. ■

```
H = matrix([[2, 0, 1], [0, 3, 0], [1, 0, 2]]) # 행렬 입력
m, n = H.nrows(), H.ncols() # 행렬의 크기
if m != n:
    raise ValueError("The matrix must be square.") # 정사각행렬이 아니면 에러
```

```
for i in range(1, n + 1):
    H_principal_minor = H.submatrix(0, 0, i, i).det() # 선행 주 소행렬식
    print("The ", i, "th principal minor is ", H_principal_minor)
```

The 1 th principal minor is 2

The 2 th principal minor is 6

The 3 th principal minor is 9 # 양의 정부호, $f(0, 2k\pi, 0) = -3$ 은 f 의 극솟값 ■

Part I에서 이미 학습한 바와 같이 f 의 헤시안(Hessian) 행렬 H_1 의 고윳값을 이용하여 3변수함수의 극대극소를 판정할 수도 있다. 3차 행렬 H_1 의 고윳값을 계산하면

```
H = matrix([[2, 0, 1], [0, 3, 0], [1, 0, 2]]) # 행렬 입력
print(H.eigenvalues()) # 행렬의 고윳값
```

[1, 3, 3]

$\mathbf{a} = (0, 2k\pi, 0)$ 에서의 헤시안행렬은 고윳값 3개가 모두 양수이므로, $f(0, 2k\pi, 0) = -3$ 은 f 의 극솟값이다. ■

(ii) 임계점 $\mathbf{a} = (0, (2k+1)\pi, 0)$ 에서의, f 의 헤시안 행렬은

$$H_2 = \nabla^2 f(x, y, z) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial x \partial z} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} & \frac{\partial^2 f}{\partial y \partial z} \\ \frac{\partial^2 f}{\partial z \partial x} & \frac{\partial^2 f}{\partial z \partial y} & \frac{\partial^2 f}{\partial z^2} \end{bmatrix} = \begin{bmatrix} 2 & 0 & 1 \\ 0 & -3 & 0 \\ 1 & 0 & 2 \end{bmatrix}$$

이고, $\Delta_1 = 2 > 0$, $\Delta_2 = -6 < 0$, $\Delta_3 = -9 < 0$ 이다. f 는 $\mathbf{x} = \mathbf{a}$ 에서 부정부호이다. 따라서 $\mathbf{x} = \mathbf{a}$ 는 f 의 안장점이다. ■

```
H = matrix([[2, 0, 1], [0, 3, 0], [1, 0, 2]]) # 행렬 입력
m, n = H.nrows(), H.ncols() # 행렬의 크기
if m != n:
    raise ValueError("The matrix must be square.") # 정사각행렬이 아니면 에러
for i in range(1, n + 1):
    H_principal_minor = H.submatrix(0, 0, i, i).det() # 선행 주 소행렬식
    print("The ", i, "th principal minor is ", H_principal_minor)
```

The 1 번째 principal minor 는 2

The 2 번째 principal minor 는 -6

The 3 번째 principal minor 는 -9

(i)과 마찬가지로 H_2 의 고윳값을 계산하면

```
H = matrix([[2, 0, 1], [0, -3, 0], [1, 0, 2]]) # 행렬 입력
print(H.eigenvalues()) # 행렬의 고윳값
```

[3, 1, -3]

양의 고윳값과 음의 고윳값을 모두 가지므로, $\mathbf{x} = \mathbf{a}$ 는 f 의 안장점이다. ■

예제 14 2변수 함수 $f(x, y) = xy - x^2 - y^2 - 2x - 2y + 4$ 의 극값을 구하여라.

```
var('x, y')
f(x, y) = x*y - x^2 - y^2 - 2*x - 2*y + 4
gradf = f.gradient() # 그래디언트 계산
print(solve([gradf[0] == 0, gradf[1] == 0], x, y)) # 임계점 계산
hessf = f.hessian() # 헤시안 계산
H = hessf(-2, -2)
print("The eigenvalues of H are ", H.eigenvalues())

m, n = H.nrows(), H.ncols() # 행렬의 크기
if m != n:
    raise ValueError("The matrix must be square.") # 정사각행렬이 아니면 예러

for i in range(1, n + 1):
    H_principal_minor = H.submatrix(0, 0, i, i).det() # 선행 주 소행렬식
    print("The ", i, "번째 principal minor 는 ", H_principal_minor)
```

[[x == -2, y == -2]]

The eigenvalues of H are [-3, -1]

The 1 번째 principal minor 는 -2

The 2 번째 principal minor 는 3

선행 주 소행렬식의 부호가 음으로 시작하여, 음과 양이 순서대로 교차되므로, $(-2, -2)$ 에서의 f 의 헤시안 행렬의 고유값이 모두 음이라는 의미이고) 아래쪽이 열린 포물면 모양이므로 극댓값을 갖게 된다. f 는 $(-2, -2)$ 에서 극댓값 $f(-2, -2) = 8$ 을 갖는다. ■

2.5 Gradient Descent Algorithm(경사-기울기 하강법, 傾斜下降法)

16장 동영상, Gradient Descent Algorithm <https://youtu.be/XWDPAdKhq-Q> (16:29)

머신러닝 및 딥러닝에서 가중치의 최적해를 구할 때 널리 사용되는 경사하강법 (Gradient Descent Algorithm)의 기본 개념은 함수의 기울기(경사)를 구하여 기울기가 낮은 쪽으로 계속 이동시켜서 극값에 이를 때까지 반복시키는 것이다.

[Key Idea 2]

- 이제 제약조건이 없는 최적화(unconstrained optimization) 문제

$$\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x})$$

를 푸는 경사하강법(gradient descent method)에 대하여 살펴보자.

[Fermat의 임계점 정리]에 의해 위 문제의 최적해(optimal solution) \mathbf{x}^* 는 다음을 만족한다.

$$\nabla f(\mathbf{x}^*) = \mathbf{0}$$

따라서 방정식 $\nabla f(\mathbf{x}) = \mathbf{0}$ 을 풀어서 나온 해들이 최적해가 되는지 판단하면 된다. 그러나 함수 f 가 비선형인 경우는 방정식을 풀어서 임계점을 구하는 것조차도 쉽지 않다. 이런 경우에는 수치적인 방법으로 임계점을 구한다. 최적화문제를 푸는 계산 방법은 대개 반복법(iterative method)으로, 초기 근사해 \mathbf{x}_1 으로부터 시작하여 특정한 반복단계를 거쳐 이전보다 나은 근사해 $\mathbf{x}_2, \mathbf{x}_3, \dots$ 를 생성한다. 목표는 k 번째 근사해 \mathbf{x}_k 또는 극한값 $\mathbf{x}_k \rightarrow \mathbf{x}^*$ 에서 $\nabla f(\mathbf{x}) = \mathbf{0}$ 을 만족하도록 하는 것이다.

- k 번째 반복단계는 보통 다음과 같은 (직선의 벡터방정식) 형식으로 구성된다.

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$$

여기서 \mathbf{d}_k 는 탐색방향(search direction), α_k 는 step-size (머신러닝에서는 이를 learning rate)라 한다. 즉 \mathbf{x}_k 상에서 \mathbf{d}_k 방향으로, (\mathbf{d}_k 의) α_k 배 만큼 이동하여 \mathbf{x}_{k+1}

을 생성한다.

- 보통 \mathbf{d}_k 는 합수값이 감소하는 방향으로 정한다. 즉 다음을 만족한다.

$$D_{\mathbf{d}_k} f(\mathbf{x}_k) = \nabla f(\mathbf{x}_k) \cdot \mathbf{d}_k = \mathbf{d}_k^T \nabla f(\mathbf{x}_k) < 0$$

이러한 \mathbf{d}_k 는 특히 하강방향(descent direction)이라고도 한다. \mathbf{d}_k 방향을 따라 움직이면 합수 f 가 감소한다는 보장이 있으므로 step-size $\alpha_k > 0$ 는 $f(\mathbf{x}_k) > f(\mathbf{x}_{k+1})$ 이 만족되도록 정한다.

- step-size α_k 를 택하는 방법은 대개 다음 세 가지로 나눌 수 있다.

① 모든 k 에 대하여 동일한 step-size ($\alpha_k = \alpha$)를 선택한다. 이를 fixed step-size라 하는데, 일반적으로는 적절한 step-size를 미리 알 수가 없다. 만일 합수 f 에 대하여 Lipschitz smoothness 또는 strong convexity와 같은 조건이 있다면, $\alpha_k = \alpha$ 를 이론적으로 추정(estimate)할 수 있으나 쉽지는 않다.

② 반직선 $\mathbf{x} = \mathbf{x}_k + \alpha \mathbf{d}_k$ ($\alpha \geq 0$) 상에서 합수 f 의 값이 가장 작은 α_k 를 찾는다.

$$f(\mathbf{x}_k + \alpha \mathbf{d}_k) = \min_{\alpha > 0} f(\mathbf{x}_k + \alpha \mathbf{d}_k) := \phi(\alpha)$$

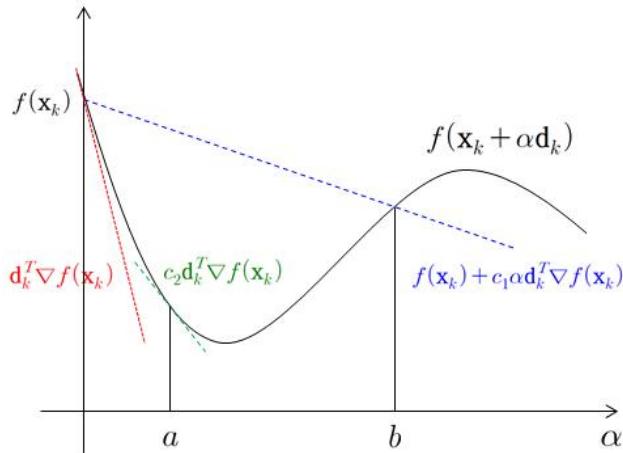
이 방법을 exact line search라 하고, 이때의 α_k 를 optimal step-size라 한다. f 가 이차함수(quadratic function)인 경우를 제외하고는, 대개 α_k 를 구하는데 computational cost가 많이 들어서 잘 사용하지 않는다.

③ 만일 $\min_{\alpha > 0} f(\mathbf{x}_k + \alpha \mathbf{d}_k)$ 을 정확하게 풀지는 않지만 합수값이 충분히 감소한다는 것을 보장하는 α_k 를 선택하는 방법이 있다면, exact line search를 피하여 cost를 상당히 줄일 수 있다. 이 방법을 inexact line search 라 한다. 즉, 다음 조건을 만족하는 α_k 를 찾는다.

$$f(\mathbf{x}_k + \alpha \mathbf{d}_k) \leq f(\mathbf{x}_k) + c_1 \alpha \mathbf{d}_k^T \nabla f(\mathbf{x}_k) \quad \phi(\alpha) \leq \phi(0) + c_1 \alpha \phi'(0)$$

$$\mathbf{d}_k^T \nabla f(\mathbf{x}_k + \alpha \mathbf{d}_k) \geq c_2 \mathbf{d}_k^T \nabla f(\mathbf{x}_k) \quad \phi'(\alpha) \geq c_2 \phi'(0)$$

여기서 $0 < c_1 < c_2 < 1$ 이다. 이를 그림으로 표현하면 다음과 같다.



따라서 위의 두 부등식을 만족하는 α_k 는 폐구간 $[a, b]$ 에서 택하면 된다.

[참고] 위의 두 부등식에서 첫 번째 조건(Armijo condition)은 \mathbf{x}_k 에서 \mathbf{x}_{k+1} 로 이동할 때 함숫값이 충분히 감소(sufficient decrease)해야 한다는 뜻이고, 두 번째 조건(curvature condition)은 너무 작은 α_k 는 배제하여 알고리즘이 적절한 진전(progress)을 이루도록 하는 것이다. 위의 두 조건을 합쳐서 Wolfe condition이라고 부른다.

- 경사하강법(gradient descent method)은 탐색방향을 $\mathbf{d}_k = -\nabla f(\mathbf{x}_k)$ 로 택하는 경우이다. 앞서 살펴본 바와 같이 음의 그레디언트 $-\nabla f(\mathbf{x}_k)$ 방향이 점 \mathbf{x}_k 에서 f 가 가장 가파르게 하강하는 방향이므로, 경사하강법 방법의 아이디어가 쉽게 이해된다(스키장에서 가장 빠르게 하강하는 길을 찾는 알고리즘의 아이디어와 일치한다). 다음은 경사하강법의 알고리즘이다.

[경사하강법] ($\epsilon \ll 1$ 의 의미는 $\epsilon = 10^{-8}$ 같이 $\epsilon \leq 1$ 보다 아주 작다는 의미이다.)

[단계 1] 초기 근사해 $\mathbf{x}_1 \in \mathbb{R}^n$ 과 허용오차(tolerance) $0 \leq \epsilon \ll 1$ 을 준다. $k := 1$ 이라 한다.

[단계 2] $\mathbf{d}_k = -\nabla f(\mathbf{x}_k)$ 를 계산한다. 만일 $\|\mathbf{d}_k\| \leq \epsilon$ 이면, 알고리즘을 멈춘다.

[단계 3] line search를 수행하여 적절한 step-size $\alpha_k > 0$ 를 구한다.

[단계 4] $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$, $k := k + 1$ 라 두고 [단계 2]로 이동한다.

- 다음은 주어진 함수에 경사하강법을 적용한 예시이다. 허용오차는 $\epsilon = 10^{-8}$ 으로 주었다.

[출처] J. Barzilai 와 J. M. Borwein, *Two-Point Step Size Gradient Methods*, IMA J. Numer. Anal. (1988) 8 (1): 141–148.

$$\min_{\mathbf{x} \in \mathbb{R}^4} f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T A \mathbf{x} - \mathbf{b}^T \mathbf{x}, \quad A = \text{diag}(20, 10, 2, 1), \quad \mathbf{b} = (1, 1, 1, 1)$$

여기서 $\nabla f(\mathbf{x}) = A\mathbf{x} - \mathbf{b}$ 이고, $f(\mathbf{x})$ 는 Hessian A 가 양의 정부호(positive definite)인 이차함수이므로 exact line search를 수행하면, α_k 는 다음과 같이 closed-form 으로 나온다.

$$\alpha_k = \frac{\nabla f(\mathbf{x}_k)^T \nabla f(\mathbf{x}_k)}{\nabla f(\mathbf{x}_k)^T A \nabla f(\mathbf{x}_k)}$$

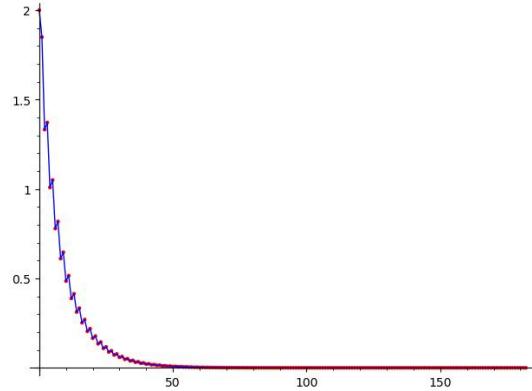
```
# initializing
A = diagonal_matrix(20, 10, 2, 1)
b = vector(1, 1, 1, 1)      # objective function
x0 = vector(0, 0, 0, 0)    # initial guess
g0 = -b                    # initial gradient
r = [] # 그래프를 그리기 위한 용도

# main iteration
for i in range(0, 200):
    gn = g0.norm()
    r.append((i, gn))
    if gn < 10^-8:
        print("Stationary point! Algorithm terminated!")
        break

    w = A*g0
    a = g0.inner_product(g0)/(g0.inner_product(w)) # step-size
    x1 = x0-a*g0
    g1 = A*x1-b
    x0 = x1; g0 = g1
```

```
show(line2d(r) + point(r, color = 'red')) # gradient 의 norm을 그래프로 그림
```

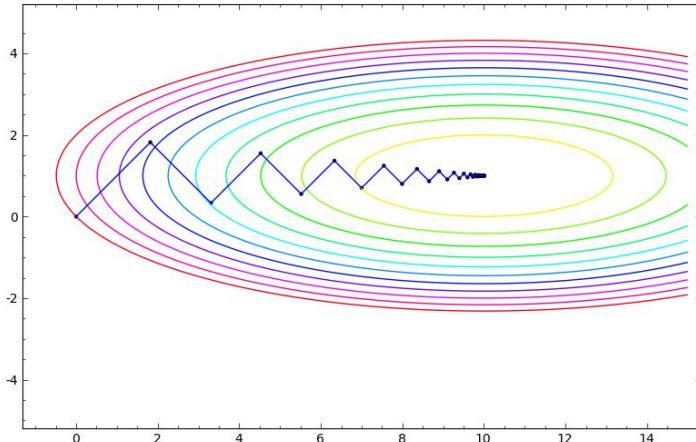
Stationary point! Algorithm terminated!



위의 그래프에서 $\|\nabla f(\mathbf{x}_k)\|$ 이 점차 0에 수렴함을 쉽게 확인할 수 있다.

[참고] 경사하강법은 탐색방향을 현재의 위치 \mathbf{x}_k 의 근방에서 가장 가파르게 하강하는 방향 $\mathbf{d}_k = -\nabla f(\mathbf{x}_k)$ 로 사용한다. 그러나 이 경우 해 근처에서 zigzag 현상이 발생하여 마지막 단계에서 수렴속도가 많이 늦어진다.

$$\min_{\mathbf{x} \in \mathbb{R}^2} f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T A \mathbf{x} - \mathbf{b}^T \mathbf{x}, \quad A = \text{diag}(0.1, 10), \quad \mathbf{b} = (1, 1)$$



- 뉴턴 방법(Newton's method)은 탐색방향을 $\mathbf{d}_k = -(\nabla^2 f(\mathbf{x}_k))^{-1} \nabla f(\mathbf{x}_k)$ 로 택하는 경우를 말한다. 왜냐하면 \mathbf{x}_k 근방에서 f 는 다음의 이차함수를 이용하여 근사화 할 수 있기 때문이다.

$$f(\mathbf{x}_k + \mathbf{d}) \approx f(\mathbf{x}_k) + \nabla f(\mathbf{x}_k)^T \mathbf{d} + \frac{1}{2} \mathbf{d}^T \nabla^2 f(\mathbf{x}_k) \mathbf{d}$$

따라서 \mathbf{x}_k 에서 \mathbf{x}_{k+1} 로 진행하기 위하여, 최적화 문제

$$\min f(\mathbf{x}_k + \mathbf{d}) \approx \min f(\mathbf{x}_k) + \nabla f(\mathbf{x}_k)^T \mathbf{d} + \frac{1}{2} \mathbf{d}^T \nabla^2 f(\mathbf{x}_k) \mathbf{d}$$

의 최적조건으로부터 \mathbf{d}_k 를 얻을 수 있다.

$$\nabla f(\mathbf{x}_k) + \nabla^2 f(\mathbf{x}_k) \mathbf{d} = \mathbf{0} \Rightarrow \mathbf{d}_k = -(\nabla^2 f(\mathbf{x}_k))^{-1} \nabla f(\mathbf{x}_k)$$

이다. 다음은 뉴턴 방법(Newton's method)의 알고리즘(Algorithm)이다.

[뉴턴 방법(Newton's method)]

- [단계 1] 초기 근사해 $\mathbf{x}_1 \in \mathbb{R}^n$ 과 허용오차(tolerance) $0 \leq \epsilon \ll 1$ 을 준다. $k := 1$ 로 한다.
- [단계 2] 만일 $\|\nabla f(\mathbf{x}_k)\| \leq \epsilon$ 이면, 알고리즘을 멈춘다.
- [단계 3] $\mathbf{x}_{k+1} = \mathbf{x}_k - (\nabla^2 f(\mathbf{x}_k))^{-1} \nabla f(\mathbf{x}_k)$ 를 계산한다.
- [단계 4] $k := k + 1$ 로 두고 [단계 2]로 이동한다.

- 다음은 주어진 함수에 뉴턴 방법(Newton's method)을 적용한 예시이다. 허용오차는 $\epsilon = 10^{-8}$ 으로 주었다.

[출처] J.E. Dennis, Jr. and R.B. Schnabel, Numerical Methods for Unconstrained Optimization and Nonlinear Equations, 1996. SIAM

$$\min f(x, y) = (x - 2)^4 + (x - 2)^2 y^2 + (y + 1)^2, \quad (x_1, y_1) = (1, 1)$$

```

var('x, y')
tol = 1e-8
f(x, y) = (x - 2)^4 + (x - 2)^2*y^2 + (y + 1)^2
gradf = f.gradient() # 그레디언트
Hessf = f.hessian() # 해시안
u = vector(RDF, [1, 1]) # 초기 근사해
g = gradf(u[0], u[1])

gn = g.norm()
k = 0
while gn > tol: # Newton's method

    H = Hessf(u[0], u[1])
    s = H.solve_right(-g)
    u = u + s
    print("x%d =" % k, u.n(digits = 7), "f(x%d) =" % k, f(u[0], u[1]).n(digits = 7))
    k = k + 1
    g = gradf(u[0], u[1])
    gn = g.norm()

print("iteration number =", k)

```

```

x0 = (1.000000, -0.500000) f(x0) = 1.500000
x1 = (1.391304, -0.6956522) f(x1) = 0.4092074
x2 = (1.745944, -0.9487981) f(x2) = 0.06489162
x3 = (1.986278, -1.048208) f(x3) = 0.002530930
x4 = (1.998734, -1.000170) f(x4) = 1.631689e-6
x5 = (2.000000, -1.000002) f(x5) = 2.754045e-12
x6 = (2.000000, -1.000000) f(x6) = 1.971425e-24
iteration number = 7

```

뉴턴 방법을 적용하여 얻은 해는 실제 최적해 $\mathbf{x}^* = (2, -1)$ 에 수렴함을 알 수 있다. ■

- $\nabla^2 f(\mathbf{x}_k)$ 이 양의 정부호(positive definite) 행렬이면, 역행렬 $(\nabla^2 f(\mathbf{x}_k))^{-1}$ 도 양의 정부호 행렬이므로, 뉴턴 방법의 탐색방향 $\mathbf{d}_k = -(\nabla^2 f(\mathbf{x}_k))^{-1} \nabla f(\mathbf{x}_k)$ 역시 하강방향(descent direction)이 된다. 즉

$$\nabla f(\mathbf{x}_k) \cdot \mathbf{d}_k = -\nabla f(\mathbf{x}_k)^T (\nabla^2 f(\mathbf{x}_k))^{-1} \nabla f(\mathbf{x}_k) < 0$$

이다. 따라서 \mathbf{d}_k 방향으로 진행하면 함숫값이 감소함을 알 수 있다.

- 그러나 뉴턴 방법은 해시안을 계산해야 하므로, 변수 n 이 큰 함수의 경우 해시안을 계산하는 데 많은 연산이 필요하여 효과적이지 않을 수 있다. 그리고 초기 근사해 \mathbf{x}_1 이 문제의 해 \mathbf{x}^* 의 근방에 있어야만 뉴턴 방법이 수렴한다는 보장이 있으나, 이는 미리 알 수 없으므로 실제 뉴턴 방법을 적용할 때는 step-size $\alpha_k > 0$ 도 같이 고려한다. 즉 적절한 line search를 동반한다. 그 후 \mathbf{x}_{k+1} 을 계산한다.

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k (\nabla^2 f(\mathbf{x}_k))^{-1} \nabla f(\mathbf{x}_k)$$

- 이외에 \mathbf{d}_k 를 택하는 방법에 따라 quasi-Newton method, conjugate gradient method 등이 있으며, α_k 를 실제로 계산하는 알고리즘으로 bisection method, interpolation method, backtracking line search 등이 있다. 이에 대한 자세한 사항은 아래의 [최적화(Optimization)] 관련 도서를 참고하라.

- [1] L. Bottou, F.E. Curtis, and J. Nocedal, *Optimization Methods for Large-Scale Machine Learning*, <https://arxiv.org/pdf/1606.04838.pdf>
- [2] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004. <http://stanford.edu/~boyd/cvxbook/>
- [3] J. Dennis and R. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, SIAM, 1987.
- [4] R. Fletcher, *Practical Methods of Optimization*, 2nd ed., John Wiley and Sons, 1987.
- [5] J. Nocedal and S. Wright, *Numerical Optimization*, 2nd ed., Springer, 2006.
- [6] W. Sun, Y.-X. Yuan, *Optimization theory and methods: Nonlinear programming*, Springer, 2010.

2.6 중적분 (double integral, multiple integral)

참고 동영상: <http://youtu.be/jZ2pAmPZYOE>

실습 사이트: <http://matrix.skku.ac.kr/Cal-Book1/Ch14/>

<http://matrix.skku.ac.kr/Cal-Book/part2/CS-Sec-14-1-Sol.html>

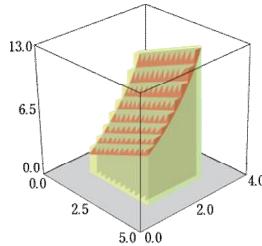
17강 동영상, 중적분 https://youtu.be/T1z_GYt85rI (44:31)

정의.

이중적분

직사각형 영역 $R = [a, b] \times [c, d] = \{(x, y) \in \mathbb{R}^2 \mid a \leq x \leq b, c \leq y \leq d\}$ 에서 $f(x, y)$ 의 이중적분은 다음과 같이 정의된다.

$$\iint_R f(x, y) dA = \lim_{m, n \rightarrow \infty} \sum_{i=1}^m \sum_{j=1}^n f(x_{ij}^*, y_{ij}^*) \Delta A_{ij}$$



[(중적분) 구분구적법을 2축에 적용하는 방법, 긴 직육면체나무막대들의 부피를 구해 더하는 방식]

■ 같은 방법으로 3차원 영역 V 위에서 $w = f(x, y, z)$ 의 3중적분(triple integral)도 정의할 수 있다.

$$\iiint_V f(x, y, z) dx dy dz$$

(3차원 평면 위의 작은 조각 부피에 높이를 곱하여 부피를 구하여 그것 모두를 더하는 방식)

정리.

Fubini의 정리

(연속 2변수함수의 2중적분 경우, 적분순서를 바꾸어도 이중적분값은 같다)

$f(x, y)$ 가 직사각형 영역 $R = [a, b] \times [c, d] = \{(x, y) \in \mathbb{R}^2 \mid a \leq x \leq b, c \leq y \leq d\}$ 에서 연속이면 다음이 성립한다.

$$\iint_R f(x, y) dA = \int_a^b \int_c^d f(x, y) dy dx = \int_c^d \int_a^b f(x, y) dx dy$$

예제 15 다음 이중 적분을 계산하라.

$$\iint_R xy^2 dA, \quad R = [1, 2] \times [0, 2] = [a, b] \times [c, d] = \{(x, y) \in \mathbb{R}^2 \mid a \leq x \leq b, c \leq y \leq d\}$$

<https://sagecell.sagemath.org/>

```
var('x, y') # https://sagecell.sagemath.org/
f(x, y) = x*y^2
print(integral(integral(f, y, 0, 2), x, 1, 2)) # y에 대한 적분 먼저
print(integral(integral(f, x, 1, 2), y, 0, 2)) # x에 대한 적분 먼저
```

4

4 # 순서에 상관없이 이중적분 값이 일치한다. ■

정리. 연속함수의 2중적분의 경우, 적분영역이 직사각형이 아닌 경우!!

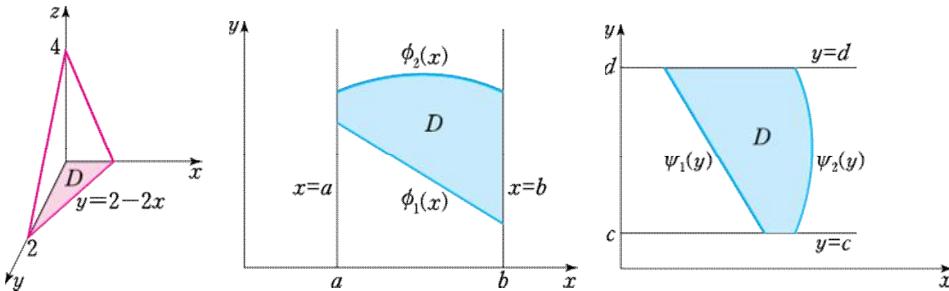
(1) $f(x, y)$ 가 영역 $D = \{(x, y) \mid a \leq x \leq b, g_1(x) \leq y \leq g_2(x)\}$ 에서 연속이면 다음이 성립한다.

$$\iint_D f(x, y) dA = \int_a^b \int_{g_1(x)}^{g_2(x)} f(x, y) dy dx$$

(2) $f(x, y)$ 가 영역 $D = \{(x, y) \mid h_1(y) \leq x \leq h_2(y), c \leq y \leq d\}$ 에서 연속이면 다음이 성립한다.

$$\iint_D f(x, y) dA = \int_c^d \int_{h_1(y)}^{h_2(y)} f(x, y) dx dy$$

[일반적인 영역에서의 중적분 예: D 위의 사면체 부피 구할 때]

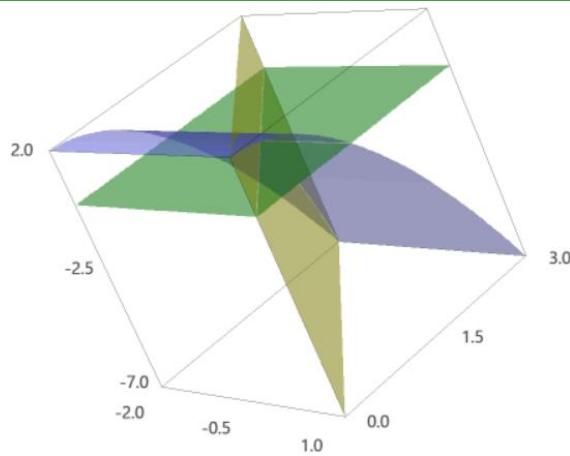


예제 16 다음 이중 적분을 계산하라.

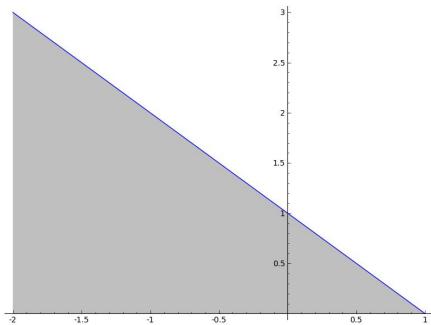
$$\int_{-2}^1 \int_0^{1-x} (2 - y^2) dy dx$$

[실습: <https://sagecell.sagemath.org/>]

```
var('x, y, z')
f(x, y) = 2 - y^2
p1 = implicit_plot3d(y == 1 - x, (x, -2, 1), (y, 0, 3), (z, -7, 2), color = 'yellow',
opacity = 0.4)
p2 = implicit_plot3d(z == 0, (x, -2, 1), (y, 0, 3), (z, -7, 2), color = 'green', opacity
= 0.4)
p3 = plot3d(f(x, y), (x, -2, 1), (y, 0, 3), opacity = 0.4)
p1 + p2 + p3
```



```
var('x, y')      # 이중 적분의 변수 정의
f(x, y) = 2 - y^2          # 주어진 함수
show(plot(1 - x, (x, -2, 1), fill = True))    # 적분영역 (함수)
print(integral(integral(f, y, 0, 1 - x), x, -2, 1))  # 이중 적분 계산
```



(xy-축 상의 적분영역, R)

$$9/4 \quad \#R \text{ 위의 } f(x, y) = 2 - y^2 \text{의 이중적분} \int_{-2}^1 \int_0^{1-x} (2-y^2) dy dx = \frac{9}{4}. \blacksquare$$



일변수 함수의 적분에서 치환적분(변수변환)은 다음과 같이 주어진다.

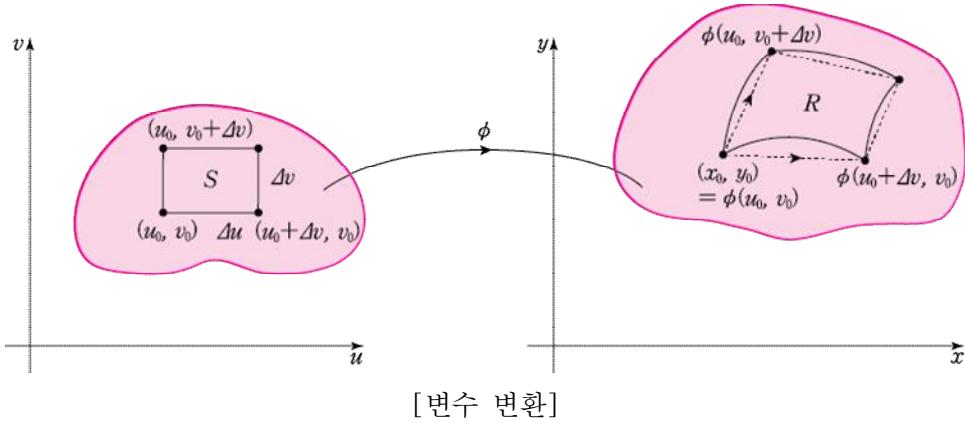
$$\int_a^b f(x) dx = \int_c^d f(\phi(u)) \phi'(u) du$$

여기서 $\phi : [c, d] \rightarrow [a, b]$ 는 전단사이고 연속인 도함수를 갖는다. 이를 다음과 같이 쓸 수 있다.

$$\int_a^b f(x) dx = \int_{c=\phi^{-1}(a)}^{d=\phi^{-1}(b)} f(x(u)) \frac{dx}{du} du$$

이제 변수변환을 이중적분으로 확장한다. 이변수 함수 $f : R \subseteq \mathbb{R}^2 \rightarrow \mathbb{R}$ 가 적분가능하다고 하고, 이중적분 $\iint_R f(x, y) dA$ 를 새로운 변수 u, v 에 관하여 나타내보자.

함수 $\phi(u, v) = (x(u, v), y(u, v))$ 가 연속인 편도함수를 갖고, $\phi : S \subseteq \mathbb{R}^2 \rightarrow R \subseteq \mathbb{R}^2$ 가 uv -평면의 영역 S 를 xy -평면의 영역 R 로 변환시킨다고 하자. 아래 그림과 같이 uv -평면에서 $\Delta u \Delta v$ 을 넓이로 갖는 직사각형 영역 S 와 xy -평면에서 변환된 이미지 R 을 고려해보자.



그리면 xy -평면에서 R의 넓이는 두 벡터 $\phi(u_0 + \Delta u, v_0) - \phi(u_0, v_0)$, $\phi(u_0, v_0 + \Delta v) - \phi(u_0, v_0)$ 로 주어지는 평행사변형으로 근사될 수 있다. 편도함수의 정의를 사용하면

$$\phi(u_0 + \Delta u, v_0) - \phi(u_0, v_0) \approx \frac{\partial \phi}{\partial u} \Delta u, \quad \phi(u_0, v_0 + \Delta v) - \phi(u_0, v_0) \approx \frac{\partial \phi}{\partial v} \Delta v$$

임을 알 수 있고, 두 벡터 \mathbf{a} 와 \mathbf{b} 로 주어지는 평행사변형의 넓이(면적)는 외적(cross product)의 크기(절대값) $|\mathbf{a} \times \mathbf{b}|$ 이므로 R의 넓이의 근사값은 다음과 같다.

$$\frac{\partial \phi}{\partial u} \times \frac{\partial \phi}{\partial v} \neq 0 \text{ 일 때}, \quad \left| \frac{\partial \phi}{\partial u} \Delta u \times \frac{\partial \phi}{\partial v} \Delta v \right| = \left| \frac{\partial \phi}{\partial u} \times \frac{\partial \phi}{\partial v} \right| \Delta u \Delta v.$$

또한

$$\frac{\partial \phi}{\partial u} \times \frac{\partial \phi}{\partial v} = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ \frac{\partial x}{\partial u} & \frac{\partial y}{\partial u} & 0 \\ \frac{\partial x}{\partial v} & \frac{\partial y}{\partial v} & 0 \end{vmatrix} = \begin{vmatrix} \frac{\partial x}{\partial u} & \frac{\partial y}{\partial u} \\ \frac{\partial x}{\partial v} & \frac{\partial y}{\partial v} \end{vmatrix} \mathbf{k} = \begin{vmatrix} \frac{\partial x}{\partial u} & \frac{\partial x}{\partial v} \\ \frac{\partial y}{\partial u} & \frac{\partial y}{\partial v} \end{vmatrix} \mathbf{k}.$$

이 성립하므로 다음 정리를 얻을 수 있다. 이때 $\begin{vmatrix} \frac{\partial x}{\partial u} & \frac{\partial x}{\partial v} \\ \frac{\partial y}{\partial u} & \frac{\partial y}{\partial v} \end{vmatrix}$ 을 $\phi(u, v)$

$= (x(u, v), y(u, v))$ 의 야코비안(Jacobian)이라 하고 $\left| \frac{\partial(x, y)}{\partial(u, v)} \right|$ 으로 나타낸다.

정리. 이중적분의 변수변환

전단사함수 $\phi : S \subseteq \mathbb{R}^2 \rightarrow R \subseteq \mathbb{R}^2$, $\phi(u, v) = (x(u, v), y(u, v))$ 가 연속인 편도함수를 갖고, $\left| \frac{\partial(x, y)}{\partial(u, v)} \right| \neq 0$ 이라 하자. 그러면 연속함수 $f : R \rightarrow \mathbb{R}$ 에 대하여 다음이 성립한다.

$$\iint_R f(x, y) dA = \iint_S f(\phi(u, v)) \left| \frac{\partial \phi}{\partial(u, v)} \right| du dv = \iint_S f(x(u, v), y(u, v)) \left| \frac{\partial(x, y)}{\partial(u, v)} \right| du dv.$$

예제 17 다음 이중 적분을 계산하라.

$$\iint_R 7xy dx dy$$

여기서 R 은 xy -평면상의 네 직선 $x - 2y = 0$, $x - 2y = 2$, $2x + 3y = 0$, $2x + 3y = 3$ 으로 둘러싸인 평행사변형이다.

풀이. 영역 R 의 경계는 $x - 2y = 0$, $x - 2y = 2$, $2x + 3y = 0$, $2x + 3y = 3$ 으로 변수 변환은 다음과 같이 줄 수 있다.

$$x - 2y = u, \quad 2x + 3y = v, \quad u = 0, \quad u = 2, \quad v = 0, \quad v = 3$$

야코비안 $\partial(x, y)/\partial(u, v)$ 을 계산하기 위해 x , y 를 u 와 v 로 표현하면,

$$x = \frac{1}{7}(3u + 2v), \quad y = \frac{1}{7}(v - 2u) \quad [\text{즉}, \quad \begin{bmatrix} x \\ y \end{bmatrix} = T \begin{pmatrix} u \\ v \end{pmatrix}] = \begin{bmatrix} \frac{3}{7}u + \frac{2}{7}v \\ -\frac{2}{7}u + \frac{1}{7}v \end{bmatrix}$$

이므로

$$\frac{\partial \phi}{\partial(u, v)} = \frac{\partial(x, y)}{\partial(u, v)} = \begin{vmatrix} \frac{\partial x}{\partial u} & \frac{\partial x}{\partial v} \\ \frac{\partial y}{\partial u} & \frac{\partial y}{\partial v} \end{vmatrix} = \begin{vmatrix} \frac{3}{7} & \frac{2}{7} \\ -\frac{2}{7} & \frac{1}{7} \end{vmatrix} = \frac{1}{7}.$$

따라서 이중적분은 다음과 같이 계산된다.

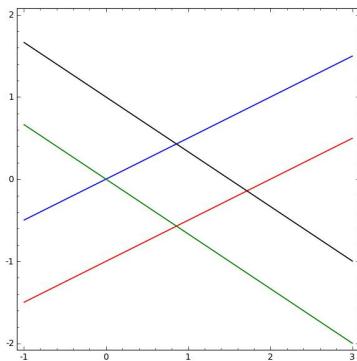
$$\begin{aligned} \iint_R 7xy dx dy &= \iint_S 7 \cdot \frac{1}{7}(3u + 2v) \frac{1}{7}(v - 2u) \left| \frac{\partial(x, y)}{\partial(u, v)} \right| du dv \\ &= \frac{1}{7} \iint_S (3u + 2v)(v - 2u) \left| \frac{1}{7} \right| du dv = \frac{1}{49} \int_0^3 \int_0^2 (2v^2 - uv - 6u^2) du dv = -\frac{3}{7}. \quad \blacksquare \end{aligned}$$

<https://sagecell.sagemath.org/>

```

var('x, y')          # https://sagecell.sagemath.org/
p1 = implicit_plot(x - 2*y == 0, (x, -1, 3), (y, -2, 2))
p2 = implicit_plot(x - 2*y == 2, (x, -1, 3), (y, -2, 2), color = 'red')
p3 = implicit_plot(2*x + 3*y == 0, (x, -1, 3), (y, -2, 2), color = 'green')
p4 = implicit_plot(2*x + 3*y == 3, (x, -1, 3), (y, -2, 2), color = 'black')
show(p1 + p2 + p3 + p4)
var('u, v')
print(solve([u == x - 2*y, v == 2*x + 3*y], x, y)) # x == 3/7*u + 2/7*v, y ==
-2/7*u + 1/7*v
T = (3/7*u + 2/7*v, -2/7*u + 1/7*v)
J = jacobian(T, (u, v)).det()
print("Jacobian =", J)
f(x, y) = 7*x*y
integral(integral(f(T[0], T[1])*abs(J), u, 0, 2), v, 0, 3)      # u=0, u=2,
v=0, v=3

```



$$[[x == 3/7*u + 2/7*v, y == -2/7*u + 1/7*v]]$$

$$\text{Jacobian} = 1/7$$

$$\text{답. } -3/7 \quad \# \iint_R 7xy \, dx \, dy = \frac{1}{49} \int_0^3 \int_0^2 (2v^2 - uv - 6u^2) \, du \, dv = -\frac{3}{7} \quad \blacksquare$$



[극좌표(Polar Coordinates)에서의 2중적분]

극좌표에서의 이중적분은 변수변환의 특수한 예이다. 극좌표

$$x = r \cos \theta, \quad y = r \sin \theta \quad 0 \leq \theta \leq 2\pi \quad \text{and} \quad r \geq 0.$$

$$\text{에서의 야코비안은 } \frac{\partial(x,y)}{\partial(r,\theta)} = \begin{vmatrix} \frac{\partial x}{\partial r} & \frac{\partial x}{\partial \theta} \\ \frac{\partial y}{\partial r} & \frac{\partial y}{\partial \theta} \end{vmatrix} = \begin{vmatrix} \cos\theta & -r\sin\theta \\ \sin\theta & r\cos\theta \end{vmatrix} = r(\cos^2\theta + \sin^2\theta) = r \geq 0$$

으로 주어지고, 이중적분은 다음과 같이 쓸 수 있다.

$$\begin{aligned} \iint_R f(x,y) dx dy &= \iint_S f(x(r,\theta),y(r,\theta)) \left| \frac{\partial(x,y)}{\partial(r,\theta)} \right| dr d\theta \\ &= \iint_S f(x(r,\theta),y(r,\theta)) \left| \frac{\partial(x(r,\theta),y(r,\theta))}{\partial(r,\theta)} \right| dr d\theta \\ &= \iint_S f(r\cos\theta, r\sin\theta) \begin{vmatrix} \frac{\partial x}{\partial r} & \frac{\partial x}{\partial \theta} \\ \frac{\partial y}{\partial r} & \frac{\partial y}{\partial \theta} \end{vmatrix} dr d\theta \\ &= \int_{\alpha}^{\beta} \int_a^b f(r\cos\theta, r\sin\theta) r dr d\theta. \end{aligned}$$

여기서 a, b, α, β 는 적분영역 R 과 변환 $x = r\cos\theta, y = r\sin\theta$ 에 따라 정해진다. 이렇게 얻은 [극좌표변환에 의한 변수변환공식을 이용한 이중적분공식]을 아래에서 풀어봅시다.

예제 18 [주어진 선형변환 함수의 이중적분을 변수변환 공식을 이용하여 구하는 예]

다음 이중적분을 구하여라.

$$f(x,y) = f\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) = T\left(\begin{bmatrix} r \\ \theta \end{bmatrix}\right) = \begin{pmatrix} 2\cos\theta \\ 2\sin\theta \end{pmatrix} \quad \text{이고 영역 } D = [0, 1] \times [0, \pi], \quad D^* = T(D)$$

는 평면상에서의 중심이 $(0,0)$ 이고 반경이 2인 반원이다. 따라서 반원 D^* 에서 $f(x,y) = x^2 + y^2$ 이라 하면

$$\begin{aligned} \iint_{D^*} f(x,y) dx dy &= \iint_D f(x(r,\theta),y(r,\theta)) \left| \frac{\partial(x(r,\theta),y(r,\theta))}{\partial(r,\theta)} \right| dr d\theta \\ &= \int_0^\pi \int_0^2 (2^2(\cos\theta)^2 + 2^2(\sin\theta)^2) r dr d\theta \\ &= \int_0^\pi \int_0^2 4r dr d\theta = \int_0^\pi [2r^2]_0^2 d\theta = 8\pi. \end{aligned}$$

<http://sage.skku.edu/> 또는 <https://sagecell.sagemath.org/>

```
var('r, theta')      # http://sage.skku.edu/ 또는 https://sagecell.sagemath.org/
f=4*r
integral(integral(f, r, 0, 2), theta, 0, pi)
```

답: 8π

- [예제 18]의 극좌표로의 변환을 통한 이중적분은 [예제 19]를 푸는데 꼭 필요하고, 통계학을 학습하는 과정에서 아래 [예제 19]에 대한 이해는 필수적이다.

■ 예제 19 통계학에서 꼭 필요한 적분값 $I = \int_0^\infty e^{-x^2} dx = \int_0^\infty e^{-y^2} dy = \frac{\sqrt{\pi}}{2}$

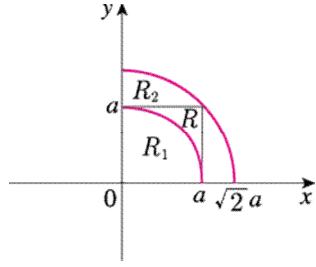
은 기존의 방법으로 손으로는 풀어서 구하기 매우 어렵다. 그러나 코드를 이용하면 바로 구해진다.

<http://sage.skku.edu/> 또는 <https://sagecell.sagemath.org/>

```
var('x')      # http://sage.skku.edu/ 또는 https://sagecell.sagemath.org/
f=exp(-x^2)
integral(f, x, 0, infinity)
```

답: $\frac{\sqrt{\pi}}{2}$

[보충설명] 기존의 방법은 $I = \int_0^\infty e^{-x^2} dx = \int_0^\infty e^{-y^2} dy$ 으므로 Fubini 정리에 의하여 $I^2 = \int_0^\infty \int_0^\infty e^{-(x^2+y^2)} dx dy$ 를 극좌표로의 변환을 통한 이중적분으로 구했다. (Gaussian Integral) <http://mathworld.wolfram.com/GaussianIntegral.html>



풀이. (원점을 중심으로 하고 반지름이 각각 a , $\sqrt{2}a$ 인 원의 제1사분면(first quadrant)의 부분을 각각 R_1 과 R_2 라 하고 ($a^2 \leq x^2 + y^2 \leq 2a^2$), $R = \{(x, y) | 0 \leq x \leq a, 0 \leq y \leq a\}$ 를 정사각형이라 하자. 그럼 $R_1 \subseteq R \subseteq R_2$. 이다. 각 영역에서 같은 식의 적분값을 구한 후, $\textcolor{blue}{a}$ 를 무한대로 보내면서 샌드위치 성질로 원하는 $\{(x, y) | 0 \leq x \leq \infty, 0 \leq y \leq \infty\}$ 에서의 주어진 식의 적분값을 얻는 것이 전통적인 방법이다. 이를 위하여 전통적인 방법에서는 이중적분의 계산과 $\lim_{a \rightarrow \infty}$ 과정이 꼭 필요하였다. 그러나 코드를 이용하면)

$I = \int_0^\infty e^{-x^2} dx = \int_0^\infty e^{-y^2} dy$ 이므로 Fubini 정리에 의하여 $I^2 = \int_0^\infty \int_0^\infty e^{-(x^2+y^2)} dx dy$

이므로, $x = r\cos\theta$, $y = r\sin\theta$, $0 \leq \theta \leq \pi/2$, $r \geq 0$ 로 치환하여 극좌표로 바꾸면,

$x^2 + y^2 = (r\cos\theta)^2 + (r\sin\theta)^2 = r^2$ 이고 $I^2 = \int_0^{\frac{\pi}{2}} \int_0^\infty e^{-r^2} r dr d\theta$ 된다. $r^2 = t$ 로 치환하

면, $2rdr = dt$ 이므로 $I^2 = \int_0^{\frac{\pi}{2}} \left(\frac{1}{2} \int_0^\infty e^{-t} dt \right) d\theta$ 을 얻는다. (여기서 아래와 같이

코드를 이용하여 적분값을 바로 얻으므로) $\int_0^\infty e^{-t} dt = 1$ 이고 $I^2 = \frac{1}{2} \int_0^{\frac{\pi}{2}} d\theta = \frac{\pi}{4} \Rightarrow$

$$I = \frac{\sqrt{\pi}}{2} \text{ 이다.} \quad \blacksquare$$

<http://sage.skku.edu/> 또는 <https://sagecell.sagemath.org/>

```
var('t')      # http://sage.skku.edu/ 또는 https://sagecell.sagemath.org/
h=exp(-t)
integral(h, t, 0, infinity)
```

답: 1 # $\int_0^\infty e^{-t} dt = 1$

```

var('x, y')
f=exp(-x^2-y^2)
integral(integral(f, y, 0, infinity), x, 0, infinity)

```

$$\text{답: } 1/4\pi \quad \# \quad I^2 = \frac{1}{2} \int_0^{\frac{\pi}{2}} d\theta = \frac{\pi}{4} \quad \text{따라서 } I = \int_0^{\infty} e^{-x^2} dx = \int_0^{\infty} e^{-y^2} dy = \frac{\sqrt{\pi}}{2} \quad \blacksquare$$

[Note]

<http://matrix.skku.ac.kr/Cal-Book1/Ch14/>
<http://matrix.skku.ac.kr/cal-book/part2/CS-Sec-14-2-Sol.html>

예제 20 영역 $D = [-1, 1] \times [-1, 2] \times [-1, 3]$ 에서 함수 $w = x - 2y + 4z$ 가 주어져 있을 때 다음 3중적분을 구하여라.

$$\iiint_D (x - 2y + 4z) dx dy dz .$$

풀이. $\int_{-1}^3 \int_{-1}^2 \int_{-1}^1 (x - 2y + 4z) dx dy dz = 72.$

<http://sage.skku.edu/> 또는 <https://sagecell.sagemath.org/>

```

var('x, y, z')      # http://sage.skku.edu/ 또는 https://sagecell.sagemath.org/
f= x - 2*y + 4*z
integral(integral( integral(f, x, -1, 1), y, -1, 2), z, -1, 3)

```

$$\text{답 } 72 \quad \# \quad \int_{-1}^3 \int_{-1}^2 \int_{-1}^1 (x - 2y + 4z) dx dy dz = 72. \quad \blacksquare$$

<http://matrix.skku.ac.kr/Cal-Book/part2/CS-Sec-14-5-Sol.html>

중적분은 [Part 3 확률통계와 빅데이터]에서 다룰 결합확률분포의 계산에도 많이 쓰인다. 아래의 예제를 살펴보자.

예제 21 연속 확률변수 X, Y 의 결합밀도함수 $f(x, y)$ 가 다음과 같이 주어져 있다
고 하자.

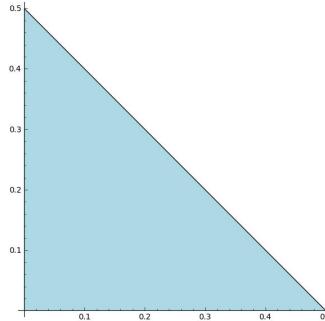
$$f(x, y) = \frac{1}{9}e^{-\frac{x+y}{3}}, \quad x, y \geq 0$$

$P(X+Y \leq \frac{1}{2})$ 일 확률을 구하여라.

풀이. 구하고자 하는 확률은 다음과 같다.

$$P(X+Y \leq \frac{1}{2}) = \iint_R f(x, y) dx dy, \quad f(x, y) = \frac{1}{9}e^{-\frac{x+y}{3}}$$

여기서 적분영역 $R = \left\{(x, y) \mid x+y \leq \frac{1}{2}\right\} = \left\{(x, y) \mid 0 \leq x \leq \frac{1}{2}, 0 \leq y \leq \frac{1}{2}-x\right\}$ 은 다음과 같다.



[Sage code] <http://mathlab.knou.ac.kr:8080/> <http://sage.skku.edu/>

```
var('x, y')
f(x, y) = 1/9*exp(-(x + y)/3)
print(integral(integral(f(x, y), y, 0, 1/2 - x), x, 0, 1/2))
print((integral(integral(f(x, y), y, 0, 1/2 - x), x, 0, 1/2)).n(digits=5))
```

$$-7/6 \cdot e^{(-1/6)} + 1$$

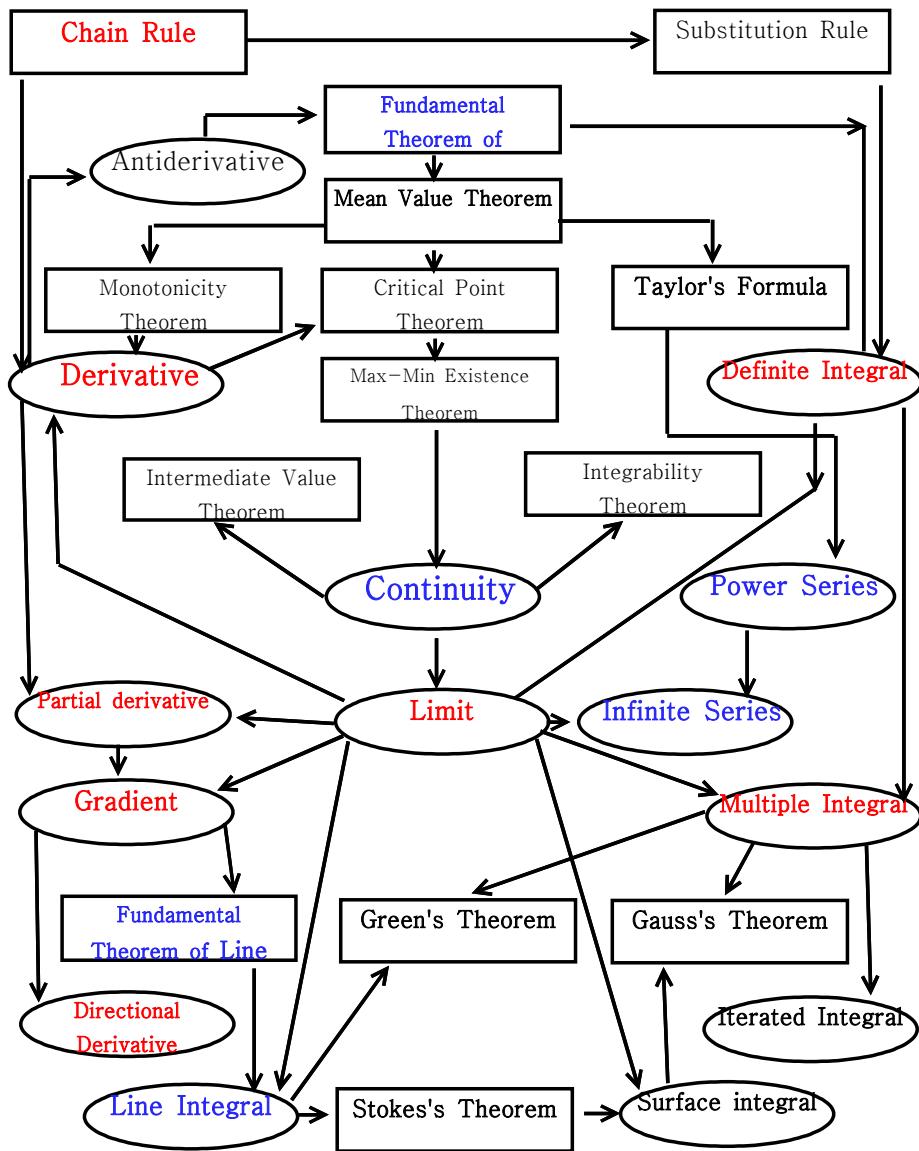
답. 0.012439 ■

Part2 과제: https://youtu.be/qxeal3Z_mG1g, 중간고사 답안 : https://youtu.be/S3xT1_SXH2c

[부록 1] 미적분학의 상호연관성(Calculus Map)

<http://matrix.skku.ac.kr/Calculus-Map/> (클릭하여 직접 각 절의 내용을 보세요)

in <http://matrix.skku.ac.kr/Cal-Book1/>



[미적분학 1, 2 지식을 더 알고 싶은 학생은 저자의 아래 동영상 강의를 참고하면 된다.]

Calculus (미분적분학) ○ 문제풀이 <http://matrix.skku.ac.kr/cal-book/>

○ 전자 교재와 강의록 Contents

<http://matrix.skku.ac.kr/Cal-Book1/Ch1/>
<http://matrix.skku.ac.kr/Cal-Book1/Ch2/>
<http://matrix.skku.ac.kr/Cal-Book1/Ch3/>
<http://matrix.skku.ac.kr/Cal-Book1/Ch4/>
<http://matrix.skku.ac.kr/Cal-Book1/Ch5/>
<http://matrix.skku.ac.kr/Cal-Book1/Ch6/>
<http://matrix.skku.ac.kr/Cal-Book1/Ch7/>
<http://matrix.skku.ac.kr/Cal-Book1/Ch8/>
<http://matrix.skku.ac.kr/Cal-Book1/Ch9/>
<http://matrix.skku.ac.kr/Cal-Book1/Ch10/>
<http://matrix.skku.ac.kr/Cal-Book1/Ch11/>
<http://matrix.skku.ac.kr/Cal-Book1/Ch12/>
<http://matrix.skku.ac.kr/Cal-Book1/Ch13/>
<http://matrix.skku.ac.kr/Cal-Book1/Ch14/>
<http://matrix.skku.ac.kr/Cal-Book1/Ch15/>

○ 실습실 <http://matrix.skku.ac.kr/Lab-Book/Sage-Lab-Manual-1.htm>

○ PBL 보고서

Calculus 1 <http://matrix.skku.ac.kr/PBL/>
Calculus 2 <http://matrix.skku.ac.kr/PBL2/>
<http://matrix.skku.ac.kr/Cal-Book1/Calculus-1/index.htm>
<http://matrix.skku.ac.kr/Cal-Book1/Calculus-2/index.htm>

[출처] <http://matrix.skku.ac.kr/2019-album/>

[부록 2] 미적분학용 Sage/Python 코드

<http://matrix.skku.ac.kr/Cal-Book/Appnd/index.htm>

<http://sage.skku.edu/>

<http://matrix.skku.ac.kr/Mobile-Sage-G/sage-grapher.html>

Define Function	<code>f(x)=exp(-2*x)</code>
Taylor expansion	<code>f(x).taylor(x, 2, 4)</code>
Equation solving	<code>solve(f(x)==0, x)</code>
Approximate solution	<code>find_root(f(x), a, b)</code>
Partial fraction	<code>f.partial_fraction(x)</code>
Limit	<code>limit(f(x), x=2)</code>
Right hand limit	<code>limit(f(x), x=2, dir='+')</code>
Left hand limit	<code>limit(f(x), x=2, dir=' -')</code>
Limit at + infinity	<code>limit(f(x), x=+oo)</code>
Limit at - infinity	<code>limit(f(x), x=-oo)</code>
Derivative	<code>diff(f(x), x)</code>
Second derivative	<code>diff(f(x), x, 2)</code>
Indefinite integral	<code>integral(f(x), x)</code>
Definite integral	<code>integral(f(x), (x, -2, 3))</code>
Series	<code>sum((e/10)^x, x, 1, +oo)</code>
Radius of convergence	<code>u(x)=1/(x*2^x) rho=limit(abs(u(x+1)/u(x)), x=+oo) 1/rho</code>
Define a vector	<code>x = vector([3, -4, 5])</code>
Norm of the vector	<code>x.norm()</code>
Distance from yz-plane	<code>sqrt(x[0]^2)</code>
Distance from zx-plane	<code>sqrt(x[1]^2)</code>
Distance from xz-plane	<code>sqrt(x[2]^2)</code>
Distance from x-axis	<code>sqrt(x[1]^2 + x[2]^2)</code>
Distance from y-axis	<code>sqrt(x[0]^2 + x[2]^2)</code>
Distance from z-axis	<code>sqrt(x[0]^2 + x[1]^2)</code>
Define a vector valued function	<code>var('t'); r=vector([2-5*t,4*t,1+3*t])</code>
Velocity	<code>v=diff(r, t)</code>
Acceleration	<code>a=diff(r, t, 2)</code>
Speed	<code>v.norm()</code>
Velocity vector at t = 2	<code>v.subs(t=2)</code>
Acceleration vector at t = 2	<code>a.subs(t=2)</code>
Arc length	<code>integral(v.norm(), (t, 0, 5))</code>
Dot product	<code>v.dot_product(a)</code>
Cross product	<code>v.cross_product(a)</code>

Find gradient

```
var('x, y, z')
f=x*y+y*z^2+x*z^3
f.gradient()
integral(integral(f, (x, 0, y)), (y, 0, 1))
integral(integral(integral(f, (x, 0, y)), (y, 0, 1)), (z, 0, 1))
```

Double integral
Triple integral

```
var('r, t')
f=arctan(tan(t))
integral(integral(f*r, (r, 1, 2)), (t, 0, pi/4))
```

Double integral
(in Polar coordinate)

```
T = Cylindrical('height', ['radius', 'azimuth'])
T.transform(radius=1, azimuth= pi, height=2)
```

Cylindrical
to Rectangular

```
T = Spherical('radius', ['azimuth', 'inclination'])
T.transform(radius=3, azimuth=pi/6, inclination=pi/6)
```

Spherical
to Rectangular

```
plot(sin(x), (x, -4, 4))
parametric_plot((sin(x), cos(x)), (x, 0, 2*pi))
var('x, y'); implicit_plot(sin(x)-y==1, (x, -2, 2), (y, -2, 2))
line([(1, 1), (2, 2)], color='red')
```

Graph in plane
Parametric function in plane
Implicit function in plane
Line segment in plane

```
var('x, y'); plot(x^2+y^2, (x, -2, 2), (y, -2, 2))
parametric_plot3d((sin(x), cos(x), x), (x, 0, 2*pi))
var('x, y, z'); implicit_plot3d(x^2+y^2==5, (x, -3, 3), (y, -3, 3), (z, -3, 3))
```

Graph in \mathbb{R}^3
Parametric function in
Implicit function in \mathbb{R}^3

```
contour_plot(x^2+y^2, (x, -1, 1), (y, -1, 1), cmap='hsv',
labels=True)
```

Level curve

```
plot_vector_field((x+y, x), (x, -3, 3), (y, -3, 3))
```

Vector field
Vector field in \mathbb{R}^3

```
plot_vector_field3d((0, 0, 1), (x, -3, 3), (y, -3, 3), (z, -3, 3))
```

Line integral

```
var('t'); r=vector([t-t^3, t^2, 0])
integral(r.dot_product(diff(r, t)), (t, 0, pi))
```

curl (A)
div (A)

```
A(x,y,z) = P*i+Q*j+R*k # conservative if curl(F)=0.
curlA=(diff(R,y)-diff(Q,z))*i+(diff(P,z)-diff(R,x))*j+(diff(Q,x)-diff(P,y))*k
```

```
divA = diff(P,x)+diff(Q,y)+diff(R,z) # divergence of A
```

Green's theorem
Stokes' theorem
Divergence theorem

```
integral(integral((diff(N, x)-diff(M, y))*r, (r, 0, 3)), (t, 0, 2*pi))
integral(integral(curl(F).dot_product(-n), (r, 0, 1)), (t, 0, 2*pi))
integral(integral(integral(Div, 0, 3), (y, 0, 2)), (z, 0, 1))
```

[부록 3] 미적분학 공식과 표(Table)

■ Rules for Inequalities

1. If $a < b$, $b < c$, then $a < c$.
2. If $a < b$, then $a + c < b + c$, $a - c < b - c$.
3. If $a < b$, $c > 0$, then $ac < bc$, $\frac{a}{c} < \frac{b}{c}$.
4. If $a < b$, then $-b < -a$.

■ Special Functions

1. Exponential Functions

If $a > 0$ and $a \neq 1$, then a function of the form $f(x) = a^x$ is called an exponential function.

The number a is called the base and x is called the exponential.

<http://matrix.skku.ac.kr/Mobile-Sage-G/sage-grapher.html>

2. Logarithmic Functions

The logarithmic with base $a > 0$ and $a \neq 1$, is defined by $y = \log_a x$.

3. Hyperbolic Functions

$$\sinh x = \frac{e^x - e^{-x}}{2} \quad \operatorname{csch} x = \frac{1}{\sinh x}$$

$$\cosh x = \frac{e^x + e^{-x}}{2} \quad \operatorname{sech} x = \frac{1}{\cosh x}$$

$$\tanh x = \frac{\sinh x}{\cosh x} \quad \operatorname{coth} x = \frac{\cosh x}{\sinh x}$$

■ Formulas of Trigonometric Functions

1. Addition and Subtraction Formulas

$$\sin(x \pm y) = \sin x \cos y \pm \cos x \sin y$$

$$\cos(x \pm y) = \cos x \cos y \mp \sin x \sin y$$

$$\tan(x \pm y) = \frac{\tan x \pm \tan y}{1 \mp \tan x \tan y}$$

2. Double Angle Formulas

$$\sin 2x = 2\sin x \cos x$$

$$\cos 2x = \cos^2 x - \sin^2 x$$

$$\tan 2x = \frac{2\tan x}{1 - \tan^2 x}$$

3. Triple Angle Formulas

$$\sin 3x = 3\sin x - 4\sin^3 x$$

$$\cos 3x = 4\cos^3 x - 3\cos x$$

$$\tan 3x = \frac{3\tan x - \tan^3 x}{1 - 3\tan^2 x}$$

4. Product-to-Sum

$$\sin x \cos y = \frac{1}{2} \{ \sin(x+y) + \sin(x-y) \}$$

$$\cos x \sin y = \frac{1}{2} \{ \sin(x+y) - \sin(x-y) \}$$

$$\cos x \cos y = \frac{1}{2} \{ \cos(x+y) + \cos(x-y) \}$$

$$\sin x \sin y = -\frac{1}{2} \{ \cos(x+y) - \cos(x-y) \}$$

5. Sum-to-Product

$$\sin A + \sin B = 2 \sin \frac{A+B}{2} \cos \frac{A-B}{2}$$

$$\sin A - \sin B = 2 \cos \frac{A+B}{2} \sin \frac{A-B}{2}$$

$$\cos A + \cos B = 2 \cos \frac{A+B}{2} \cos \frac{A-B}{2}$$

$$\cos A - \cos B = -2 \sin \frac{A+B}{2} \sin \frac{A-B}{2}$$

■ Limits

<http://matrix.skku.ac.kr/cal-lab/SKKU-Cell-Epsilon-Delta.html>

<http://matrix.skku.ac.kr/cal-lab/cal-Limit.html>

1. Limit Laws

Suppose that the limits $\lim_{x \rightarrow a} f(x)$ and $\lim_{x \rightarrow a} g(x)$ exist and c is a constant. Then

$$\lim_{x \rightarrow a} [f(x) + g(x)] = \lim_{x \rightarrow a} f(x) + \lim_{x \rightarrow a} g(x)$$

$$\lim_{x \rightarrow a} [f(x) - g(x)] = \lim_{x \rightarrow a} f(x) - \lim_{x \rightarrow a} g(x)$$

$$\lim_{x \rightarrow a} c f(x) = c \lim_{x \rightarrow a} f(x)$$

$$\lim_{x \rightarrow a} \sqrt[n]{f(x)} = \sqrt[n]{\lim_{x \rightarrow a} f(x)}, \text{ where } n \text{ is a positive integer.}$$

(If n is even, we require $\lim_{x \rightarrow a} f(x) \geq 0$)

<http://matrix.skku.ac.kr/cal-lab/cal-2-1-7.html>

<http://matrix.skku.ac.kr/cal-lab/cal-2-2-3.html>

2. Squeeze Theorem (or Sandwich Theorem)

If $f(x) \leq g(x) \leq h(x)$, when x is near a and

$$\lim_{x \rightarrow a} f(x) = \lim_{x \rightarrow a} h(x) = L, \text{ then } \lim_{x \rightarrow a} g(x) = L.$$

<http://matrix.skku.ac.kr/cal-lab/cal-2-1-12.html>

■ Derivatives

1. Differentiation Rules

$$\frac{d}{dx}(cx^n) = ncx^{n-1} \quad \text{where } n \text{ is a positive integer and } c \text{ is a constant.}$$

$$\frac{d}{dx}[cf(x)] = c \frac{d}{dx}f(x)$$

$$\frac{d}{dx}[f(x) + g(x)] = \frac{d}{dx}f(x) + \frac{d}{dx}g(x)$$

$$\frac{d}{dx}[f(x) - g(x)] = \frac{d}{dx}f(x) - \frac{d}{dx}g(x)$$

$$\frac{d}{dx}(f(x)g(x)) = f(x)\frac{d}{dx}(g(x)) + g(x)\frac{d}{dx}(f(x))$$

$$\left\{\frac{f(x)}{g(x)}\right\}' = \frac{f'(x)g(x) - f(x)g'(x)}{\{g(x)\}^2}$$

2. Chain Rule

If n is any real number and $u = g(x)$ is differentiable and $y = u^n$, then

$$\frac{dy}{dx} = n[g(x)]^{n-1} \cdot g'(x).$$

3. Parametric Formula

For the parametric equations: $x = f(t)$ and $y = g(t)$,

$$\frac{dy}{dx} = \frac{dy/dt}{dx/dt}, \quad \frac{dx}{dt} \neq 0$$

4. Implicit Function

Let $F(x, y) = 0$ be an implicit function.

$$D_x F(x, y) dx + D_y F(x, y) dy = 0$$

$$\frac{dy}{dx} = -\frac{D_x F(x, y)}{D_y F(x, y)} \text{ where } D_y F(x, y) \neq 0.$$

5. Inverse Function

$$(f^{-1})'(x) = \frac{1}{f'(f^{-1}(x))} \quad \text{or} \quad \frac{dy}{dx} = \frac{1}{dx/dy}$$

6. Trigonometric Functions

$$\frac{d}{dx}(\sin x) = \cos x \quad \frac{d}{dx}(\csc x) = -\csc x \cot x$$

$$\frac{d}{dx}(\cos x) = -\sin x \quad \frac{d}{dx}(\sec x) = \sec x \tan x$$

$$\frac{d}{dx}(\tan x) = \sec^2 x \quad \frac{d}{dx}(\cot x) = -\csc^2 x$$

<http://matrix.skku.ac.kr/cal-lab/cal-3-3-Exm24.html>

7. Inverse Trigonometric Functions

$$\frac{d}{dx}(\sin^{-1} x) = \frac{1}{\sqrt{1-x^2}} \quad \frac{d}{dx}(\csc^{-1} x) = -\frac{1}{x\sqrt{x^2-1}}$$

$$\frac{d}{dx}(\cos^{-1} x) = -\frac{1}{\sqrt{1-x^2}} \quad \frac{d}{dx}(\sec^{-1} x) = \frac{1}{x\sqrt{x^2-1}}$$

$$\frac{d}{dx}(\tan^{-1} x) = -\frac{1}{1+x^2} \quad \frac{d}{dx}(\cot^{-1} x) = -\frac{1}{1+x^2}$$

<http://matrix.skku.ac.kr/cal-lab/cal-3-2-13.html>

8. Logarithmic Functions

$$\frac{d}{dx}(\ln x) = \frac{1}{x} \quad \frac{d}{dx}(\log_a x) = \frac{1}{x \ln a}$$

$$\frac{d}{dx}[\ln f(x)] = \frac{f'(x)}{f(x)} \quad \frac{d}{dx}[\log_a f(x)] = \frac{f'(x)}{f(x) \ln a}$$

9. Exponential Functions

$$\frac{d}{dx}(e^x) = e^x \quad \frac{d}{dx}(a^x) = a^x \ln a$$

$$\frac{d}{dx}(e^{f(x)}) = f'(x)e^{f(x)} \quad \frac{d}{dx}(a^{f(x)}) = f'(x)a^{f(x)} \ln a$$

10. Hyperbolic Functions

$$\frac{d}{dx}(\sinh x) = \cosh x \quad \frac{d}{dx}(\coth x) = -\operatorname{csch}^2 x$$

$$\frac{d}{dx}(\cosh x) = \sinh x \quad \frac{d}{dx}(\sech x) = -\operatorname{sech} x \tanh x$$

$$\frac{d}{dx}(\tanh x) = \operatorname{sech}^2 x \quad \frac{d}{dx}(\operatorname{csch} x) = -\operatorname{csch} x \coth x$$

11. Inverse Hyperbolic Functions

$$\frac{d}{dx}(\sinh^{-1} x) = \frac{1}{\sqrt{1+x^2}} \quad (x \in R)$$

$$\frac{d}{dx}(\cosh^{-1} x) = \frac{1}{\sqrt{x^2-1}} \quad (x > 1)$$

$$\frac{d}{dx}(\tanh^{-1} x) = \frac{1}{1-x^2} \quad (|x| < 1)$$

■ General Rules of Integration

<http://matrix.skku.ac.kr/cal-lab/cal-RiemannSum.html>

<http://matrix.skku.ac.kr/cal-lab/cal-6-3-17.html>

<http://matrix.skku.ac.kr/cal-lab/cal-12-1-Rotations-B.html>

1. Basic Forms

$$\int a dx = ax + C$$

$$\int x^n dx = \frac{x^{n+1}}{n+1} + C \quad (n \neq -1)$$

$$\int \frac{1}{x} dx = \ln|x| + C$$

$$\int e^x dx = e^x + C$$

<http://matrix.skku.ac.kr/cal-lab/cal-7-7-Exm-8.html>

$$\int a^x dx = \frac{a^x}{\ln a} + C, \quad a > 0$$

<http://matrix.skku.ac.kr/cal-lab/cal-5-2-20.html>

$$\int \frac{1}{\sqrt{x^2+A}} dx = \ln|x + \sqrt{x^2+A}| + C \quad (A \neq 0)$$

2. Trigonometric Forms

<http://matrix.skku.ac.kr/cal-lab/cal-5-4-exm-7.html>

<http://matrix.skku.ac.kr/cal-lab/cal-8-3-3.html>

$$\int \sin x dx = -\cos x + C$$

$$\int \cos x dx = \sin x + C$$

$$\int \sin^2 x dx = \frac{x}{2} - \frac{\sin 2x}{4} + C$$

$$\int \cos^2 x dx = \frac{x}{2} + \frac{\sin 2x}{4} + C$$

3. Hyperbolic Forms

$$\int \sinh ax dx = \frac{1}{a} \cosh ax + C$$

$$\int \cosh ax dx = \frac{1}{a} \sinh ax + C$$

$$\int \tanh ax dx = \frac{1}{a} \ln(\cosh ax) + C$$

<http://matrix.skku.ac.kr/cal-lab/cal-8-1-9.html>

4. Forms Involving $a^2 \pm x^2$

$$\int \frac{1}{x^2 - a^2} dx = \frac{1}{2a} \ln \left| \frac{x-a}{x+a} \right| + C \quad (a \neq 0)$$

$$\int \frac{1}{a^2 - x^2} dx = \frac{1}{2a} \ln \left| \frac{x+a}{x-a} \right| + C \quad (a \neq 0)$$

$$\int \frac{1}{\sqrt{a^2 - x^2}} dx = \sin^{-1} \left(\frac{x}{a} \right) + C \quad (a \neq 0)$$

5. Inverse Trigonometric Forms

$$\int \sin^{-1} x dx = x \sin^{-1} x + \sqrt{1-x^2} + C$$

$$\int \cos^{-1} x dx = x \cos^{-1} x - \sqrt{1-x^2} + C$$

$$\int \tan^{-1} x dx = x \tan^{-1} x - \frac{1}{2} \ln(1+x^2) + C$$

http://myhandbook.info/form_integ.html

■ Series

1. Taylor Series

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!} (x-a)^n = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \dots$$

<http://matrix.skku.ac.kr/cal-lab/cal-10-5-Exm-11.html>

2. Maclaurin Series

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(0)}{n!} x^n = f(0) + \frac{f'(0)}{1!}x + \frac{f''(0)}{2!}x^2 + \dots$$

$$\frac{1}{1-x} = \sum_{n=0}^{\infty} x^n = 1 + x + x^2 + x^3 + \dots, \quad (-1, 1)$$

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots, \quad (-\infty, \infty)$$

$$\sin x = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n+1}}{(2n+1)!} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots, \quad (-\infty, \infty)$$

$$\cos x = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n}}{(2n)!} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots, \quad (-\infty, \infty)$$

$$\tan^{-1} x = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n+1}}{2n+1} = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \dots, \quad [-1, 1]$$

3. Binomial Series

If a and b are any real numbers and k is a positive integer, we have

$$(a+b)^k = \sum_{n=0}^k \binom{k}{n} a^{k-n} b^n = a^k + k a^{k-1} b + \frac{k(k-1)}{2!} a^{k-2} b^2 + \frac{k(k-1)(k-2)}{3!} a^{k-3} b^3 + \dots + \frac{k(k-1)(k-2) \dots (k-n+1)}{n!} a^{k-n} b^n + \dots + k a b^{k-1} + b^k$$

where $\binom{k}{0} = 1$ and $\binom{k}{n} = \frac{k(k-1)(k-2) \dots (k-n+1)}{n!}$, $1 \leq n \leq k$.

■ Vectors

1. Dot Product

Let $\mathbf{a} = \langle a_1, a_2, \dots, a_n \rangle$ and $\mathbf{b} = \langle b_1, b_2, \dots, b_n \rangle$.

$$\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2 + \dots + a_n b_n = \sum_{i=1}^n a_i b_i$$

<http://matrix.skku.ac.kr/cal-lab/cal-11-3-2.html>

2. Projections

Scalar projection of \mathbf{b} onto \mathbf{a} : $\text{comp}_{\mathbf{a}} \mathbf{b} = \frac{\mathbf{a} \cdot \mathbf{b}}{|\mathbf{a}|}$,

Vector projection of \mathbf{b} onto \mathbf{a} : $\text{proj}_{\mathbf{a}} \mathbf{b} = \left(\frac{\mathbf{a} \cdot \mathbf{b}}{|\mathbf{a}|} \right) \frac{\mathbf{a}}{|\mathbf{a}|} = \frac{\mathbf{a} \cdot \mathbf{b}}{|\mathbf{a}|^2} \mathbf{a}$

<http://matrix.skku.ac.kr/cal-lab/cal-11-5-20.html>

3. Definition and Properties of Cross Product

<http://matrix.skku.ac.kr/cal-lab/cal-11-4-Exs-6.html>

$$\mathbf{a} \times \mathbf{b} = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{vmatrix} = \begin{vmatrix} a_2 & a_3 \\ b_2 & b_3 \end{vmatrix} \mathbf{i} - \begin{vmatrix} a_1 & a_3 \\ b_1 & b_3 \end{vmatrix} \mathbf{j} + \begin{vmatrix} a_1 & a_2 \\ b_1 & b_2 \end{vmatrix} \mathbf{k}$$

Let two nonzero vectors \mathbf{a} and \mathbf{b} are two sides of a parallelogram,

then the area of the parallelogram is $A = \frac{1}{2} |\mathbf{a} \times \mathbf{b}|$.

<http://matrix.skku.ac.kr/cal-lab/cal-11-4-10.html>

4. Rules of Limits

$$\lim_{t \rightarrow a} c\mathbf{u}(t) = c \lim_{t \rightarrow a} \mathbf{u}(t)$$

$$\lim_{t \rightarrow a} [\mathbf{u}(t) + \mathbf{v}(t)] = \lim_{t \rightarrow a} \mathbf{u}(t) + \lim_{t \rightarrow a} \mathbf{v}(t)$$

$$\lim_{t \rightarrow a} \{f(t)\mathbf{u}(t)\} = \lim_{t \rightarrow a} f(t) \lim_{t \rightarrow a} \mathbf{u}(t)$$

$$\lim_{t \rightarrow a} [\mathbf{u}(t) \cdot \mathbf{v}(t)] = \lim_{t \rightarrow a} \mathbf{u}(t) \cdot \lim_{t \rightarrow a} \mathbf{v}(t)$$

$$\lim_{t \rightarrow a} [\mathbf{u}(t) \times \mathbf{v}(t)] = \lim_{t \rightarrow a} \mathbf{u}(t) \times \lim_{t \rightarrow a} \mathbf{v}(t)$$

$$\lim_{s \rightarrow s_0} \mathbf{u}(g(s)) = \lim_{t \rightarrow a} \mathbf{u}(t), \quad \lim_{s \rightarrow s_0} g(s) = a$$

<http://matrix.skku.ac.kr/cal-lab/9-5-Example-7.html>

5. Rules of Differentiation

http://myhandbook.info/form_diff.html

$$\frac{d}{dt} [\mathbf{u}(t) + \mathbf{v}(t)] = \mathbf{u}'(t) + \mathbf{v}'(t)$$

$$\frac{d}{dt} [c\mathbf{u}(t)] = c\mathbf{u}'(t) \text{ where } c \text{ is a scalar}$$

$$\frac{d}{dt} [f(t)\mathbf{u}(t)] = f'(t)\mathbf{u}(t) + f(t)\mathbf{u}'(t)$$

$$\frac{d}{dt} [\mathbf{u}(t) \cdot \mathbf{v}(t)] = \mathbf{u}'(t) \cdot \mathbf{v}(t) + \mathbf{u}(t) \cdot \mathbf{v}'(t)$$

$$\frac{d}{dt} [\mathbf{u}(f(t))] = f'(t)\mathbf{u}'(f(t))$$

<http://matrix.skku.ac.kr/cal-lab/cal-4-3-24.html>

<http://matrix.skku.ac.kr/cal-lab/cal-4-2-9.html>

6. Derivative of a Vector Function

If $\mathbf{r}(t) = \langle f(t), g(t), h(t) \rangle$, then $\mathbf{r}'(t) = \langle f'(t), g'(t), h'(t) \rangle$

<http://matrix.skku.ac.kr/cal-lab/cal-13-2-Exm6.html>

<http://matrix.skku.ac.kr/cal-lab/cal-12-4-3.html>

<http://matrix.skku.ac.kr/cal-lab/cal-13-2-20.html>

7. Integral of a Vector Function

If $\mathbf{r}(t) = \langle f(t), g(t), h(t) \rangle$,

$$\text{then } \int \mathbf{r}(t) dt = \left\langle \int f'(t) dt, \int g'(t) dt, \int h'(t) dt \right\rangle$$

8. Arc Length

$$L = \int_a^b |\mathbf{r}'(t)| dt$$

<http://matrix.skku.ac.kr/cal-lab/cal-8-1-9.html>

<http://matrix.skku.ac.kr/cal-lab/cal-8-1-11.html>

<http://matrix.skku.ac.kr/cal-lab/cal-13-3-2.html>

9. Curvature

$$\kappa(t) = \frac{|\mathbf{r}'(t) \times \mathbf{r}''(t)|}{|\mathbf{r}'(t)|^3}, \quad \kappa(x) = \frac{|f''(x)|}{[1 + (f'(x))^2]^{3/2}}$$

<http://matrix.skku.ac.kr/cal-lab/cal-13-3-12.html>

10. Equations of Line

$$\mathbf{r} = \mathbf{r}_0 + t\mathbf{v} \quad (\mathbf{r} = \overrightarrow{OP}, \mathbf{r}_0 = \overrightarrow{OP}) \quad : \text{a vector equation}$$

$$\begin{cases} x = x_0 + ta \\ y = y_0 + tb \quad (t \in \mathbb{R}) \\ z = z_0 + tc \end{cases} \quad : \text{parametric equations}$$

<http://matrix.skku.ac.kr/cal-lab/11-5-Exmaple-7.html>

$$\frac{x - x_0}{a} = \frac{y - y_0}{b} = \frac{z - z_0}{c} \quad : \text{a symmetric equation}$$

<http://matrix.skku.ac.kr/cal-lab/11-5-Exmaple-14.html>

11. Equation of Plane

$$a(x - x_0) + b(y - y_0) + c(z - z_0) = 0 \quad : \text{a standard form of a plane}$$

$$\mathbf{r} = \mathbf{r}_0 + t_1\mathbf{v}_1 + t_2\mathbf{v}_2 \quad : \text{a vector version of a plane}$$

$$x = x_0 + a_1t_1 + a_2t_2$$

$$y = y_0 + b_1t_1 + b_2t_2, \quad (t_1, t_2 \in \mathbb{R}) \quad : \text{parametric equations of a plane}$$

$$z = z_0 + c_1t_1 + c_2t_2$$

<http://matrix.skku.ac.kr/cal-lab/cal-11-5-20.html>

■ Formulas of Vector Calculus

http://en.wikipedia.org/wiki/Vector_calculus_identities

1. Line Integral

$$\int_C f(x, y) ds = \int_a^b f(x(t), y(t)) \sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2} dt$$

<http://matrix.skku.ac.kr/cal-lab/Sec15-2-1.html>

<http://matrix.skku.ac.kr/cal-lab/Sec15-4-Exs-1.html>

2. Path Independent Theorem

Let f be a potential function for $\mathbf{F} = M\mathbf{i} + N\mathbf{j}$.

For any piecewise smooth curve C from A and B ,

$$\int_C M dx + N dy = f(B) - f(A).$$

<http://matrix.skku.ac.kr/cal-lab/Sec15-2-Exm-1.html>

3. Area of a Plane Region

If D has a piecewise smooth boundary C with positive orientation, then the area of D is

$$A = \int_C x dy - \int_C y dx = \frac{1}{2} \int_C (xdy - ydx)$$

4. Area of a Surface

$$A(S) = \iint_R \left| \frac{\partial \mathbf{r}}{\partial u} \times \frac{\partial \mathbf{r}}{\partial v} \right| dA$$

<http://matrix.skku.ac.kr/cal-lab/cal-8-2-Exm-4.html>

<http://matrix.skku.ac.kr/cal-lab/cal-8-2-3.html>

<http://matrix.skku.ac.kr/cal-lab/cal-8-2-4.html>

<http://matrix.skku.ac.kr/cal-lab/cal-0-a.html>

5. Surface Integral

$$\iint_S f(x, y, z) dS = \iint_R f(x(u, v), y(u, v), z(u, v)) \left| \frac{\partial \mathbf{r}}{\partial u} \times \frac{\partial \mathbf{r}}{\partial v} \right| dA$$

<http://matrix.skku.ac.kr/cal-lab/cal-14-5-1.html>

<http://matrix.skku.ac.kr/cal-lab/Sec15-6-Exm-4.html>

<http://matrix.skku.ac.kr/cal-lab/Sec15-6-Exm-5.html>

<http://matrix.skku.ac.kr/cal-lab/Sec15-6-Exm-8.html>

<http://matrix.skku.ac.kr/cal-lab/Sec15-7-Exm-2.html>

<http://matrix.skku.ac.kr/cal-lab/cal-15-9-Exam-3.html>

6. Gradient

$$\nabla f = \frac{\partial f}{\partial x} \mathbf{i} + \frac{\partial f}{\partial y} \mathbf{j} + \frac{\partial f}{\partial z} \mathbf{k}$$

<http://matrix.skku.ac.kr/cal-lab/cal-12-2-2.html>

7. Divergence

$$\nabla \cdot \mathbf{F} = \frac{\partial F_1}{\partial x} + \frac{\partial F_2}{\partial y} + \frac{\partial F_3}{\partial z}$$

<http://matrix.skku.ac.kr/cal-lab/Sec15-5-Exm-3.html>

8. Curl

$$\nabla \times \mathbf{F} = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \\ F_1 & F_2 & F_3 \end{vmatrix}$$

<http://matrix.skku.ac.kr/cal-lab/Sec15-8-Exm-2.html>

9. Laplace Operator

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2}$$

<http://matrix.skku.ac.kr/cal-lab/cal-12-2-6.html>

10. Vector Triple Products

$$(\mathbf{u} \times \mathbf{v}) \cdot \mathbf{w} = (\mathbf{v} \times \mathbf{w}) \cdot \mathbf{u} = (\mathbf{w} \times \mathbf{u}) \cdot \mathbf{v}$$

$$\mathbf{u} \times (\mathbf{v} \times \mathbf{w}) = (\mathbf{u} \cdot \mathbf{w}) \mathbf{v} - (\mathbf{u} \cdot \mathbf{v}) \mathbf{w}$$

<http://matrix.skku.ac.kr/cal-lab/cal-11-4-16.html>

Theorems on Vector Calculus

1. Green's Theorem

$$\int_C M dx + N dy = \iint_R \left(\frac{\partial N}{\partial x} - \frac{\partial M}{\partial y} \right) dx dy$$

<http://matrix.skku.ac.kr/cal-lab/Sec15-4-Exm-1.html>

<http://matrix.skku.ac.kr/cal-lab/Sec15-4-Exs-10.html>

<http://matrix.skku.ac.kr/cal-lab/cal-15-2-10.html>

2. Stoke's Theorem

$$\int_C \mathbf{F} \cdot d\mathbf{r} = \iint_S \operatorname{curl} \mathbf{F} \cdot d\mathbf{S}$$

<http://matrix.skku.ac.kr/cal-lab/Sec15-5-Exm-3.html>

<http://matrix.skku.ac.kr/cal-lab/cal-14-7-4.html>

<http://matrix.skku.ac.kr/cal-lab/cal-15-5-5.html>

3. Divergence Theorem

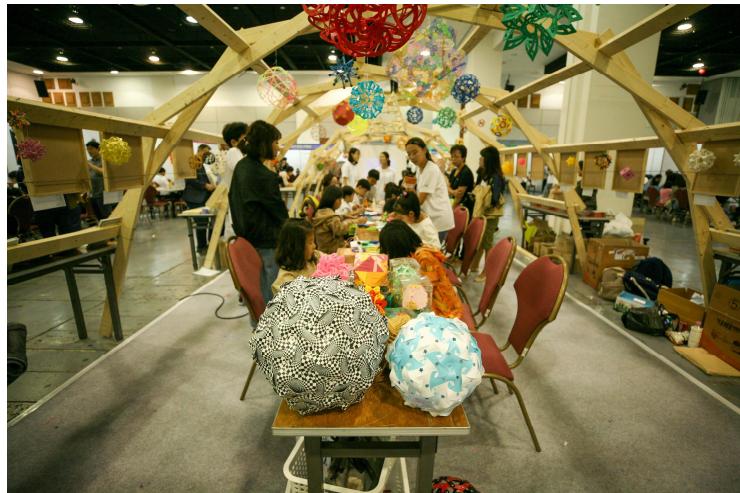
$$\iint_S \mathbf{A} \cdot \mathbf{n} dS = \iiint_V \nabla \cdot \mathbf{A} dV = \iiint_V \operatorname{div} \mathbf{A} dV.$$

<http://matrix.skku.ac.kr/cal-lab/Sec15-5-Exs-7.html>

<http://matrix.skku.ac.kr/cal-lab/cal-14-8-1.html>

<http://matrix.skku.ac.kr/cal-lab/cal-14-8-5.html>

<http://matrix.skku.ac.kr/cal-lab/cal-15-8-Exs-8.html>



2019년 9월 21일 제주 수학 축전

Part III

확률통계와 빅데이터

<http://matrix.skku.ac.kr/math4ai/part3/>

Basic
Mathematics
for Artificial
Intelligence

18강 동영상, 통계학과 R <https://youtu.be/u82BC1Rj0A> (11:51)

[참고도서] 최용석, [빅북총서008] R과 함께하는 통계학의 이해, BigBook, 2014.

<http://matrix.skku.ac.kr/e-math/>

<http://matrix.skku.ac.kr/2018-album/R-Sage-Stat-Lab-1.html>

<http://matrix.skku.ac.kr/2018-album/R-Sage-Stat-Lab-2.html>

The screenshot displays a presentation slide titled "Statistics Laboratory using Sage" by Jae Hwa Lee¹, Geung-Hee Lee² and Sang-Gu Lee³. The slide is divided into several sections:

- Abstract:** Describes a simple and effective model of a statistics laboratory using an open-source program, Sage, for free. It mentions that learners can easily design and modify it due to the simplicity of its structure.
- What the Lab Covers:** Lists basic computation (Sage commands), computation related to statistics, visualization (histogram, chart...), and browsing website (including java materials) using R commands.
- Problem:** Shows a screenshot of a Sage notebook with a histogram and a bar chart.
- R / Sage command:** Shows a screenshot of a Sage cell command input and output.
- Computation related to statistics:** Shows a screenshot of a Sage notebook with statistical calculations.
- Video Lecture:** Shows a screenshot of a video lecture interface.
- Java tools:** Shows screenshots of Java-based tools for statistics experiments.
- Internet resource:** Shows screenshots of various websites and resources.
- Conclusion:** Summarizes the introduction of a simple and effective model of a statistics laboratory using Sage.

[수학사] 통계학의 탄생

<https://youtu.be/3wh7RSbsicQ>

0. 통계학과 R

1. 순열, 조합, 확률
2. 확률변수
3. 확률분포
4. 데이터 활용의 실제

[부록 4] 기초통계 개념

0.1 통계학

- 데이터의 홍수 속에서 필요한 통계정보를 얻기 위해서는 수학지식 뿐만 아니라 데이터를 체계적으로 수집, 정리, 요약, 판단하는 데이터 과학의 기초인 통계학(statistics)이 필요하다.
- 통계학적 이론에 근거하여 데이터를 분석하려면 데이터를 처리하는 통계 프로그램이 필요하다. 대표적인 무료 통계 프로그램으로는 R 프로그램이 있다. <https://www.r-project.org/>에서 R 프로그램을 다운로드 받아 본인의 컴퓨터에 설치해도 되고, 앞서 설명한 SageMath 셀 <http://sage.skku.edu/>의 빈칸에 R 명령어를 직접 입력한 후, 셀 오른쪽 하단의 Language를 R로 지정하여 실행하면 결과가 나타난다.

0.2 R 명령어 예시

- SageMath 셀에서 직접 R 명령어를 수행해보자. SageMath 셀에서 R명령어로 그린 그래프를 직접 확인할 때는 dev.off() 명령어를 추가하면 된다. 아래는 몇 가지 사용 예시이다.

예제 1 어느 대학에서 통계학 수업을 수강하는 55명의 학생들을 대상으로 혈액형을 조사한 결과는 다음과 같다. 이 자료를 도수분포표, 원도표와 막대도표로 요약하라.

B	A	B	A	A	B	O	A	A	A	O
B	AB	B	AB	AB	A	A	O	AB	O	A
B	O	B	B	A	A	O	A	A	AB	B
B	O	B	B	B	A	AB	A	A	B	O
B	B	O	B	O	B	A	A	AB	A	A

[참고자료] 원 도표

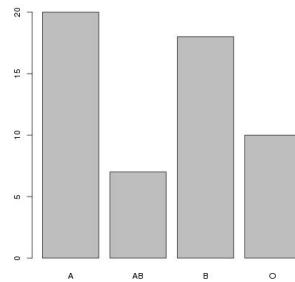
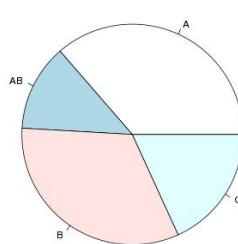
https://youtu.be/Eph_Y0BmHU0 http://en.wikipedia.org/wiki/Pie_chart

풀이. 아래와 같이 자료를 R 코드로 입력하여 실행하면 원하는 결과를 얻을 수 있다.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
blood = c("B", "A", "B", "A", "A", "B", "O", "A", "A", "A", "O", "B", "AB", "B",
"AB", "A", "A", "O", "AB", "O", "A", "B", "O", "B", "B", "A", "A", "O", "A",
"A", "AB", "B", "B", "O", "B", "B", "B", "A", "AB", "A", "A", "B", "O", "B",
"B", "O", "B", "O", "B", "A", "A", "AB", "A", "A")
cnt = table(blood)
prop = prop.table(cnt)
pie(cnt)          # 원도표
barplot(cnt)       # 막대도표
dev.off()
cbind(cnt, prop)  # 도수분포표
```

	cnt	prop
A	20	0.363636364
AB	7	0.1272727
B	18	0.3272727
O	10	0.1818182



[용어 소개]

- n 개의 표본자료를 x_1, x_2, \dots, x_n 이라 할 때 표본평균(sample mean) \bar{x} 는 다음과 같이 계산한다.

$$\bar{x} = \frac{x_1 + x_2 + \cdots + x_n}{n} = \frac{1}{n} \sum_{i=1}^n x_i$$

- n 개의 표본자료를 크기순으로 나열한 것을 x_1, x_2, \dots, x_n 이라 할 때 중위수(median)는

(i) 관측값의 개수(n)가 홀수면, 중위수는 $\frac{n+1}{2}$ 번째 관측값이다.

(ii) 관측값의 개수(n)가 짝수면, 중위수는 $\frac{n}{2}$ 번째 관측값과 $\frac{n}{2} + 1$ 번째 관측값의 평균이다.

- 사분위 수 : 4등분하는 위치의 수 (3개): Q1(제1사분위수, 제25백분위수), Q2(제2사분위수, 중위수, 제50백분위수), Q3(제3사분위수, 제75백분위수)

(평균) <http://en.wikipedia.org/wiki/Mean>

(중위수) <http://en.wikipedia.org/wiki/Median>

(표준편차) http://en.wikipedia.org/wiki/Standard_deviation

(분산) <http://en.wikipedia.org/wiki/Variance>

예제 2 어떤 교과목을 수강한 6명 학생들의 중간고사 성적은 다음과 같다. 학생들의 성적에 대한 표본평균(mean)과 중위수(median), 사분위수(quantile)를 구하여라.

85

72

91

83

76

84

풀이. 아래와 같이 자료를 R 코드로 입력하여 실행하면 원하는 결과를 얻을 수 있다.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```

gr = c(85, 72, 91, 83, 76, 84)
print(mean(gr))          # 표본의 평균
print(median(gr))        # 중위수
quantile(gr,type = 2)    # 4분위수, 4등분하는 위치의 수

```

```

[1] 81.83333
[1] 83.5
 0% 25% 50% 75% 100%
72.0 76.0 83.5 85.0 91.0  # 4분위수, 4등분하는 위치의 수(3개)
Min. 1st Qu. Median Mean 3rd Qu. Max.
72.00 77.75 83.50 81.83 84.75 91.00

```

예제 3 어떤 특정도로를 지나가는 차량의 교통소음을 측정한 값은 아래와 같다.

이 자료에 대한 기술통계량(Descriptive Statistics, 자료의 정보를 수치적으로 나타내는 다양한 방법. 예를 들어 중심위치의 측도는 주어진 자료가 어떤 값을 중심으로 분포되어 있는가를 나타내는 것이며, 산포의 측도는 자료들이 중심위치에서 얼마나 벗어나서 어느 방향으로 얼마나 치우쳐 있는가를 알 수 있게 한다. 구체적으로는 표본평균과 중위수, 사분위수(4등분하는 위치의 수), 중앙값, 최빈값, 분산, 표준편차, 도표, ... 등)을 구하여 데이터가 갖는 의미를 파악하고 개선을 위한 다양한 의견을 개진할 수 있다.

55.9	63.8	57.2	59.8	65.7	62.7	60.8	51.3	61.8	56.0
66.9	56.8	66.2	64.6	59.5	63.1	60.6	62.0	59.4	67.2
63.6	60.5	66.8	61.8	64.8	55.8	55.7	77.1	62.1	61.0
58.9	60.0	66.9	61.7	60.3	51.5	67.0	60.2	56.2	59.4
67.9	64.9	55.7	61.4	62.6	56.4	56.4	69.4	57.6	63.8

[참고자료]

- n 개의 표본자료를 x_1, x_2, \dots, x_n 이라 하고, 이들의 표본평균을 \bar{x} 라고 하면, 표본분산(sample variance) s^2 은 다음과 같다.

$$s^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2$$

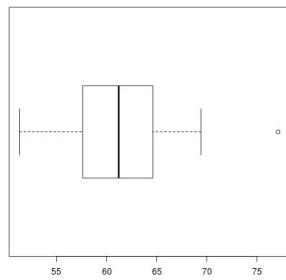
- 표본 표준편차(sample standard deviation)는 다음과 같다. $s = +\sqrt{s^2}$
- 관측값을 크기 순으로 정렬한 후, 관측값의 개수 n 에 0과 1 사이의 수 p 를 곱했을 때, (단, $p = (\frac{1}{100}, \frac{2}{100}, \dots, \frac{99}{100})$ 중 하나)
 - np 가 정수이면, 제 $100 \times p$ 백분위수(percentile)는 np 번째 관측값과 $np + 1$ 번째 관측값의 평균이다.
 - np 가 정수가 아니면, 제 $100 \times p$ 백분위수는 (np 의 정수부분에 1을 더한 값) 번째 관측값이다.
- 사분위(4등분하는 위치, Q1, Q2, Q3) 범위(inter-quartile range, IQR) $IQR = Q3 - Q1$

[풀이]. 아래와 같이 자료를 R 코드로 입력하여 실행하면 원하는 결과를 얻을 수 있다.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
noise = c(55.9, 63.8, 57.2, 59.8, 65.7, 62.7, 60.8, 51.3, 61.8,
       56.0, 66.9, 56.8, 66.2, 64.6, 59.5, 63.1, 60.6, 62.0,
       59.4, 67.2, 63.6, 60.5, 66.8, 61.8, 64.8, 55.8, 55.7,
       77.1, 62.1, 61.0, 58.9, 60.0, 66.9, 61.7, 60.3, 51.5,
       67.0, 60.2, 56.2, 59.4, 67.9, 64.9, 55.7, 61.4, 62.6,
       56.4, 56.4, 69.4, 57.6, 63.8)
print(mean(noise))          # 표본의 평균
print(var(noise))           # 분산
print(sd(noise))            # 표준편차
print(quantile(noise,type = 2)) # 사분위수
boxplot(noise, horizontal = T) # 상자그림
dev.off()
```

```
[1] 61.374                      # 표본의 평균
[1] 22.84972
[1] 4.780138                    # 표준편차
0% 25% 50% 75% 100%
51.3 57.6 61.2 64.6 77.1      # 4분위수, 4등분하는 위치의 수
```



0.3 학습할 내용

- 본 <인공지능을 위한 기초수학>에서는 다음의 웹사이트

[출처] <https://mingrammer.com/translation-the-mathematics-of-machine-learning/>

의 조언에 따라 머신러닝에 필요한 기초 통계학 및 확률론 개념인

순열/조합, 확률, 베이즈 정리, 분산과 기댓값, 확률변수, 확률분포

에 관해서 3장에서 다룬다. 그 밖에

적률생성함수 (모멘트 생성 함수, Moment Generating Functions, mgf),
 최대 우도 추정 (Maximum Likelihood Estimation, MLE)
 사전 및 사후 확률 (Prior and Posterior)
 최대 사후 추정 (Maximum a Posteriori Estimation, MAP)
 샘플링 방식 (Sampling Methods), 추정, 검정 등

의 개념에 관하여는 추후 <심화 과정>에서 다루도록 한다. ■

1. 순열, 조합, 확률

19강 동영상, 순열, 조합, 확률 <https://youtu.be/KQXO-XbJauU> (33:10)

이 절에서는 특정 사건이 일어날 가능성을 수로 나타낸 확률에 관하여 다룬다. 그런데, 확률은 특정 사건이 일어나는 경우의 수를 전체 사건의 경우의 수로 나눈 것으로 정의되므로, 우선 경우의 수를 계산할 수 있어야 한다. 따라서 순열과 조합을 계산하는 방법을 먼저 살펴본다.

1.1 순열과 조합 (Counting Methods, 경우의 수를 구하는 법)

참고 동영상: (기초) <https://youtu.be/P5rDAqiHsXE> (심화) <https://youtu.be/I6XW6DKLoCU>

실습 사이트: <http://matrix.skku.ac.kr/2018-DM/DM-Ch-6-Lab.html>

문제풀이: <http://matrix.skku.ac.kr/2018-DM-Sol/Ch6/> https://youtu.be/_lxv-cysbGQ

- 경우의 수를 세는 방법 중 기본이 순열과 조합이다. 여기서는 순열과 조합에 대하여 알아본다.
- 서로 다른 물건들 중 몇 개를 골라 순서를 주어 나열한 경우의 수를 순열(permuation)이라 하고, 서로 다른 물건들 중 몇 개를 골라 순서 없이 나열한 경우의 수를 조합(combination)이라고 한다.

정리. 순열

(1) 서로 다른 n 개에서 k 개를 순서대로 고르는 경우의 수(순열)는

$${}_n P_k = \frac{n!}{(n-k)!} \text{이다. (특히 } k=n\text{일 때는 } {}_n P_n = n! = 1 \cdot 2 \cdot 3 \cdot \dots \cdot n\text{이다.)}$$

(2) 서로 다른 n 개에서 k 개를 뽑아 원형으로 배열할 수 있는 경우의 수는

$${}_n P_k \times \frac{1}{k}$$
이다.

(3) 서로 다른 n 개에서 k 개를 뽑아 목걸이를 만들 수 있는 경우의 수는 ${}_n P_k \times \frac{1}{2k}$ 이다.

정리. 조합, 중복조합, 이항정리(Binomial theorem)

(1) [조합] 서로 다른 n 개에서 중복 없이 k 개를 택하는 방법의 수는 다음과 같다.

$\binom{n}{k} = {}_n C_k = \frac{n!}{k! (n-k)!} = \binom{n}{n-k}$, 이를 이항계수(binomial coefficient)라고 한다.

(2) [중복조합] 서로 다른 n 개에서 중복을 허락하여 k 개를 택하는 방법의 수는 k 개의 빈칸과 칸막이의 수 $n-1$ 개를 합한 $k+n-1$ 개의 빈칸에 칸막이가 들어갈 $n-1$ 개의 칸을 선택하는 문제이다. 따라서 중복조합 ${}_n H_k$ 은 ${}_n H_k = \binom{n+k-1}{n-1}$ 이 된다. 그런데 ${}_m C_r = {}_m C_{m-r}$ 이므로 ${}_n H_k = \binom{n+k-1}{k}$ 이 된다.

(3) [이항정리] : $2^n = \binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{n}$, $(x+y)^n = \sum_{k=0}^n \binom{n}{k} x^{n-k} y^k$

(4) [Pascal의 공식] $1 \leq k \leq n-1$ 를 만족하는 정수 n 과 k 는 다음 식을 만족한다.

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1} \quad [\text{조합론적 증명}]$$

1.2 Multiset(중복집합) 의 순열

■ S 를 무한반복수를 갖는 k 가지 서로 다른 물건(object)들의 multiset(중복집합)이라 하면, (multiset) S 의 전체 r -permutations (순서를 주어 r 개를 고르는 경우)의 수는 k^r 이다.

■ repetition numbers(유한반복의 수) n_1, n_2, \dots, n_k 개를 가진 (multi-set) 주며 S 를 생각하자. 모두 유한이므로 $n_1 + n_2 + \cdots + n_k = n$ 이라 놓자. 그러면 (multi-set) S 의 permutation (모든 n 개를 순서대로 늘어놓는) 경우의 수는 $\frac{n!}{n_1! n_2! \cdots n_k!}$ 이다.

증명

$S = \{n_1 a_1, n_2 a_2, \dots, n_k a_k\}$, $n_1 + n_2 + \dots + n_k = n$ 라 하자. 이제 이 S 안에서 n 개 물건의 치환을 만들려면 n 개의 자리 중 n_1 개의 a_1 을 놓을 자리를 먼저 택하고 $\{\binom{n}{n_1}\}$ 가지 경우가 있다}, 남은 $n - n_1$ 개의 자리 중 n_2 개의 a_2 을 놓을 자리를 택하고, $\{\binom{n-n_1}{n_2}\}$ 가지 경우 }, …, 끝으로 n_k 개의 a_k 는 남은 n_k 개의 자리에 놓으면 된다. – 각 a_i 들은 같은 type 이므로 순열이 아닌 조합이 된다. 곱셈 법칙에 의해

$$\begin{aligned} & \binom{n}{n_1} \binom{n-n_1}{n_2} \binom{n-n_1-n_2}{n_3} \cdots \binom{n-n_1-n_2-\cdots-n_{k-1}}{n_k} \\ &= \frac{n!}{n_1!(n-n_1)!} \cdot \frac{(n-n_1)!}{n_2!(n-n_1-n_2)!} \cdot \frac{(n-n_1-n_2)!}{n_3!(n-n_1-n_2-n_3)!} \cdot \cdots \cdot \frac{(n-n_1-\cdots-n_{k-1})!}{n_k!(n-n_1-\cdots-n_k)!} \\ &= \frac{n!}{n_1!n_2!\cdots n_k!} = \frac{n!}{n_1!n_2!\cdots n_k!} \quad \text{이다.} \end{aligned} \quad \blacksquare$$

■ 다항계수를 이용하여 아래와 같이 쓰기도 한다.

$$\binom{n}{n_1, n_2, \dots, n_k} = \frac{n!}{n_1!n_2!\cdots n_k!} .$$

예제 1 52장의 포커 카드에서 5장의 카드(a poker hand)를 고르는 경우의 수는?

풀이 이 문제는 52장의 카드에서 순서를 고려하지 않고 5장을 뽑는 문제이다. 따라서

$$\binom{52}{5} = {}_{52}C_5 = \frac{52!}{5!47!} = \frac{311875200}{120} = 2598960$$

즉, 총 2,598,960 가지가 된다. ■

[Sage code] <http://mathlab.knou.ac.kr:8080/> <http://sage.skku.edu/>

```
print(factorial(52)/factorial(5)/factorial(47)) # 52!
5!47!
```

답. 2598960 ■

예제 2 500개의 전구로부터 5개의 백열전구를 택하는 방법의 개수는
 $\binom{500}{5} = \frac{500!}{5! 495!}$ 이다.

[Sage code] [이항계수를 구하는 코드]

`print(binomial(500, 5))` # $\binom{500}{5} = \frac{500!}{5! 495!}$

답. 255,244,687,600 # (2천5백52억 4천4백만가지 이상) ■

■ 이항계수와 관련하여 다음이 성립한다.

$$\binom{a}{k} = \frac{a(a-1)(a-2)\cdots(a-k+1)}{k!} \quad (k \geq 0, k \in \mathbb{Z}),$$

$$\binom{-m}{k} = (-1)^k \binom{m+k-1}{k} \quad (k \geq 0, k \in \mathbb{Z}, m > 0)$$

$$\sum_{s=0}^{n-1} \binom{k+s}{k} = \binom{n+k}{k+1} \quad (k \geq 0, n \geq 1, k, n \in \mathbb{Z}), \quad \sum_{k=0}^r \binom{p}{k} \binom{q}{r-k} = \binom{p+q}{r} \quad (r \geq 0, r \in \mathbb{Z})$$

* 대부분의 $\binom{n}{k}$ 는 위의 관계식을 이용하여 쉽게 구할 수 있다.

1.3 확률

[수학사] 확률론의 탄생 <https://youtu.be/oseQFNsc3sA>

- S 를 전체 사건(event)의 집합(표본공간, sample space)이라 하고, A 를 특정 사건의 집합이라 하자. 그러면 사건 A 가 일어나는 가능성을 수로 나타낸 확률(probability) $P(A)$ 는 A 가 일어나는 경우의 수 $n(A)$ 를 전체 경우의 수 $n(S)$ 로 나누어서 구한다.

정의. 수학적 확률, 기하학적 확률, 통계적 확률

(1) 수학적 확률

$$P(A) = \frac{\text{사건 } A \text{가 일어나는 경우의 수}}{\text{일어날 수 있는 모든 경우의 수}} = \frac{n(A)}{n(S)}$$

(2) 기하학적 확률

$A \subset S$ 인 영역 A 에 속하는 확률은

$$P(A) = \frac{\text{영역 } A \text{의 크기}}{\text{전체의 영역 } S \text{의 크기}}$$

■ 통계적 확률과 대수의 법칙(Law of large number)

시행 횟수를 n , 특정 사건 A 가 일어날 횟수를 k 라 하면, n 이 한없이 커질 때 통계적 확률 $\frac{k}{n}$ 는 일정한 값 p (수학적 확률)에 가까워진다. 즉 다음이 성립한다.

$$P(A) = \lim_{n \rightarrow \infty} \frac{k}{n} = a$$

확률의 정의를 공리로 나타내면, 다음과 같다.

■ 확률의 공리

다음을 만족하는 $P(A)$ 를 사건 A 의 확률이라 한다.

- ① 표본공간 S 에서 임의의 사건 A 에 대하여 $0 \leq P(A) \leq 1$ 이 성립한다.
- ② 표본공간 S 에 대하여 $P(S) = 1$ (표본공간 전체의 확률은 1)이 성립한다.
- ③ 공사건 \emptyset 에 대하여 $P(\emptyset) = 0$ 이 성립한다.
- ④ A_1, A_2, \dots 이 서로 배반사건(exclusive events, 背反事件) 이면 다음이 성립한다.

$$P(A_1 \cup A_2 \cup \dots) = P(A_1) + P(A_2) + \dots$$

■ 확률의 기본개념

A : 어떤 사건

A^c : 사건 A 가 일어나지 않을 경우

$P(A)$: 사건 A 가 일어날 확률

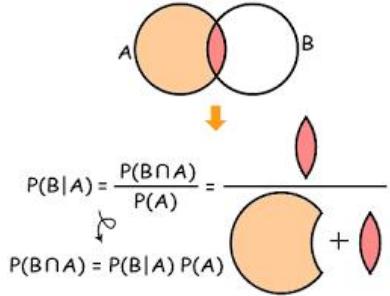
$P(A^c) = 1 - P(A)$: 사건 A 가 일어나지 않을 확률

- [사건 A 가 반드시 일어날 때, 사건 A 가 일어날 확률 $P(A)$ 는 1 이다.]
- [사건 A 가 절대로 일어나지 않을 때, 사건 A 가 일어날 확률 $P(A)$ 는 0이다.]
- ▣ 조건부확률은 확률과 데이터 분석에서 가장 중요한 개념이다.

정의. 조건부 확률

어떤 사건 A 가 일어났다는 조건하에서 사건 B 가 일어날 확률을 사건 A 에 대한 사건 B 의 조건부 확률(conditional probability)이라 하고 $P(B|A)$ 로 표시하며 다음과 같이 정의한다.

$$P(B|A) = P_A(B) = \frac{P(A \cap B)}{P(A)} = \frac{P(B \cap A)}{P(A)} \quad (\text{단, } P(A) > 0)$$



[그림출처] <https://blog.naver.com/alwaysneoi/100148922781>

정리. 조건부 확률의 곱셈정리

조건부 확률의 정의로부터 다음의 곱셈정리(관계식)를 얻을 수 있다.

$$P(A \cap B) = P(A)P(B|A) = P(B)P(A|B) \quad [\text{Note } P(A_i \cap B) = P(A_i)P(B|A_i)]$$

또한, 일반적으로 사건 A_1, A_2, \dots, A_n 에 대하여 다음이 성립한다.

$$\begin{aligned} & P(A_1)P(A_2|A_1)P(A_3|A_1 \cap A_2) \cdots P(A_n|A_1 \cap A_2 \cap \cdots \cap A_{n-1}) \\ &= P(A_1) \cdot \frac{P(A_1 \cap A_2)}{P(A_1)} \cdot \frac{P(A_1 \cap A_2 \cap A_3)}{P(A_1 \cap A_2)} \cdots \frac{P(A_1 \cap A_2 \cap \cdots \cap A_n)}{P(A_1 \cap A_2 \cap \cdots \cap A_{n-1})} = P(A_1 \cap A_2 \cap \cdots \cap A_n) \end{aligned}$$

예제 3 다음 물음에 답하여라.

(1) 아래의 R코드로 동전 한 개를 10회 던져 보고, 뒷면의 수와 앞면의 수를 기록해 보자. 동일한 방법으로 같은 동전을 100회 던지는 실습을 해 보자.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
coin = sample(c("앞","뒤"), 10, replace = TRUE) # 10회 반복
print(coin)
table(coin)
```

```
[1] "뒤" "뒤" "뒤" "앞" "뒤" "뒤" "뒤" "앞" "뒤" "앞"
coin
뒤 앞
7 3
```

[출처] <http://matrix.skku.ac.kr/2018-album/R-Sage-Stat-Lab-2.html>

▶ 시행의 횟수를 아주 크게 늘려 가면, 앞서 대수의 법칙에서 설명한 바와 같이 뒷면과 앞면이 나오는 확률이 $\frac{1}{2}$ (수학적 확률)로 수렴함을 확인할 수 있다.

(2) 1부터 45까지의 숫자에서 임의로 숫자 6개를 뽑으시오.

풀이. 아래 R 명령어를 이용하여 실습하면 된다.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
# 45개에서 6개 숫자 뽑기
sample(1:45, 6)
```

```
[1] 20 43 24 7 30 33
```

(3) 로또를 고객이 1부터 45사이의 숫자 중 6개를 임의로 선택하여 매주 토요일 추첨되는 당첨 번호와 같을 경우 당첨금을 받는 방식으로 운영되는 복권이라 하자. '숫자 6개가 모두 일치하는 로또 1등'으로 당첨될 확률을 구해보시오.

풀이. 1부터 45사이의 숫자 중 6개를 임의로 선택하는 경우의 수는 $\binom{45}{6}$ 이고, 1등으로 당첨되는 경우는 한 가지 밖에 없으므로, 구하고자 하는 확률은 다음과 같다.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
# 45개의 수에서 6개의 숫자가 나올 모든 종류의 수
print(choose(45, 6))
# 1등 당첨될 확률
1/choose(45, 6)          #  $P(A) = \frac{\text{1 등 경우의 수}}{\text{일어날 수 있는 모든 경우의 수}} = \frac{1}{n(S)}$ 
```

```
[1] 8,145,060      # n(S)
[1] 1.227738e-07  #  $P(A) = n(A) / n(S)$ , 800만분의 1
```

(4) 주사위 1개를 던져서 짝수가 나타날 확률을 구하시오.

풀이. 주사위 1개를 던지는 시행에서 일어나는 전체 사건은 1, 2, 3, 4, 5, 6이고, 짝수가 나타나는 사건은 2, 4, 6 이므로, 구하고자 하는 확률은 다음과 같다.

[Sage code] <http://mathlab.knou.ac.kr:8080/> <http://sage.skku.edu/>

```
S = Set([1, 2, 3, 4, 5, 6])      # 표본공간
A = Set([2, 4, 6])                # 짝수 사건
P_A = A.cardinality()/S.cardinality() # 짝수가 나타날 확률
P_A
```

```
1/2          #  $P(A) = n(A) / n(S) = 3/6$ , 2분의 1
```

(5) 1000개의 제품 중에 불량품이 3개 있다. 이 제품 중에서 10개의 제품을 구입했을 때 다음과 같은 확률은?

- (i) 구입제품 중 불량품이 한 개도 없는 경우
- (ii) 구입제품 중 불량품이 적어도 한 개 이상 있는 경우

풀이. 1000개의 제품 중에 10개의 제품을 선택하는 경우의 수는 $\binom{1000}{10}$ 이다. 따라서

(i) 불량품이 한 개도 없는 경우는 정상 제품인 997개에서 10개를 모두 선택하고, 불량품 3개에서는 하나도 선택하지 않는 경우 밖에 없으므로 그 경우의 수는 $\binom{997}{10} \times \binom{3}{0}$ 이다.

(ii) 불량품이 적어도 한 개 이상 있을 확률은, 1에서 불량품이 한 개도 없는 확률을 빼면 되므로, 구하고자 하는 확률은 $1 - \frac{\binom{997}{10} \times \binom{3}{0}}{\binom{1000}{10}}$ 이 된다.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

모두 불량품이 아닐 확률

```
print(choose(997,10)* choose(3,0)/ choose(1000,10))
```

적어도 불량품이 1개 이상 있을 확률

```
1 - choose(997,10)* choose(3,0)/ choose(1000,10)
```

```
[1] 0.9702695 # (i)의 답,  $\binom{997}{10} \times \binom{3}{0} / \binom{1000}{10}$ 
```

```
[1] 0.02973045 # (ii)의 답 1 -  $\binom{997}{10} \times \binom{3}{0} / \binom{1000}{10}$ 
```

(6) 주사위 1개를 던질 때 짹수가 나오거나 3의 배수가 나올 확률은?

풀이. 주사위 1개를 던지는 시행에서 일어나는 전체 사건은 1, 2, 3, 4, 5, 6이고, 짹수가 나오거나 3의 배수가 나오는 사건은 2, 3, 4, 6 이므로, 구하고자 하는 확률은 다음과 같다.

[Sage code] <http://mathlab.knou.ac.kr:8080/> <http://sage.skku.edu/>

```
S = Set([1, 2, 3, 4, 5, 6]) # 표본공간
A = Set([2, 4, 6]) # 짹수 사건
B = Set([3, 6]) # 3의 배수 사건
AcupB = A.union(B) # 합집합 사건
P_AcupB = AcupB.cardinality()/S.cardinality() # 합집합 사건이 나타날 확률
P_AcupB # P(A ∪ B) = n(A ∪ B) / n(S) = 4/6 = 2/3
```

```
2/3 # n(A ∪ B) / n(S) = 4/6 = 2/3
```

(7) 주사위 1개를 던질 때 짹수가 나왔다는 조건 하에 3의 배수가 나올 확률은?

풀이. 주사위 1개를 던질 때, 짹수가 나오는 사건은 2, 4, 6이고, 이 조건 하에서 3의 배 수가 나오는 사건은 6 뿐이므로, 구하고자 하는 확률은 3가지 경우 중 한 가지 뿐 이므로 $\frac{1}{3}$ 이다.

[Sage code] <http://mathlab.knou.ac.kr:8080/> <http://sage.skku.edu/>

```
S = Set([1, 2, 3, 4, 5, 6]) # 표본공간
A = Set([2, 4, 6])          # 짹수 사건
B = Set([3, 6])            # 3의 배수 사건
AcapB = A.intersection(B)  # 짹수이면서 3의 배수인 사건
# 조건부 확률
P_AcapB = AcapB.cardinality()/A.cardinality() # P(B | A) = P(A∩B) / P(A) =
n(A∩B) / n(A)
P_AcapB
```

1/3 # $1/3 = n(A \cap B) / n(A)$



2019 Conference of Joint Societies for Mathematics Education
by KSESM & KSME <https://youtu.be/BnrBMYuCdil>

1.3 베이즈 정리 (Bayes' theorem)

참고 동영상: <https://youtu.be/VAGLigLt2Hw>

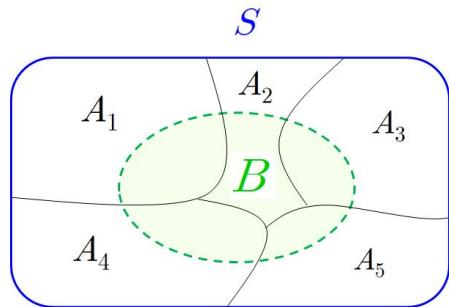
- 베이즈 정리는 불확실성 하에서 의사결정 문제를 수학적으로 다룰 때 중요하게 이용된다. 특히, 정보와 같이 눈에 보이지 않는 무형자산이 지닌 가치를 계산할 때 유용하게 사용된다.
- ▶ 사전확률(prior probability)은 일이 일어나기 전, 즉 사전에 미래에 어떤 사건이 일어날 확률을 측정한 것을 말한다. 베이즈(Bayesian) 통계 추정에서, 사전확률분포는 확률 변수에 대해 관측 자료를 고려하지 않고 획득한 결과를 말한다. $P(A)$ 는 A에 대한 사전 확률을 나타낸다.
- ▶ 사후확률(posterior probability)은 사전확률과 대비되는 개념으로 확률변수에 대한 관측이나 증거에 대한 조건부 확률을 말한다. 즉 어떤 특정사건이 이미 발생하였는데, 이 특정사건이 나온 연원이 무엇인지 불확실한 상황을 식으로 나타낸 것이다. $P(A|B)$ 로 표현될 수 있다 (B 는 이미 일어난 사건이고, 사건 B 를 관측한 후에 그 원인이 되는 사건 A 의 확률을 따졌다는 의미로 사후확률이라고 정의한다).
- 베이즈 정리(Bayes' theorem)는 사전확률과 사후확률의 관계를 조건부 확률을 이용하여 계산하는 이론이다.

정리.

베이즈 정리(Bayes' theorem)

$\{A_1, A_2, \dots, A_n\}$ 이 표본공간 S 의 분할(partition)을 이룬다고 하자. 그러면 임의의 사건 B 에 대하여 다음이 성립한다.

$$B = S \cap B = (A_1 \cup A_2 \cup \dots \cup A_n) \cap B = (A_1 \cap B) \cup (A_2 \cap B) \cup \dots \cup (A_n \cap B)$$



이 때 $A_i \cap B$ ($i = 1, 2, \dots, n$)는 서로 배반(exclusive)이다. 따라서

$$P(B) = P(A_1 \cap B) + P(A_2 \cap B) + \dots + P(A_n \cap B)$$

이다. 한편, 확률의 곱셈정리로부터 아래 전확률공식(Law of Total Probability)을 얻을 수 있다.

$$P(B) = P(A_1)P(B|A_1) + P(A_2)P(B|A_2) + \dots + P(A_n)P(B|A_n)$$

또한, 임의의 i 에 대한 조건부확률 $P(A_i|B) = \frac{P(A_i \cap B)}{P(B)}$ 에

$P(A_i \cap B) = P(A_i)P(B|A_i)$ 와 위의 전확률(total probability) 공식을 대입하면 다음 식을 얻을 수 있는 데 이를 베이즈 정리(Bayes' theorem)라고 한다.

$$P(A_i|B) = \frac{P(A_i \cap B)}{P(B)} = \frac{P(A_i) \cdot P(B|A_i)}{P(A_1)P(B|A_1) + P(A_2)P(B|A_2) + \dots + P(A_n)P(B|A_n)}$$

▶ 베이즈 정리에서 $P(A_i)$ 를 사건 A_i 의 사전확률, $P(A_i|B)$ 를 사건 A_i 의 사후확률이라 한다.

예제 4 3대의 기계 A , B , C 가 각각 이 공장의 생산품 전체의 50%, 30%, 20%를 생산한다. 그리고 이들 기계가 불량품을 생산할 비율은 각각 4%, 3%, 2%이다. 한 제품을 임의로 선택할 때 그 제품이 불량일 확률을 구하여라. 또 한 불량품이 기계 C 에 의하여 생산될 확률을 구하여라.

풀이. 구입한 1개의 제품이 C 사의 제품인 사건을 C 로 나타내고, 그것이 불량품이라는 사건을 X 로 나타내면,

(제품을 생산하는 사건) $A \cup B \cup C = S$ 이고, $P(A) = 0.5$, $P(B) = 0.3$, $P(C) = 0.2$.

(불량품을 생산하는 사건) $X = (A \cap X) \cup (B \cap X) \cup (C \cap X)$

(불량품을 생산하는 확률) $P(X|A) = 0.04$, $P(X|B) = 0.03$, $P(X|C) = 0.02$

$$P(X) = P(A)P(X|A) + P(B)P(X|B) + P(C)P(X|C)$$

$$= 0.5 \times 0.04 + 0.3 \times 0.03 + 0.2 \times 0.02 = 0.033$$

이므로, 베이즈 정리에 의하여 불량품 중 C사 제품이 불량품일 확률은 다음과 같다.

$$\begin{aligned} P(C|X) &= \frac{P(C)P(X|C)}{P(A)P(X|A) + P(B)P(X|B) + P(C)P(X|C)} \\ &= \frac{0.2 \times 0.02}{0.5 \times 0.04 + 0.3 \times 0.03 + 0.2 \times 0.02} = \frac{4}{33} \quad \blacksquare \end{aligned}$$

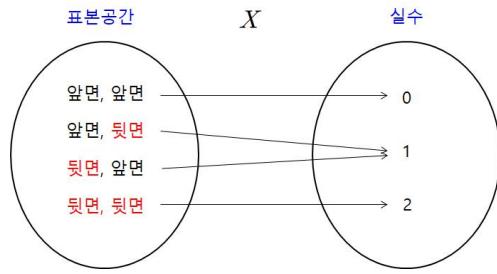
2. 확률변수 (random variable)

20강 동영상, 확률변수 <https://youtu.be/SUsZHarQqqg> (28:03)

- ▶ 확률적 데이터를 수학기호로 표시할 때는 변수를 표시할 때처럼 문자로 표시한다. 하지만 일반적인 변수가 특정한 하나의 숫자를 대표하는 변수(예, 생년월일)인 것과 달리 확률적 데이터를 대표하는 변수(예, 혈압)는 나올 수 있는 값이 확률적 분포를 가진다. 즉 특정한 값은 자주 나오고 다른 어떤 값은 드물게 나올 수 있다. 이러한 변수를 확률변수라고 한다.

2.1 확률변수, 기댓값, 분산 및 표준편차

- 일정한 확률을 갖고 발생하는 사건(event)에 수치가 부여되는 함수를 확률변수(random variable)라 한다. 즉 표본 공간의 모든 표본에 대해 어떤 실수 값을 대응시킨(할당한) 것이다. 확률 변수는 숫자 혹은 벡터를 생성하는 기계(블랙박스)에 비유할 수 있다. 예를 들어, 동전 2개를 동시에 던지는 시행에서 뒷면이 나오는 동전의 개수를 X 라 하면, X 에 0, 1, 2를 할당 할 수 있다. 따라서 X 는 확률변수이다.



- X 가 가질 수 있는 값의 범위가 이산적인지/연속적인지(셀 수 있는지/없는지)에 따라 이산확률변수(discrete random variable)와 연속확률변수(continuous random variable)로 구분한다. 이 절에서는 확률변수와 기댓값 개념을 소개한다.

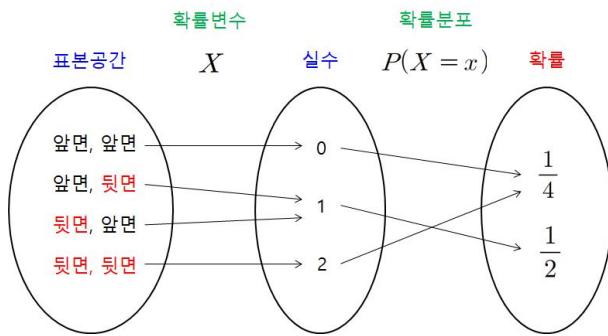
정의. 확률분포(probability distribution)

- (1) 확률변수 X 가 가지는 x 에 확률 $P(X=x)$ 를 대응시키는 함수를 X 의 확률분포(probability distribution)라 한다.
- (2) 변수 X 가 취할 수 있는 모든 값 x_1, x_2, \dots, x_n 이 취하는 확률이 각각 $f(x_1), f(x_2), \dots, f(x_n)$ 로 주어질 때, X 를 이산확률변수라 하고 $f(x_i)$ 를 X 의 이산확률함수(probability function) 또는 확률질량함수(probability mass function, pmf)라 한다.

X	x_1	x_2	...	x_n	합(sum)
확률	$f(x_1)$	$f(x_2)$...	$f(x_n)$	1

- 확률변수 X 가 a 이상 b 이하인 값을 취하는 확률을 $P(a \leq x \leq b)$ 와 같이 나타낸다. X 가 이산확률변수일 때 $P(a \leq x \leq b) = \sum_{x=a}^b P(X=x)$ 이다.

앞서 언급한 동전 2개를 동시에 던지는 시행에서, 뒷면이 나오는 동전의 개수 X 의 확률분포를 그림으로 나타내면 다음과 같다.



- 확률질량함수(probability mass function, pmf)는 다음과 같은 성질이 있다.

정리. 확률질량함수(pmf)의 성질

$$(1) 0 \leq f(x_i) \leq 1 \quad (i = 1, 2, \dots, n)$$

$$(2) f(x_1) + f(x_2) + \dots + f(x_n) = \sum_{i=1}^n f(x_i) = 1$$

(3) X 가 이산형 분포를 가지면 실수의 부분집합 x_i 의 확률은

$$P(X \leq x_i) = \sum_{x \leq x_i} f(x)$$

■ 확률변수의 기댓값(expectation)은 확률적 사건에 대한 평균값으로, 사건이 벌어졌을 때의 얻은 값과 그 사건이 일어날 확률을 곱한 것을 전체 사건에 대해 합한 값이다. 이것은 어떤 확률적 사건에 대한 평균의 의미를 갖는다. 확률변수의 분산(variance)은 그 확률변수가 기댓값으로부터 얼마나 떨어진 곳에 분포하는지를 가늠하는 숫자이고, 표준편차(standard deviation)는 분산의 양의 제곱근으로 정의된다.

- 이산확률변수 X 의 기댓값과 분산, 표준편차는 다음과 같이 계산한다.

$$\text{기댓값} : E(X) = \mu = \sum_i x_i f(x_i)$$

$$\text{분산} : V(X) = \sigma^2 = \sum_i (x_i - \mu)^2 f(x_i) = E(X^2) - \mu^2$$

$$\text{표준편차} : \sigma = \sqrt{V(X)} = \sqrt{\sum_i (x_i - \mu)^2 f(x_i)}$$

예제 1 확률변수 X 의 확률분포가 다음과 같을 때, 기댓값과 분산, 표준편차를 구하여라.

x	0	1	2	3	합계
$f(x)$	0.010	0.840	0.145	0.005	1

[출처] <http://matrix.skku.ac.kr/2018-album/R-Sage-Stat-Lab-2.html>

풀이. 기댓값과 분산, 표준편차의 정의를 이용하여 아래와 같이 R코드로 구하면 된다.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```

x <- c(0, 1, 2, 3)                      # 확률변수
pr.x <- c (0.010, 0.840, 0.145, 0.005)    # 확률분포
e.x <- sum(x*pr.x)                       # 기댓값
print(e.x)
var.x <- sum((x^2)*pr.x)-e.x^2           # 분산
print(var.x)
sd.x <- sqrt(var.x)                       # 표준편차
sd.x

```

```

[1] 1.145                                # 기댓값       $\mu$ 
[1] 0.153975                             # 분산         $\sigma^2$ 
[1] 0.3923965                            # 표준편차    $\sigma$ 

```

- ▶ 이산확률변수의 확률분포를 나타내는 것이 확률질량함수(pmf)고, 연속확률변수의 확률을 결정하는 함수가 확률밀도함수(pdf)이다. 여기서 ‘밀도’라는 단어가 어떻게 쓰이게 되었을까? 확률을 일종의 양(질량)으로 보고, 구간길이를 일종의 부피로 본다면, [확률/구간길이] \times [구간길이] = [확률] 이므로 [확률/구간길이]는 [질량/부피]가 되므로 ‘밀도’를 의미하게 된다. 따라서 ‘확률밀도함수’라는 용어가 사용되었다.

정의. 확률밀도함수(probability density function, pdf)

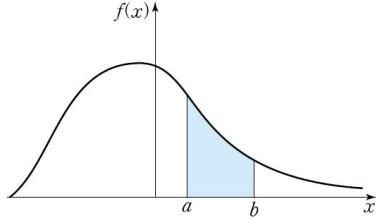
연속확률변수란 어떤 범위에 속하는 모든 실수값을 취할 수 있는 확률변수로, 그 분포는 확률밀도함수(probability density function, pdf)를 이용하여 다음과 같이 나타낸다. 즉,

연속확률변수 X 의 확률밀도함수가 $f(x)$ 일 때, $X \in [a \leq x \leq b]$ 있을 확률은
 $P(a \leq X \leq b) = \int_a^b f(x) dx$ 이다.

정리. 확률밀도함수(pdf)의 성질, (<https://youtu.be/UngRPUYdtoc>)

확률밀도함수는 확률변수의 분포를 나타내는 함수로, 다음 조건을 만족해야 한다.

$$(1) f(x) \geq 0 \quad (2) P(a \leq X \leq b) = \int_a^b f(x) dx \quad (3) \int_{-\infty}^{\infty} f(x) dx = 1$$



- 연속확률변수 X 의 기댓값과 분산, 표준편차는 다음과 같이 계산한다.

$$\text{기댓값 : } E(X) = \mu = \int_{-\infty}^{\infty} x f(x) dx$$

$$\text{분산 : } V(X) = \sigma^2 = \int_{-\infty}^{\infty} (x - \mu)^2 f(x) dx = E(X^2) - \mu^2$$

$$\text{표준편차 : } \sigma = \sqrt{V(X)} = \sqrt{\int_{-\infty}^{\infty} (x - \mu)^2 f(x) dx}$$

정리. 기댓값, 분산의 성질, 표준화 확률변수

기댓값, 분산에 대하여 다음 성질이 만족한다.

$$(1) E(X+b) = E(X)+b, \quad E(aX) = aE(X), \quad E(aX+b) = aE(X)+b$$

$$(2) V(X+b) = V(X), \quad V(aX) = a^2 V(X), \quad V(aX+b) = a^2 V(X)$$

$$(3) \text{ 확률변수 } X \text{에 대하여 새로운 확률변수 } Z \text{를 } Z = \frac{X - \mu}{\sigma} \text{로 정의하면 평균}$$

과 분산이 항상 0과 1이다. 따라서 이 확률변수 Z 를 확률변수 X 의 표준화 확률변수라 한다.

예제 2 확률변수 X 의 확률밀도함수가 $f(x) = 3x^2$, $0 < x < 1$ 일 때, X 와 $Y = 4X + 2$ 의 분산을 구하여라.

풀이. $E(X) = \mu = \int_0^1 3x^3 dx = \left[\frac{3}{4}x^4 \right]_0^1 = \frac{3}{4}$, $E(X^2) = \int_0^1 3x^4 dx = \left[\frac{3}{5}x^5 \right]_0^1 = \frac{3}{5}$

$$\Rightarrow V(X) = \sigma^2 = E(X^2) - \mu^2 = \frac{3}{5} - \left(\frac{3}{4} \right)^2 = \frac{3}{80}$$

$$E(Y) = E(4X + 2) = 4E(X) + 2 = 4 \times \frac{3}{4} + 2 = 5$$

$$\Rightarrow V(Y) = V(4X + 2) = V(4X) = 4^2 V(X) = 16 \times \frac{3}{80} = \frac{3}{5}$$
 [답]

[Sage code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
var('t')
f = 3*t^2
Ex = integral(t*f, t, 0, 1)
Ex2 = integral(t^2*f, t, 0, 1)
print(Ex)
print(Ex2)
print(Ex2 - Ex^2)          # X의 분산
Ey = integral((4*t+2)*f, t, 0, 1)
Ey2 = integral((4*t+2)^2*f, t, 0, 1)
print(Ey)
print(Ey2)
print(Ey2 - Ey^2)          # Y의 분산
```

3/4

3/5

$$\frac{3}{80} \quad # X의 분산, V(X) = \sigma^2 = E(X^2) - \mu^2 = \frac{3}{5} - \left(\frac{3}{4} \right)^2 = \frac{3}{80}$$

5

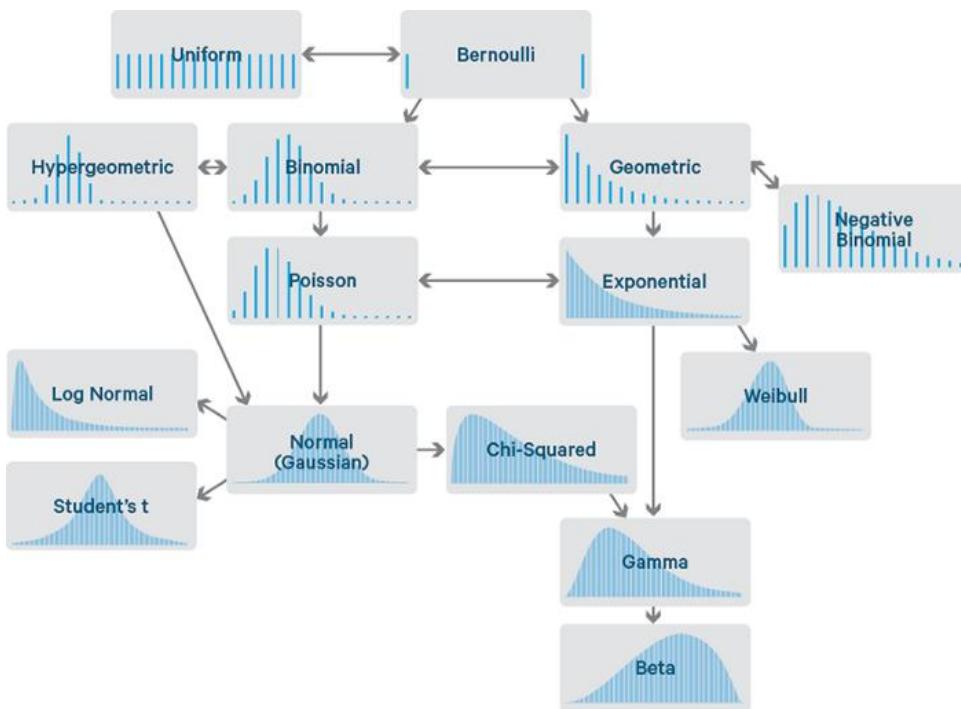
128/5

$$\frac{3}{5} \quad # Y = 4X + 2의 분산, V(Y) = V(4X + 2) = 16V(X) = 16 \times \frac{3}{80} = \frac{3}{5} \blacksquare$$

3. 확률분포

21강 동영상, 이산확률분포 https://youtu.be/Fq7D7bGG_cE (24:28)

- ▶ 확률분포는 확률이 어디에 어느 정도 분포되어 있는가를 수학적으로 명시하고 명확하게 전달하기 위한 도구이다. 아래 그림은 여러 확률분포 사이의 관련성을 나타낸 그림이다. 이 장에서는 데이터 과학에서 주로 사용되는 대표적인 확률분포에 대하여 학습한다. 확률 분포는 확률 변수가 어떤 종류의 값을 가지는가에 따라서 크게 이산 확률분포와 연속 확률분포 중 하나에 속하며, 둘 중 어디에도 속하지 않는, multivariate 확률분포 경우도 존재한다.



[그림출처]

<https://towardsdatascience.com/probability-distributions-in-data-science-cce6e64873a7>

3.1 이산확률분포

- ▶ 이산확률변수를 가지는 이산확률분포에는 베르누이 분포, 이항분포, 포아송 분포 등이 있다. 이 절에서는 이들의 정의와 개념에 대하여 살펴보고, 여러 확률분포 사이의 관계에 대하여 다룬다.

■ 베르누이 분포(Bernoulli distribution) $B(1, p)$

(1) 베르누이 시행(Bernoulli trial)은 1회 시행의 결과가 성공(Success) 혹은 실패(Fail) 두 가지 중 하나로만 나오는 실험을 말한다. 예를 들어, 동전을 한 번 던지는 시행에서 앞면이 나오면 ‘성공’, 뒷면이 나오면 ‘실패’라고 할 수 있다. 따라서 동전 던지기는 베르누이 시행이다. 이를 확률변수 X 로 나타낼 때, 일반적으로 성공한 결과를 1, 실패한 결과를 0으로 나타낸다.

(2) 성공할 확률이 p 인 베르누이 시행에서 확률변수 X 의 확률분포는 다음과 같다.

$$P(X=x) = p^x (1-p)^{1-x}, \quad x=0, 1$$

이때, 확률변수 X 는 베르누이 분포를 따른다고 하며, $X \sim B(1, p)$ 로 나타낸다.

(3) $X \sim B(1, p)$ 인 확률변수 X 의 기댓값과 분산은 다음과 같다.

$$E(X) = p, \quad V(X) = p(1-p)$$

예제 1 확률변수 X 를 주사위 한 개를 던져서 나온 결과라 하자. 1이 나오면 $X=1$ 이고 그 외의 숫자가 나오면 $X=0$ 이다. X 의 확률분포 $P(X=x)$ 를 구하여라.

풀이. X 는 베르누이 분포 $B(1, \frac{1}{6})$ 를 따른다. 따라서 X 의 확률분포는 다음과 같다.

$$P(X=x) = p^x (1-p)^{1-x} = \left(\frac{1}{6}\right)^x \left(\frac{5}{6}\right)^{1-x}, \quad x=0, 1 \quad \square$$

[Sage code] <https://sagecell.sagemath.org/> <http://sage.skku.edu/>

```
p = [(1/6)^t*(5/6)^(1-t) for t in range(0, 2)]  
print(p)
```

[5/6, 1/6] # 베르누이 시행에서 X 의 확률분포

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
dbinom(0:1, 1, 1/6)
```

[1] 0.8333333 0.1666667 # 베르누이 시행에서 X 의 확률분포

■ 이항분포(Binomial distribution) $B(n, p)$ (n 번 베르누이 시행의 성공 확률분포)

(1) 한 번의 시행에서 사건 A 가 일어날 확률이 p 라 하자. 이 시행을 n 회 독립적으로 반복할 때, 사건 A 가 일어나는 횟수를 x 라 하면, X 의 확률분포는 다음과 같다.

$$P(X=x) = \binom{n}{x} p^x q^{n-x} \quad (\text{단, } p+q=1, x=0, 1, 2, \dots, n)$$

이때, 확률변수 X 는 이항분포 $B(n, p)$ 를 따른다고 하며, $X \sim B(n, p)$ 로 나타낸다.

(2) $n=1$ 인 이항분포는 베르누이 분포 $B(1, p)$ 과 같다.

(3) 무한모집단에서 표본을 비복원추출(sampling without replacement)하거나 유한모집단에서 복원추출(sampling with replacement, 반복을 허용)을 하는 경우에는, 각 시행이 베르누이 시행의 조건을 만족하므로 이항분포를 사용할 수 있다.

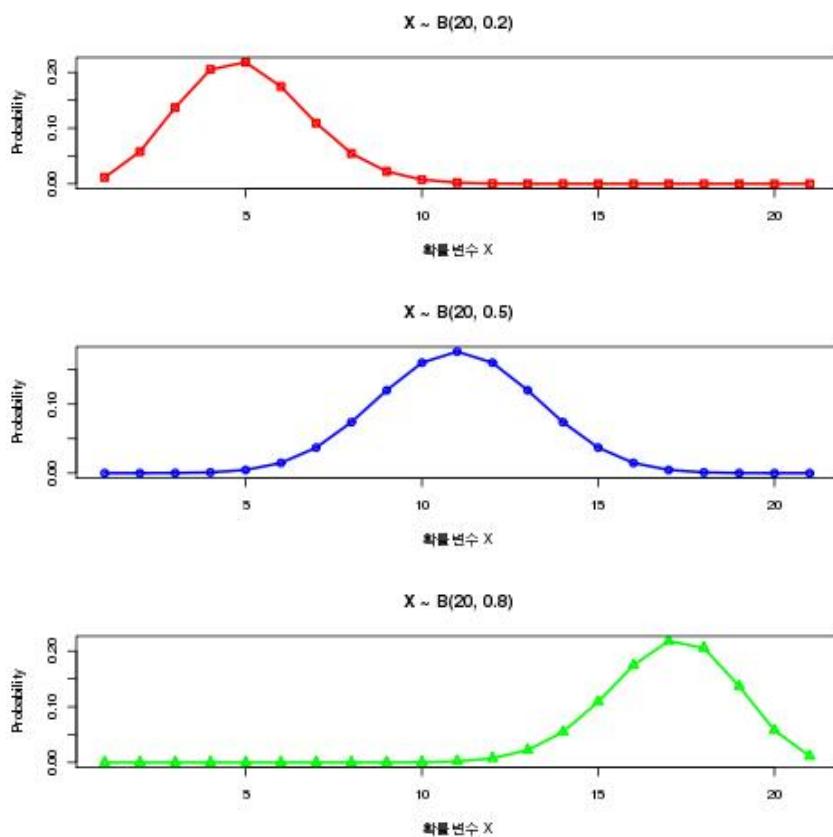
(4) $X \sim B(n, p)$ 인 확률변수 X 의 기댓값과 분산은 다음과 같다.

$$E(X) = np, \quad V(X) = np(1-p) = npq$$

(5) 이항분포의 형태는 모수(parameter)가 p 와 q 에 의해서 결정된다.

- ① $p < q$ 이면 왼쪽으로 치우친 형태
- ② $p = q$ 이면 완전 대칭
- ③ $p > q$ 이면 오른쪽으로 치우친 형태

예를 들어, 아래는 $n=20$ 이고, 각각 $p=0.2, 0.5, 0.8$ 인 이항분포의 그래프를 그린 것이다.



(6) 이항분포를 따르는 모집단에서 임의로 표본을 택하는 코드는 다음과 같다. 예를 들어, $B(150, 0.3)$ 인 모집단에서 표본 100개를 추출하려면 다음을 사용하면 된다.

[R code] <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

rbinom(100, size = 150, prob = 0.3)

```
[1] 50 49 49 49 52 40 45 44 39 54 55 39 41 44 38 41 38 49 46 44 40 41 47 50 42 36 43 37 45 55 40  
[32] 51 47 43 46 38 46 39 50 42 45 41 39 46 39 43 46 47 52 37 46 44 46 50 47 41 39 44 48 36 41 37  
[63] 41 41 43 48 51 39 41 31 40 40 45 42 41 37 43 56 44 41 41 37 29 52 47 50 51 42 45 50 44 34 36  
[94] 52 45 45 42 40 46 43
```

예제 2 대도시의 시민 중 30%만이 대중교통에 만족한다고 한다. 만약 이 도시의 시민 20명을 임의로 선택했을 때, (1) 이들 중 3명 미만의 시민이 대중교통에 만족할 확률과 (2) 6명의 시민이 만족할 확률, (3) 10명 이상의 시민이 만족할 확률을 구하라.

[출처] <http://matrix.skku.ac.kr/2018-album/R-Sage-Stat-Lab-2.html>

풀이. 시민 20명을 임의로 선택했을 때, 만족하는 사람의 수를 X 라 하면, X 는 이 항분포 $B(20, 0.3)$ 를 따른다. 따라서 구하고자 하는 확률은 다음과 같다.

$$(1) \ P(X \leq 2) \quad (2) \ P(X=6) \quad (3) \ P(X \geq 10)$$

아래의 R코드를 이용하여 각각의 확률을 얻는다.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
print(pbinom(2, size=20, prob=0.3, lower.tail=T)) # 3명 미만의 시민이 만족할 확률, T는 왼쪽  
print(dbinom(6, size=20, prob=0.3)) # 6명의 시민이 만족할 확률  
pbinom(10, 20, 0.3, lower.tail=F) # 10명 이상의 시민이 만족할 확률, F는 오른쪽
```

```
[1] 0.03548313 # 3명 미만의 시민이 만족할 확률  
[1] 0.191639 # 6명의 시민이 만족할 확률  
[1] 0.01714482 # 10명 이상의 시민이 만족할 확률
```

■

예제 3 어떤 바이러스 질환은 감염자와 접촉을 하게 되면 감염되며, 건강한 사람이 감염자와 한 번 접촉하였을 때, 감염될 확률은 20%라고 한다. 감염자가 임의의 건강한 사람 5명과 접촉했을 때, 5명 모두 감염될 확률은 얼마인가? 그리고 감염자 수의 기댓값과 분산, 표준편차는 얼마인가?

[출처] <http://matrix.skku.ac.kr/2018-album/R-Sage-Stat-Lab-2.html>

풀이. 감염자가 임의의 건강한 사람 5명과 접촉했을 때 감염자의 수를 X 라 하면, X 는 이항분포 $B(5, 0.2)$ 를 따른다. 따라서 구하고자 하는 5명 모두 감염될 확률은 $P(X=5)$ 이다. 아래의 R코드를 이용하여 $P(X=5)$ 와 X 의 기댓값, 분산, 표준편차를 얻는다.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
n=5;p=0.2
print(dbinom(5, size=n, prob=p))      # 5명 모두 감염될 확률
e.x=n*p                                # 기대값
print(e.x)
var.x=n*p*(1-p)                         # 분산
print(var.x)
sd.x=sqrt(var.x)                        # 표준편차
sd.x

[1] 0.00032    # 5명 모두 감염될 확률
[1] 1          # 기댓값    $\mu$ 
[1] 0.8        # 분산      $\sigma^2$ 
[1] 0.8944272 # 표준편차  $\sigma$ 
```

■

■ 포아송 분포(Poisson distribution) $P(\lambda)$

(1) 이항분포 $B(n,p)$ 에서 평균 $\lambda = np$ 는 일정하고, n 을 한없이 크게 할 때(상대적으로 p 는 한없이 작아짐) 이 확률분포는 포아송 분포가 된다. 즉, 포아송 분포 (Poisson distribution)는 이항분포의 극한분포(limiting distribution)로 설명될 수

있다.

[이항분포에서 포아송 분포의 유도]

이항분포에서 $\lambda = np$ 이고 p 가 작으면 n 이 크고 $np \leq 5$ 인 관계가 성립될 때 이항분포에서 포아송 분포가 유도된다.

$$\lambda = np \text{ 가 일정하면 } p = \frac{\lambda}{n} \text{ 이므로, } n \rightarrow \infty \text{ 일 때는}$$

$$\begin{aligned} [\text{이항분포}] \quad {}_n C_x p^x (1-p)^{n-x} &= \frac{n!}{x!(n-x)!} \left(\frac{\lambda}{n}\right)^x \left(1 - \frac{\lambda}{n}\right)^{n-x} \\ &= \frac{n(n-1)(n-2) \cdots (n-x+1)}{x!} \cdot \frac{\lambda^x}{n^x} \cdot \frac{\left(1 - \frac{\lambda}{n}\right)^n}{\left(1 - \frac{\lambda}{n}\right)^x} \\ &= \frac{\lambda^x}{x!} \cdot 1 \cdot \left(1 - \frac{1}{n}\right) \left(1 - \frac{2}{n}\right) \cdots \left(1 - \frac{x-1}{n}\right) \cdot \frac{\left(1 - \frac{\lambda}{n}\right)^n}{\left(1 - \frac{\lambda}{n}\right)^x} \end{aligned}$$

$$(n \rightarrow \infty \text{ 이므로}) \quad \lim_{n \rightarrow \infty} 1 \cdot \left(1 - \frac{1}{n}\right) \left(1 - \frac{2}{n}\right) \cdots \left(1 - \frac{x-1}{n}\right) = 1, \quad \lim_{n \rightarrow \infty} \left(1 - \frac{\lambda}{n}\right)^x = 1.$$

$$\text{이때 } -\frac{\lambda}{n} = \frac{1}{t} \text{ 로 치환하면 } n = -\lambda t \text{ 이고 } [n \rightarrow \infty \Rightarrow t \rightarrow -\infty]$$

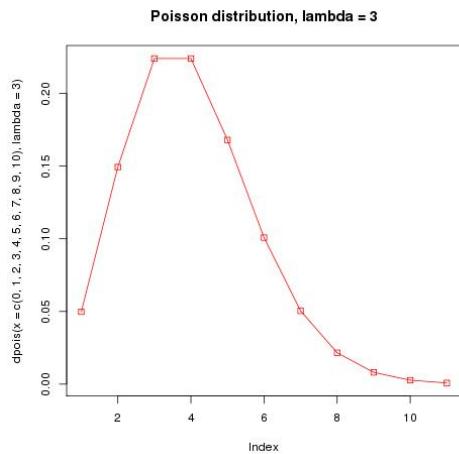
$$\lim_{n \rightarrow \infty} \left(1 - \frac{\lambda}{n}\right)^n = \lim_{t \rightarrow -\infty} \left(1 + \frac{1}{t}\right)^{-\lambda t} = \lim_{t \rightarrow -\infty} \left\{ \left(1 + \frac{1}{t}\right)^t \right\}^{-\lambda} = e^{-\lambda} \quad (\because \lim_{t \rightarrow -\infty} \left(1 + \frac{1}{t}\right)^t = e)$$

$$\text{따라서 } \lim_{n \rightarrow \infty} B(n, p) = \lim_{n \rightarrow \infty} {}_n C_x p^x (1-p)^{n-x} = \frac{\lambda^x}{x!} e^{-\lambda} = \frac{e^{-\lambda} \lambda^x}{x!} = P(\lambda) \text{ [Poisson분포, pmf]}$$

(2) 포아송 분포는 단위 시간 안에 어떤 사건이 발생하는 횟수를 확률변수 X 로 나타내는 확률분포이다. 어떤 사건이 단위 시간 안에 평균적으로 λ 번 발생한다고 가정할 때, $X=x$ 일 확률은 다음과 같이 주어진다.

$$P(X=x) = f(x) = \frac{e^{-\lambda} \lambda^x}{x!} \quad (\text{단, } x=0, 1, 2, \dots)$$

이때, 확률변수 X 는 포아송 분포 $P(\lambda)$ 를 따른다고 하며, $X \sim P(\lambda)$ 로 나타낸다.



(3) $X \sim P(\lambda)$ 인 확률변수 X 의 기댓값과 분산은 다음과 같다.

$$E(X) = \lambda, \quad V(X) = \lambda$$

(4) 주어진 시간 또는 정해진 영역에서 '성공'의 출현 횟수를 X 라 하면 X 는 포아송 분포를 따른다.

예를 들어 주어진 시간 내에 전화가 걸려 오는 횟수, 어떤 지역에서의 1일 교통사고 사망자 수, 하루 동안 고장 나는 기계의 수, 인쇄된 책자의 페이지 당 오자수, 일정량의 혈액 속에 있는 적혈구의 수는 포아송 분포 $P(X=x) = f(x) = \frac{e^{-\lambda} \lambda^x}{x!}$ 를 따른다. 여기서 λ 는 평균이다.

예제 4 포아송 분포를 따르는 모집단에서 임의로 표본을 택하는 코드는 다음과 같다. 예를 들어, $\lambda=2$ 인 포아송 분포 $P(2)$ 를 따르는 모집단에서 표본 100개를 추출하려면 다음을 사용하면 된다.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

rpois(100, lambda = 2)

```
[1] 0 1 1 3 1 3 1 4 0 1 2 3 4 3 0 3 2 0 4 0 1 2 2 2 3 2 1 3 1 4 1 0 1 2 2 0 1 0 0 2 0 3 1 1 3 0
[48] 4 3 5 2 2 2 1 1 4 5 4 1 2 3 1 0 1 1 3 4 3 0 1 4 1 4 3 2 1 3 2 1 4 0 1 0 1 3 0 2 5 1 2 1 3 2 4
[95] 1 2 2 4 2 3
```

예제 5 $P(X=1) = P(X=2)$ 인 포아송 분포를 따르는 X 가 있다. $P(X=3)$ 를 구하여라.

풀이. 포아송 분포를 따르는 X 의 확률질량함수(pmf)는 $f(x) = \frac{e^{-\lambda} \lambda^x}{x!}$ $x = 0, 1, 2, \dots$ 이다.

$$P(X=1) = P(X=2) \Rightarrow e^{-\lambda} \lambda = \frac{e^{-\lambda} \lambda^2}{2} \Rightarrow \frac{1}{2} \lambda e^{-\lambda} (2-\lambda) = 0 \text{ 를 얻는다. 이때},$$

$\lambda > 0$, $e^{-\lambda} > 0$ 이므로 $\lambda = 2$ 여야 한다. 따라서 $f(x) = \frac{e^{-2} 2^x}{x!}$ $x = 0, 1, 2, \dots$ 이다.

$$P(X=3) = f(3) = \frac{e^{-2} 2^3}{3!} = \frac{4}{3} e^{-2} \quad \blacksquare$$

[Sage code] <http://mathlab.knou.ac.kr:8080/> <http://sage.skku.edu/>

```
var('lam')
f(x, lam) = e^(-lam)*lam^x/factorial(x)
print(solve(f(1, lam) == f(2, lam), lam))
lam = 2
g(x) = e^(-lam)*lam^x/factorial(x)
print(g(3))
```

[lam == 0, lam == 2]

$$4/3 * e^{-2} \quad \# P(X=3) = f(3) = \frac{e^{-2} 2^3}{3!} = \frac{4}{3} e^{-2}$$

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
lam = 2
dpois(3, lam)
```

[1] 0.180447

$$\# P(X=3) = f(3) = \frac{e^{-2} 2^3}{3!} = \frac{4}{3} e^{-2} \approx 0.180447 \quad \blacksquare$$

3.2 연속확률분포

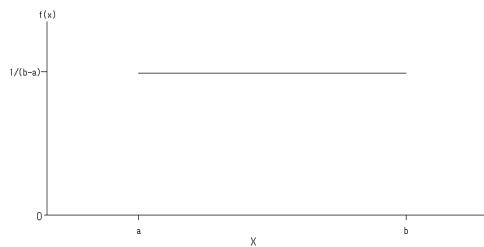
22장 동영상, 연속확률분포 <https://youtu.be/4wx1raETI8o> (42:46)

- ▶ 연속 확률분포(continuous probability distribution)는 확률밀도함수(pdf)를 이용해 분포를 표현할 수 있는 경우를 의미한다. 연속 확률 분포를 가지는 확률변수는 연속 확률 변수라고 부른다. 자주 사용되는 연속확률분포에는 균등분포, 정규분포, 지수분포 등이 있다. 이 절에서는 이들의 정의와 개념에 대하여 살펴보고, 여러 확률분포 사이의 관계에 대하여 다룬다.

■ 균등분포(Uniform distribution) $U(a,b)$

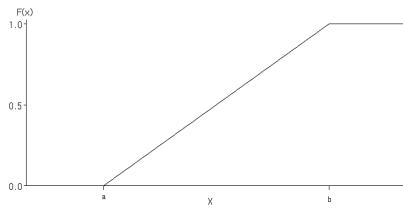
(1) 연속확률변수 X 가 a 와 b 사이에서 일정한 값을 취하고 $P(a \leq X \leq b) = 1$ 일 때 X 는 균등분포(uniformly distribution)를 따른다고 하며, $X \sim U(a,b)$ 로 표시한다. 균등분포의 확률밀도함수(probability density function, pdf) $f(x)$ 는 다음과 같다.

$$f(x) = \begin{cases} \frac{1}{b-a} & a \leq x \leq b \\ 0 & x < a, x > b \end{cases}$$



(2) $X \sim U(a,b)$ 일 때, 균등분포의 누적 분포함수(distribution function)는 다음과 같다.

$$F(x) = P(X \leq x) = \begin{cases} 0 & x < a \\ \frac{x-a}{b-a} & a \leq x < b \\ 1 & b \leq x \end{cases}$$



(3) $X \sim U(a, b)$ 일 때, 확률변수 X 의 기대값과 분산은 다음과 같다.

$$E(X) = \mu = \frac{1}{b-a} \int_a^b x dx = \frac{b+a}{2}, \quad V(X) = \sigma^2 = E(X^2) - \mu^2 = \frac{(b-a)^2}{12}$$

(4) 균등분포를 따르는 모집단에서 임의로 표본을 택하는 코드는 다음과 같다. 예를 들어, $U(5, 10)$ 인 모집단에서 표본 100개를 추출하려면 다음을 사용하면 된다.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

`runif(100, min=5, max=10)`

```
[1] 9.582467 5.240983 6.965723 5.286435 8.594378 6.800147 8.150651 8.008016 5.901423 5.814560
[11] 8.467467 9.354255 8.827177 7.741089 7.541865 8.774575 7.884471 8.320332 5.763640 7.769348
[21] 6.254356 9.307199 9.133660 6.215710 9.102613 7.894397 8.278282 6.843179 5.120876 5.545676
[31] 5.382515 5.673898 8.073488 7.408012 5.506840 8.312672 6.818407 6.878845 7.067366 8.818738
[41] 8.712181 6.452967 6.426883 9.000322 9.463622 9.257326 5.705153 6.736011 7.416410 8.427323
[51] 9.388521 6.444950 9.177553 5.551566 9.357263 9.443828 9.601413 7.495817 5.820730 6.915595
[61] 8.456109 5.598172 9.434568 9.492568 7.504497 7.736749 7.201645 5.888987 6.270696 8.490379
[71] 6.293398 5.609770 5.552201 9.728607 8.287793 9.702512 9.292375 8.747148 6.341373 5.872654
[81] 6.518375 8.587775 8.533157 6.627544 8.754870 8.338423 8.042655 8.708491 8.736092 7.620533
[91] 6.085618 5.724026 6.132644 7.881293 5.322445 9.900492 5.564567 7.293617 6.706736 5.555627
```

예제 6 확률변수 X 가 $U(0, 10)$ 를 따른다고 하자. 다음 확률을 구하여라.

$$(1) P(X < 3) \quad (2) P(X > 6) \quad (3) P(3 < X < 8)$$

풀이. $f(x) = \begin{cases} \frac{1}{b-a} & a \leq x \leq b \\ 0 & x < a, x > b \end{cases}$ 이고 $a = 0, b = 10$ 이므로 $f(x) = \frac{1}{10-0}$ 이고

$$(1) P(X < 3) = \int_0^3 f(x) dx = \int_0^3 \frac{1}{10} dx = \left[\frac{x}{10} \right]_0^3 = \frac{3}{10}$$

$$(2) P(X > 6) = \int_6^{10} \frac{1}{10} dx = \left[\frac{x}{10} \right]_6^{10} = \frac{4}{10} = \frac{2}{5}$$

$$(3) P(3 < X < 8) = \int_3^8 \frac{1}{10} dx = \left[\frac{x}{10} \right]_3^8 = \frac{5}{10} = \frac{1}{2}$$

□

[Sage code] <http://mathlab.knou.ac.kr:8080/> <http://sage.skku.edu/>

```
print(integral(1/10, x, 0, 3))
print(integral(1/10, x, 6, 10))
print(integral(1/10, x, 3, 8))
```

```
3/10 # 0.3      P(X < 3)
2/5  # 0.4      P(X > 6)
1/2  # 0.5      P(3 < X < 8)
```

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
print(punif(3, min = 0, max = 10, lower.tail = TRUE))
print(1 - punif(6, min = 0, max = 10, lower.tail = TRUE))
punif(8, min = 0, max = 10, lower.tail = TRUE) - punif(3, min = 0, max = 10, lower.tail = TRUE)
```

```
[1] 0.3      # P(X < 3)
[1] 0.4      # P(X > 6)
[1] 0.5      # P(3 < X < 8)
```

■

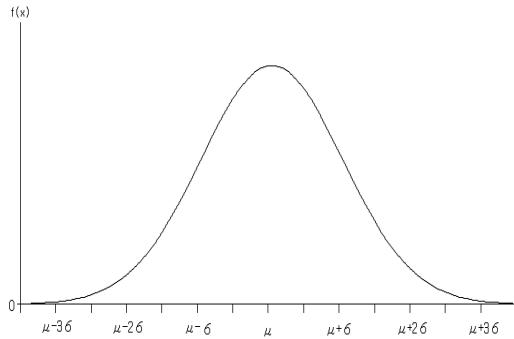
■ 정규분포(Normal distribution) $N(\mu, \sigma^2)$

(1) 확률변수 X 의 확률밀도함수가 다음과 같을 때, X 는 정규분포(normal distribution)를 따른다고 하고 $X \sim N(\mu, \sigma^2)$ 으로 표시한다.

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad -\infty < x < \infty$$

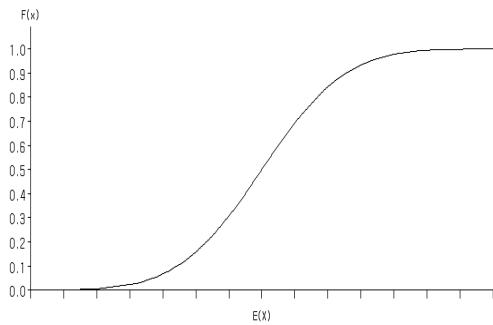
정규분포의 확률밀도함수 $f(x)$ 가 그리는 아래 그래프를 정규분포곡선(normal

distribution curve) 이라 한다.



(2) $X \sim N(\mu, \sigma^2)$ 일 때, 정규분포의 확률분포함수는 다음과 같다.

$$F(x) = P(X \leq x) = \int_{-\infty}^x \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(t-\mu)^2}{2\sigma^2}} dt$$



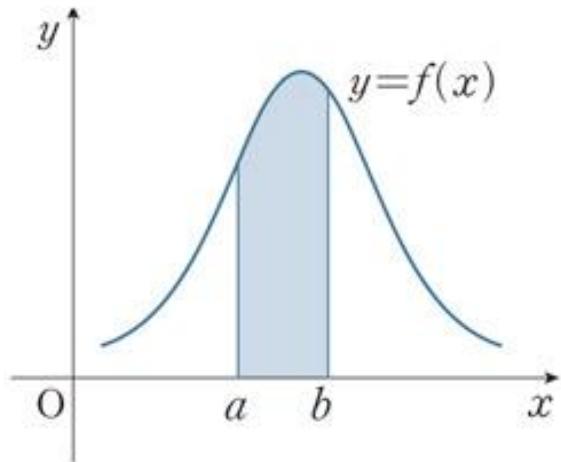
(3) $X \sim N(\mu, \sigma^2)$ 일 때, 정규분포의 기댓값과 분산은 다음과 같다.

$$E(X) = \mu, \quad V(X) = \sigma^2$$

(4) 확률변수 X 가 정규분포를 따를 때, X 가 a 이상 b 이하의 값을 취할 확률은

$$P(a \leq X \leq b) = \int_a^b f(x) dx$$

이때 이것은 바로 아래쪽 그림의 색칠한 부분의 넓이를 의미한다.



(5) 정규분포를 따르는 모집단에서 임의로 표본을 택하는 코드는 다음과 같다. 예를 들어, $\mu = 10$ 이고 $\sigma = 2$ 인 정규분포를 따르는 $N(10, 2^2)$ 인 모집단에서 표본 100개를 추출하려면 다음을 사용하면 된다.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
rnorm(100, mean = 10, sd = 2)
```

```
[1] 4.993634 10.853598 11.534801 7.972298 11.064551 10.598454 11.649311 11.133513 7.039146
[10] 11.687325 7.986739 14.007140 9.843295 12.698029 9.209033 10.520834 11.686081 10.824240
[19] 9.023518 12.356724 8.898054 7.234926 7.521364 14.499351 10.923138 6.566358 10.651991
[28] 12.188283 10.206761 7.902254 9.861373 12.544210 8.257245 10.320291 9.724923 8.356419
[37] 12.453693 10.580581 9.524559 6.396732 10.828679 9.522431 9.688252 8.420539 7.779511
[46] 6.587484 10.874041 12.091645 10.240461 11.466272 7.087663 10.003768 7.513210 9.436918
[55] 8.905059 11.507355 7.415323 12.922770 12.710045 10.184638 9.176710 9.311333 11.232487
[64] 11.027508 12.320341 7.745402 11.041632 11.202462 11.393217 4.395834 12.739655 10.115566
[73] 8.053541 10.414316 15.644675 10.518735 12.703266 10.648767 10.114506 10.446217 9.877279
[82] 7.485706 10.100924 8.614479 7.848015 10.256710 11.351979 6.247603 10.813502 8.855665
[91] 9.725255 7.850070 3.618453 9.633205 9.443438 8.911703 13.243177 11.834099 11.776822
[100] 9.576924
```

■ 정규분포곡선은 아래 성질을 갖는다.

정리. 정규분포 곡선의 성질

- (1) 형태는 종(Bell) 모양이다.
- (2) $f(x) \geq 0$ 이고 $x = \mu$ 을 축으로 좌우 대칭이다.
- (3) $x = \mu$ 에서 최대값 $\frac{1}{\sqrt{2\pi}\sigma}$ 을 갖고 그 점이 최빈수(mode, 가장 많이 관측되는 수)이다.
- (4) x 축을 점근선으로 갖는다. 단, $-\infty < x < \infty$
- (5) 변곡점은 $x = \mu \pm \sigma$ 이고 $x = \mu \pm 3\sigma$ 에서 거의 x 축과 접하게 된다.
- (6) 곡선과 수평축(x 축) 사이의 면적은 1이다.
- (7) $\mu - \sigma < x < \mu + \sigma$ 에서 아래로 오목하고 $x < \mu - \sigma, \mu + \sigma < x$ 에서 아래로 볼록하다.
- (8) σ 가 일정하고 μ 가 변화할 때 μ 가 크면 오른쪽으로, μ 가 작으면 왼쪽으로 이동한다.
- (9) μ 가 일정하고 σ 가 변화할 때 σ 가 크면 넓고 완만한 곡선, σ 가 작으면 좁고 뾰족한 곡선을 이룬다.

■ 표준정규분포(Standard normal distribution) $N(0,1)$

- (1) 확률변수 X 가 정규분포 $N(\mu, \sigma^2)$ 를 따를 때, 표준화 확률변수 $Z = \frac{X - \mu}{\sigma}$ 는 다음과 같은 확률밀도함수를 갖는다.

$$f(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} \quad -\infty < z < \infty$$

이 때, Z 는 표준정규분포(standard normal distribution)를 따른다고 하고, $Z \sim N(0,1)$ 로 표시한다.

- (2) $Z \sim N(0,1)$ 일 때, Z 의 확률분포함수 $F(z)$ 는

$$F(z) = P(Z \leq z) = \int_{-\infty}^z \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}} dt$$

이다. $F(z)$ 를 표준정규분포함수(standard normal distribution function)라 한다.

(3) $Z \sim N(0,1)$ 일 때, 표준정규분포의 기댓값과 분산은 다음과 같다.

$$E(Z) = 0, \quad V(Z) = 1$$

💡 정규분포의 표준화 [표준정규분포의 기댓값과 분산]

(1) 확률변수 X 의 평균이 μ , 표준편차가 σ 라 하면

$$E(Z) = E\left(\frac{X-\mu}{\sigma}\right) = E\left(\frac{X}{\sigma} - \frac{\mu}{\sigma}\right) = \frac{1}{\sigma}E(X) - \frac{\mu}{\sigma} = \frac{\mu}{\sigma} - \frac{\mu}{\sigma} = 0$$

$$V(Z) = V\left(\frac{X-\mu}{\sigma}\right) = V\left(\frac{X}{\sigma} - \frac{\mu}{\sigma}\right) = \frac{1}{\sigma^2}V(X) = \frac{\sigma^2}{\sigma^2} = 1$$

따라서 확률변수 $Z = \frac{X-\mu}{\sigma}$ 의 평균과 분산은 각각 0, 1이 된다.

$N(\mu, \sigma^2)$	표준화	$N(0,1)$
X 의 분포	$Z = \frac{X-\mu}{\sigma}$	Z 의 분포

(2) 확률변수 X 가 정규분포 $N(\mu, \sigma^2)$ 을 따르는 분포일 때, 확률 $P(a \leq X \leq b)$ 을

구하려면 변환 $Z = \frac{X-\mu}{\sigma}$ 에 의하여 표준화된 확률변수 Z 가 표준정규분포

$N(0,1)$ 에 따르는 분포를 이용하여 $P\left(\frac{a-\mu}{\sigma} \leq Z \leq \frac{b-\mu}{\sigma}\right)$ 을 구한다.

$$\frac{x-\mu}{\sigma} = z \text{ 라고 하면 } \frac{1}{\sigma} dx = dz$$

$$x \rightarrow a \Rightarrow z \rightarrow \frac{a-\mu}{\sigma}, \quad x \rightarrow b \Rightarrow z \rightarrow \frac{b-\mu}{\sigma}$$

$$P(a \leq X \leq b) = \int_a^b \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx = \int_{\frac{a-\mu}{\sigma}}^{\frac{b-\mu}{\sigma}} \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} dz = P\left(\frac{a-\mu}{\sigma} \leq Z \leq \frac{b-\mu}{\sigma}\right)$$

(3) 정규분포에서 많이 쓰이는 확률

$$P(\mu - \sigma \leq X \leq \mu + \sigma) = P(-1 \leq Z \leq 1) = 0.6826$$

$$P(\mu - 1.96\sigma \leq X \leq \mu + 1.96\sigma) = P(-1.96 \leq Z \leq 1.96) = 0.9500$$

$$P(\mu - 2\sigma \leq X \leq \mu + 2\sigma) = P(-2 \leq Z \leq 2) = 0.9544$$

$$P(\mu - 2.58\sigma \leq X \leq \mu + 2.58\sigma) = P(-2.58 \leq Z \leq 2.58) = 0.9902$$

$$P(\mu - 3\sigma \leq X \leq \mu + 3\sigma) = P(-3 \leq Z \leq 3) = 0.9974$$

예제 7 다음 물음에 답하여라.

(1) 확률변수 X 가 표준정규분포 $N(0, 1)$ 을 따를 때 $P(-3 \leq X \leq 3)$ 을 구하여라.

(2) 확률변수 X 가 $N(2, 25)$ 을 따를 때 $P(-3 \leq X \leq 8)$ 을 구하여라.

풀이].

$$\begin{aligned} (1) \quad P(-3 \leq X \leq 3) &= P\left(\frac{-3-0}{1} \leq Z \leq \frac{3-0}{1}\right) = P(-3 \leq Z \leq 3) \\ &= 2P(0 \leq Z \leq 3) = 2 \int_0^3 \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} dz \approx 2 \times 0.4987 = 0.9974 \end{aligned}$$

$$\begin{aligned} (2) \quad P(-3 \leq X \leq 8) &= P\left(\frac{-3-2}{5} \leq Z \leq \frac{8-2}{5}\right) = P(-1 \leq Z \leq 1.2) \\ &= P(0 \leq Z \leq 1) + P(0 \leq Z \leq 1.2) = 0.3413 + 0.3849 = 0.7262 \end{aligned}$$

[Sage code] <http://mathlab.knou.ac.kr:8080/> <http://sage.skku.edu/>

```
f(x) = 1/sqrt(2*pi)*e^(-x^2/2)
print(numerical_integral(f(x), -3, 3)[0])
print(numerical_integral(f(x), -1, 1.2)[0])
```

```
0.997300203937    # P(-3 ≤ X ≤ 3)
0.726275075847    # P(-3 ≤ X ≤ 8)
```

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
print(pnorm(3, mean = 0, sd = 1, lower.tail = TRUE) - pnorm(-3, mean = 0, sd = 1, lower.tail =
TRUE))
pnorm(8, mean = 2, sd = 5, lower.tail = TRUE) - pnorm(-3, mean = 2, sd = 5, lower.tail = TRUE)
```

```
[1] 0.9973002
```

[1] 0.7262751

예제 8 어떤 회사에서 제조되는 전구의 수명 시간은 정규분포를 따른다고 한다. 제조되는 전구들에 대한 수명시간에 대한 평균은 3,000시간이고 표준편 차는 80시간이라 할 때, 임의로 선택한 전구 1개의 수명시간이 2,948시간에서 3,080시간 사이일 확률을 구하여라.

[출처] <http://matrix.skku.ac.kr/2018-album/R-Sage-Stat-Lab-2.html>

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

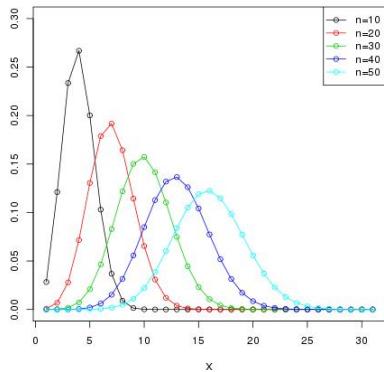
```
mu = 3000
sigma = 80
x1 = 2948
x2 = 3080
print(pnorm(x2, mean=mu, sd=sigma) - pnorm(x1, mean=mu, sd=sigma))
z1 = (x1-mu)/sigma
z2 = (x2-mu)/sigma
pnorm(z2) - pnorm(z1)
```

[1] 0.5834986

[1] 0.5834986

💡 이항분포의 정규근사

확률변수 X 의 분포가 이항분포일 때 X 의 평균은 np , 분산은 npq 이므로 X 는 정규분포 $N(np, npq)$ 에 근사 한다. 표준화한 변수 $Z = \frac{X - np}{\sqrt{npq}}$ 는 표준정규분포 $N(0,1)$ 에 가까워진다.



예제 9 확률변수 X 가 $n = 150$, $p = 0.6$ 의 이항분포를 따른다고 하자. 이러한 확률변수 X 가 82이상 102미만일 확률은 얼마인가?

[출처] <http://matrix.skku.ac.kr/2018-album/R-Sage-Stat-Lab-2.html>

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```

n = 150
p = 0.6
print(pbinom(101, n, p) - pbinom(81, n, p))
mu = n*p
sigma = sqrt(n*p*(1-p))
pnorm(101.5, mu, sigma) - pnorm(81.5, mu, sigma)

```

```

[1] 0.8945357    # 이항분포
[1] 0.8940697    # 정규분포

```

■ 지수분포(Exponential distribution) $G(1, 1/\lambda)$

(1) 사건이 서로 독립적일 때, 단위 시간동안 발생하는 사건의 횟수가 푸아송 분포를 따른다면, 다음 사건이 일어날 때까지 대기 시간은 지수분포(Exponential distribution)를 따른다고 한다.

(2) 확률변수 X 가 지수분포를 따를 때, 확률밀도함수는 다음과 같이 주어진다.

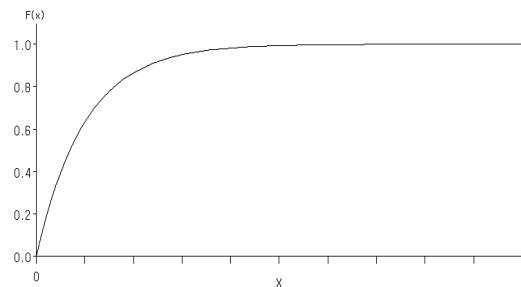
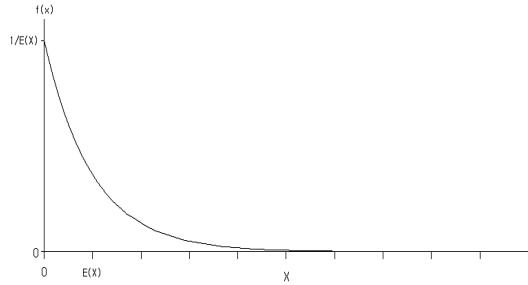
$$f(x) = \begin{cases} \lambda e^{-\lambda x} & (\lambda > 0) \quad 0 < x < \infty \\ 0 & x \leq 0 \end{cases}$$

이때 λ 는 단위 시간당 발생하는 사건의 빈도를 나타내며, $X \sim G(1, \frac{1}{\lambda})$ 으로 표시한다.

(3) $X \sim G(1, \frac{1}{\lambda})$ 일 때, X 의 확률분포함수는 다음과 같다.

$$F(x) = \begin{cases} 1 - e^{-\lambda x} & (\lambda > 0) \quad 0 < x < \infty \\ 0 & x \leq 0 \end{cases}$$

예를 들어, $\lambda = 1$ 일 때, 지수분포의 확률밀도함수와 확률분포함수의 그래프는 다음과 같다.



(4) $X \sim G(1, \frac{1}{\lambda})$ 일 때, 확률변수 X 의 기댓값과 분산은 다음과 같다.

$$E(X) = \frac{1}{\lambda}, \quad V(X) = \frac{1}{\lambda^2}$$

* $E(X) = \frac{1}{\lambda}$ 는 단위 시간에 사건이 λ 번 발생할 때, 사건과 사건 사이에 평균적인

대기 시간이 $\frac{1}{\lambda}$ 임을 뜻한다.

(5) 지수분포는 부품이나 시스템의 고장시간에 대한 모형으로 사용되고, 신뢰성 이론, 기대 시간, 수명 시간 등의 분포에서 이용되며, 지수분포는 종종 어떤 특정한 사건이 발생할 때 까지 걸린 시간의 분포에서 사용 한다. 예를 들어, 지금부터 지진이 발생할 때까지의 시간, 새로운 전쟁이 일어날 때까지의 시간, 잘못 걸려온 전화를 받을 때까지의 시간 등에서 이용된다.

(6) 지수분포를 따르는 모집단에서 임의로 표본을 택하는 코드는 다음과 같다. 예를 들어,

$G(1, \frac{1}{3})$ 인 모집단에서 표본 100개를 추출하려면 다음을 사용하면 된다.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
rexp(100, rate = 3)
```

```
[1] 0.009637094 0.028599375 0.220575222 0.881992402 0.107908586 0.174507394 0.289087504  
[8] 0.144788279 0.056751222 0.157662599 0.418217215 0.297680454 0.030597738 0.308571341  
[15] 0.069470179 1.985052134 0.008106374 0.006782367 0.389050356 0.022559032 0.471607458  
[22] 0.024061740 0.290314284 0.257322722 0.351029982 0.557251844 0.761589695 0.374779185  
[29] 0.241116442 0.219590702 0.071937825 0.332210956 0.583000477 0.091549826 0.457057041  
[36] 0.211143317 0.337828456 1.280247700 0.022465112 0.600578774 0.397424868 0.058848894  
[43] 0.165956474 0.226836211 0.378891041 0.125913506 0.681248719 0.215952364 0.143921471  
[50] 0.090602909 0.096559436 0.358013336 0.170499277 0.068396033 0.238725951 0.066493922  
[57] 0.041184089 0.156303674 0.094957141 0.932155570 0.072865179 0.389643479 0.349312843  
[64] 0.079083657 0.153782450 0.853548589 0.101173403 0.364976998 0.166433293 0.061507393  
[71] 0.167637501 0.116266113 0.002271554 0.530852424 0.400505415 0.210143337 0.316803445  
[78] 0.098759557 0.597583457 0.014970370 1.070611700 0.831700200 0.142613613 0.393174229  
[85] 0.564409405 0.494550440 0.134004474 0.236286590 1.285781847 0.364512329 0.676487528  
[92] 0.103572963 0.146961554 1.107290171 0.210892605 1.015288333 1.236282291 0.000903822  
[99] 0.229512015 0.128073351
```

예제 10 어떤 회사의 고객센터에는 전화가 평균적으로 10분에 3회가 걸려온다.
다음 물음에 답하여라.

- (1) 한 번 전화가 걸려온 후에 다음 전화가 걸려올 때까지 걸린 시간이 2분 이내일 확률
- (2) 다음 전화가 걸려올 때까지 걸린 시간이 5분 이상일 확률

풀이. 고객센터에는 평균적으로 1분당 0.3회 전화가 걸려오므로 전화가 한번 걸려온 후 다음 전화가 걸려올 때까지 걸린 시간은 $\lambda = 0.3$ 인 지수분포 $G(1, \frac{1}{0.3})$ 를 따른다. 따라서 R 코드를 이용하여 확률을 구하면 다음과 같다.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

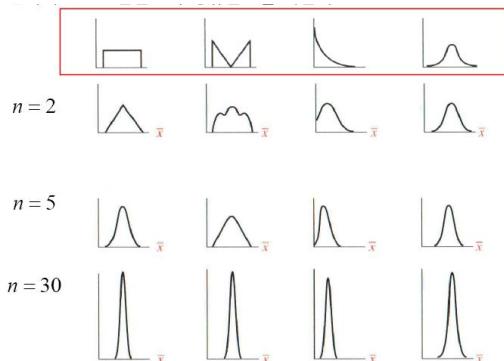
```
print(pexp(2, rate = 0.3, lower.tail = TRUE, log.p = FALSE))
1 - pexp(5, rate = 0.3, lower.tail = TRUE, log.p = FALSE)
```

```
[1] 0.4511884 # 한 번 전화가 걸려온 후에 다음 전화가 걸려올 때까지 걸린 시간이 2분 이내일 확률
[1] 0.2231302 # 다음 전화가 걸려올 때까지 걸린 시간이 5분 이상일 확률 ■
```

■ 중심극한정리 (central limit theorem, CLT)

* 중심극한정리는 통계학 역사상 가장 중요한 발견이라고 한다. 프랑스 수학자 라플라스 (Laplace)는 1774년에서 1786년 사이 몇 개의 논문에서 이러한 정리의 발견과 증명을 시도하였다. 여러 통계분석기법에서 분석모형이 정규분포를 따른다는 가정을 하는 경우가 많다. 그 근거가 바로 중심극한정리이다. 이 정리는 확률, 통계 이론뿐만 아니라 실용적인 면에서도 의미가 크며, 품질관리에서도 많이 이용된다.

중심극한정리(Central Limit Theorem, CLT)란 동일한 확률분포를 가진 독립 확률변수 n 개의 평균의 분포는 n 이 적당히 크다면 정규분포에 가까워진다는 이론이다. 즉, 평균이 μ 이고 분산이 σ^2 인 모집단으로부터 추출한 크기가 n 인 확률표본의 표본 평균 \bar{X} 는 n 이 증가할수록 모집단의 분포유형에 상관없이 많은 경우는 극사적으로 정규분포 $N\left(\mu, \frac{\sigma^2}{n}\right)$ 을 따른다는 것이다.



이를 정리하면 다음과 같다.

정리. 표본평균의 확률분포와 중심극한정리

(1) 정규 모집단으로부터의 표본평균에 대한 확률분포

크기가 n 인 확률표본 X_1, X_2, \dots, X_n 이

$$X_i \sim N(\mu, \sigma^2), i = 1, \dots, n$$

이면, 표본평균 \bar{X} 의 확률분포는 다음과 같은 정규분포를 따르게 된다.

$$\bar{X} \sim N\left(\mu, \frac{\sigma^2}{n}\right)$$

(2) 중심극한정리 [Key Idea 3]

평균이 μ 이고 분산이 σ^2 인 모집단으로부터 추출한 크기 n 인 확률표본의 표본평균 \bar{X} 는 표본의 크기가 큰 경우(보통 30 이상), 근사적으로 평균이 μ 이고 분산이 σ^2/n 인 정규분포를 따른다.

[참고] <https://wsyang.com/2011/04/clt-with-r/> https://ko.wikipedia.org/wiki/중심_극한_정리

예제 11 만 2세 유아들의 신장은 평균이 87.6cm이고 표준편차가 3.3cm인 정규 분포를 따른다고 한다. 만약 6명의 2세 유아들에 대해 신장을 조사하였을 경우, 이들의 평균 신장이 86.6cm에서 89.4cm사이일 확률은 얼마인가?

[출처] <http://matrix.skku.ac.kr/2018-album/R-Sage-Stat-Lab-2.html>

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
mu = 87.6
sigma = 3.3
n = 6
x1 = 86.6
x2 = 89.4
print(pnorm(x2, mu, sigma/sqrt(n)) - pnorm(x1, mu, sigma/sqrt(n)))
z1 = (x1-mu)/(sigma/sqrt(n))
z2 = (x2-mu)/(sigma/sqrt(n))
pnorm(z2) - pnorm(z1)
```

[1] 0.6802773

[1] 0.6802773

예제 12 어떤 열 매의 무게는 평균이 0.5g 이고 표준편차가 0.15g 이라고 한다. 100개의 열매를 임의로 선택한 경우, 이들의 평균 무게가 0.48g 에서 0.53g 사이일 확률은 얼마인가?

[출처] <http://matrix.skku.ac.kr/2018-album/R-Sage-Stat-Lab-2.html>

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
mu = 0.5
sigma = 0.15
n = 100
x1 = 0.48
x2 = 0.53
print(pnorm(x2, mu, sigma/sqrt(n)) - pnorm(x1, mu, sigma/sqrt(n)))
z1 = (x1-mu)/(sigma/sqrt(n))
z2 = (x2-mu)/(sigma/sqrt(n))
pnorm(z2) - pnorm(z1)
```

```
[1] 0.8860386
[1] 0.8860386
```

■ 확률과 통계에서 중심극한정리(Central Limit Theorem)가 의미하는 것은 (대부분의 경우) 모집단의 분포가 연속형이든, 이산형이든, 또는 한쪽으로 치우친 형태이든 간에 표본의 크기가 클수록 표본평균의 분포가 점점 정규분포에 가까워진다는 의미이다. 아래 R 코드를 통하여 확인할 수 있다.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
CLT.plot <- function(r.dist, n, ...) {
means <- double()
# 사이즈 n의 샘플을 500회 생성하여 표본평균을 계산
for(i in 1:1000) means[i] = mean(r.dist(n,...))
# 표본평균을 표준화
std.means <- scale(means)

# 플롯의 파라메터 설정(2개의 플롯을 한 화면에)
```

```

par(mfrow = c(1, 2))

# 히스토그램과 표본의 밀도
hist(std.means, prob = T, col = "light grey", border = "grey", main = NULL, ylim = c(0,
0.5))
lines(density(std.means))
box()

# 표준정규분포 곡선
curve(dnorm(x, 0, 1), -3, 3, col = 'blue', add = T)

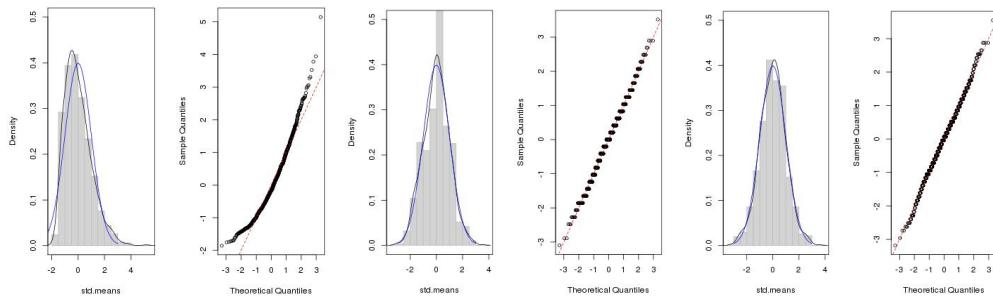
# Q-Q plot
qqnorm(std.means, main="", cex = 0.8)
abline(0, 1, lty = 2, col = "red")
par(mfrow = c(1, 1))
}

# 카이제곱분포
CLT.plot(rchisq, n = 10, df = 1)
CLT.plot(rchisq, n = 30, df = 1)
CLT.plot(rchisq, n = 50, df = 1)

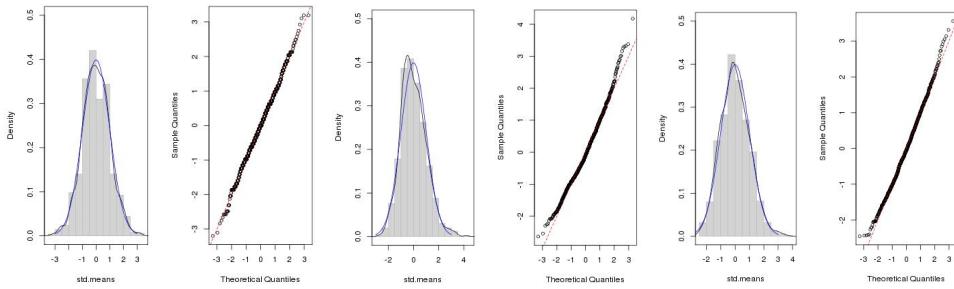
# 이항분포
CLT.plot(rbinom, n = 10, size = 10, p = .5)
CLT.plot(rbinom, n = 30, size = 10, p = .5)
CLT.plot(rbinom, n = 50, size = 10, p = .5)
dev.off()

```

카이제곱(χ^2) 분포



이항(Binomial) 분포



여기서 표본의 크기 n 이 커질수록 표본평균의 분포가 과단색 실선인 정규 분포에 점점 가까워져 감을 알 수 있다. 그리고 오른쪽에 있는 Q-Q(이론상의 Quantile vs. 획득한 샘플값의 Quantile) 플롯은 분석할 표본 데이터의 분포와 정규분포의 분포 형태를 비교하여 표본 데이터가 정규분포를 따르는지 (즉, red line과 중간 부분에서 오차가 없는지) 검사하는 간단한 시각적 도구가 된다.

중심극한정리가 잘 작동하는지 R을 활용하여 살펴보자.

예제 13 균등분포 $U(0, 1)$ 에서 크기가 $n = 1, 4, 16, 64, 256$ 인 임의표본의 평균을 각 n 에 대하여 1000번씩 계산하여 히스토그램과 QQ-plot을 작성하라. (아래 R 코드 참조) 표본평균이 정규분포에 근접하려면 n 은 얼마나 커야 하는가? 마찬가지로 표본의 분산을 구해보고, n 이 커질 때 이 값은 어떻게 변화하는지 토의하라.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
set.seed(222)
ITER <- 1000
N <- c(1, 4, 16, 64, 256) # 샘플의 크기
variances = rep(NA, length(N)) # 평균들의 분산을 저장
par(mfrow = c(1,2)) # 하나의 창에 2개 그림 동시에 그리기
for(i in 1:length(N)){
  # Unif(0, 1)에서 1000 x N[i] 행렬을 얻기
  unif.mat <- matrix(runif(ITER*N[i], 0, 1), nrow = ITER, ncol = N[i], byrow = T)
  # 각 행의 평균
```

```

sample.means <- apply(unif.mat, MARGIN = 1, FUN = mean)
# 표본 평균의 분산
variances[i] <- var(sample.means)
# 히스토그램 만들기
hist(sample.means, main = eval(paste("표본 평균, n = ", N[i])),
     xlab = expression(bar(X)), xlim=c(0, 1), probability = T)
# q-q plot 그리기
qqnorm(sample.means, ylim=c(0, 1)); qqline(sample.means, col = "red")
}

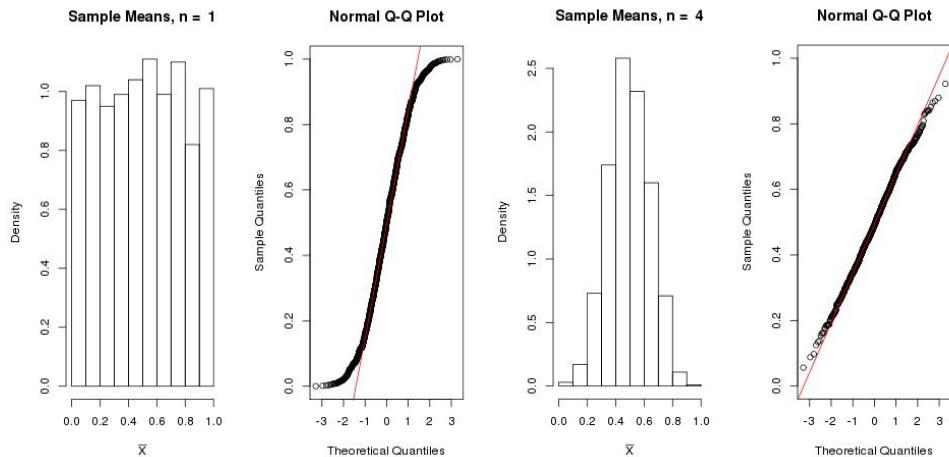
print(variances)
print(1/(12*N))
dev.off()

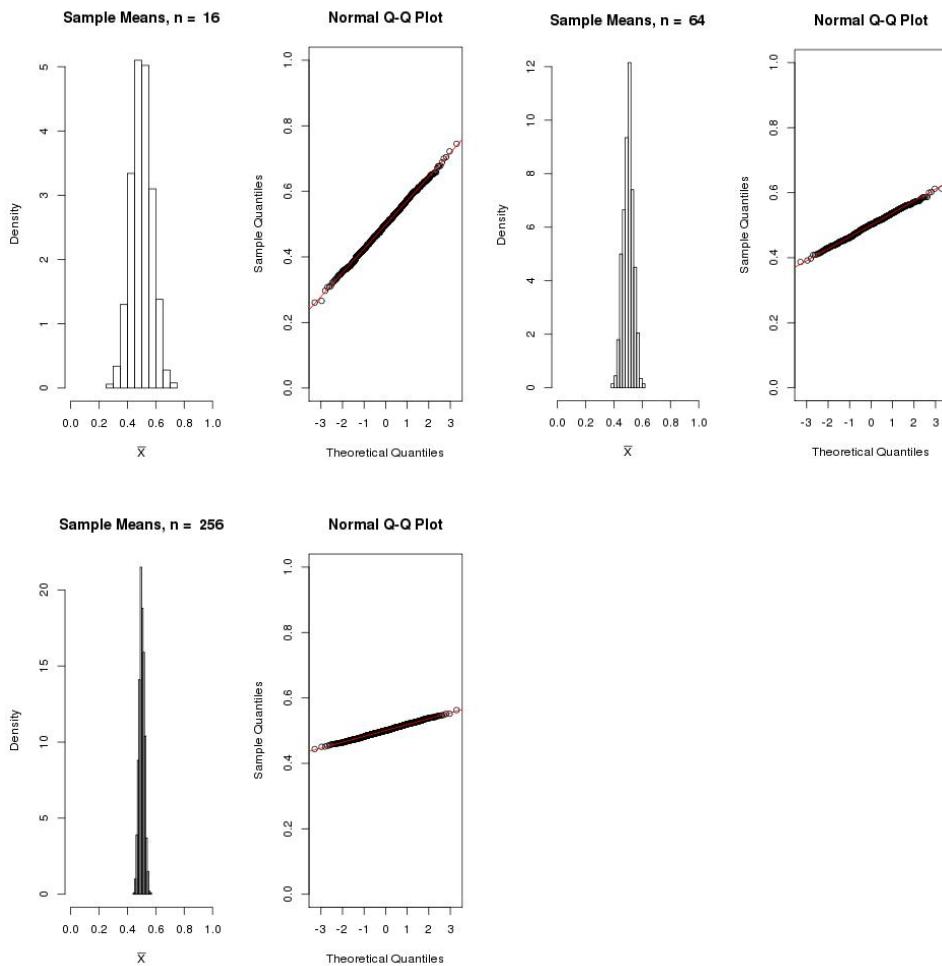
```

```

[1] 0.0808451462 0.0211910408 0.0054114866 0.0013017499 0.0003425692
[1] 0.0833333333 0.0208333333 0.0052083333 0.0013020833 0.0003255208

```

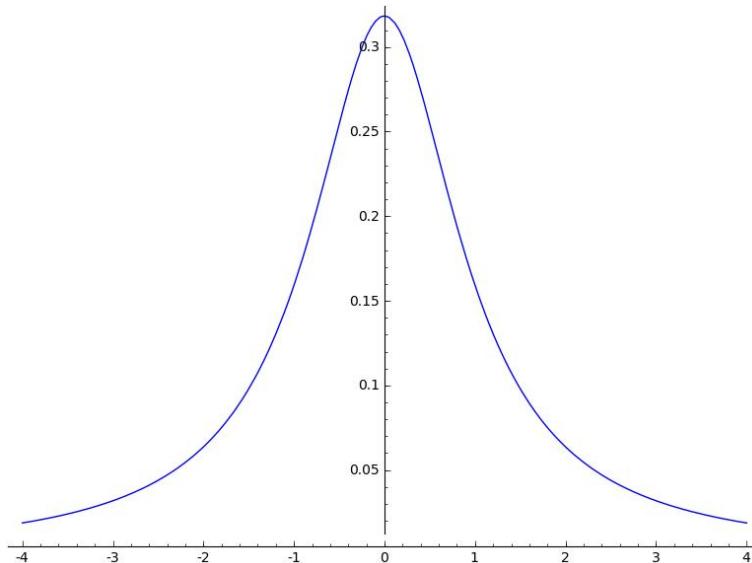




그러나 아래 예인 <표준 코시 분포(standard Cauchy distribution)>의 경우와 같이 조건이 달라서 ‘표본의 크기 n 이 커져도 표본평균의 분포가 정규 분포에 가까지지 않을 수 있다’. (적분값인) 적률생성함수가 존재하지 않는 코시분포의 경우 평균과 분산도 갖지 않는다.

예제 14 확률변수 X 의 확률밀도함수가 다음과 같을 때 X 는 표준 Cauchy 분포 (standard Cauchy distribution)를 따른다고 하며, $X \sim \text{Cauchy}(1, 0)$ 와 같이 나타낸다.

$$f(x) = \frac{1}{\pi(1+x^2)}, \quad -\infty < x < \infty$$



$X \sim \text{Cauchy}(1, 0)$ 에서 크기가 $n = 1, 4, 16, 64, 256$ 인 임의표본의 평균을 각 n 에 대하여 1000번씩 계산하여 히스토그램과 qq-plot을 작성하라.
 (아래 R 코드 참조) n 이 커질수록 표본평균의 분포가 정규분포에 근접하는가? 마찬가지로 표본의 분산을 구해보고, n 이 커질 때 이 값은 어떻게 변화하는지 토의하라.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
set.seed(222)
ITER <- 1000
N <- c(1, 4, 16, 64, 256) # 샘플의 크기
variances = rep(NA, length(N)) # 평균들의 분산을 저장
par(mfrow = c(1,2)) # 하나의 창에 2개 그림 동시에 그리기
for(i in 1:length(N)){
  # 표준 Cauchy 분포에서 1000 x N[i] 행렬을 얻기
  cauchy.mat <- matrix(rcauchy(ITER*N[i], 0, 1), nrow = ITER, ncol = N[i], byrow
  = T)
```

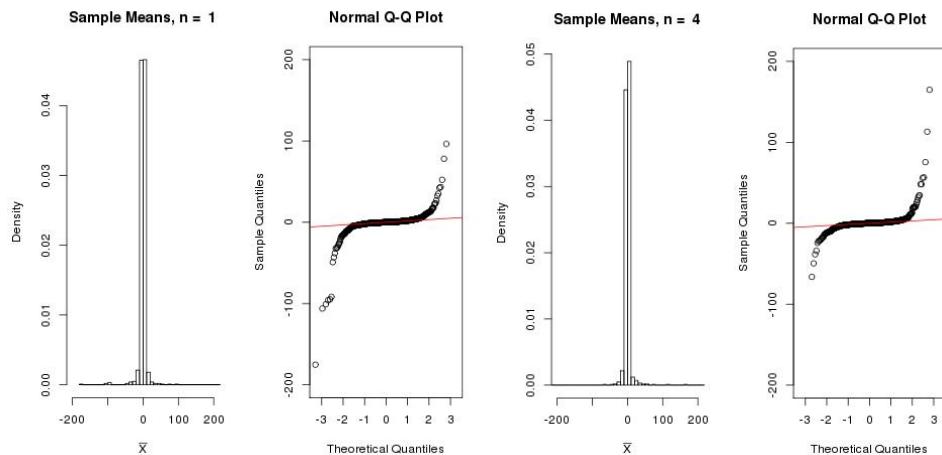
```

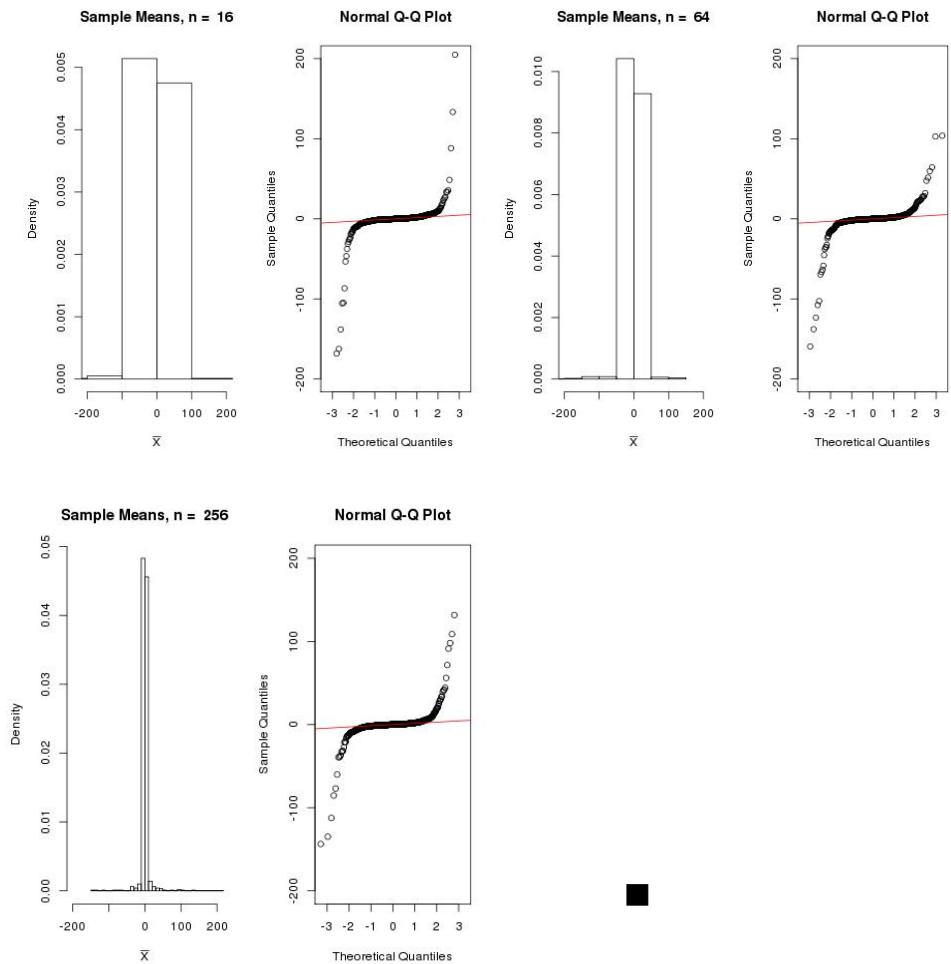
# 각 행의 평균
sample.means <- apply(cauchy.mat, MARGIN = 1, FUN = mean)
# 표본 평균의 분산
variances[i] <- var(sample.means)
# 히스토그램 만들기
hist(sample.means, main = eval(paste("표본 평균, n = ", N[i])),
      xlab = expression(bar(X)), xlim=c(-200, 200), probability = T,
      breaks=50*i)
# q-q plot 그리기
qqnorm(sample.means, ylim=c(-200, 200)); qqline(sample.means, col = "red")
}

print(variances)
dev.off()

```

[1] 372.4018 1752.4899 108397.3767 108332.3578 11285.3542





3.3 결합 확률분포, 공분산(Covariance)과 상관계수(Correlation)

23강 동영상, 공분산과 상관계수, 데이터 활용의 실제 <https://youtu.be/oUSPhkyEWp4> (50:05)

* 확률변수가 두 개 이상 있는 경우에는 각각의 확률변수에 대한 확률분포 이외에도 확률분포 쌍이 가지는 복합적인 확률분포를 살펴보아야 한다. 두 확률변수 값의 쌍이 어떤 확률분포를 가지는지 안다면 둘 중 하나의 확률분포의 값을 알고 있을 때 다른 확률분포가 어떻게 되는지도 알 수 있다. 이를 위하여 결합 확률분포(또는 결합분포)에 대한 개념이 필요하다.

정의. 결합 확률함수, 결합 확률분포

(1) X 와 Y 가 이산확률변수이면 X 와 Y 의 결합 확률함수(joint probability function)는 다음과 같다.

$$p_{ij} = p(x_i, y_j) = P(X=x_i, Y=y_j) \quad (x_i, y_j \in \mathbb{R}, i=1, \dots, s, j=1, \dots, t)$$

(2) X 와 Y 의 가능한 모든 값에 대하여 $p(x_i, y_j)$ 의 값을 나타낸 것을 결합 확률분포(joint probability distribution)라 한다. 이를 표로 나타내면 다음과 같다.

X	y_1	y_2	\cdots	y_t		합
x_1	p_{11}	p_{12}	\cdots	p_{1t}	→	$P(X=x_1)$
x_2	p_{21}	p_{22}	\cdots	p_{2t}	→	$P(X=x_2)$
\vdots	\vdots	\vdots	\ddots	\vdots	→	\vdots
x_s	p_{s1}	p_{s2}	\cdots	p_{st}	→	$P(X=x_s)$

합	$P(Y=y_1)$	$P(Y=y_2)$	\cdots	$P(Y=y_t)$	↓	1
---	------------	------------	----------	------------	---	---

Y 에 관한 주변확률분포

(3) X, Y 의 결합분포가 주어져 있을 때 주변확률분포(marginal probability distribution)는 다음과 같이 정의된다.

$$P(X=x_i) = \sum_{j=1}^t P(X=x_i, Y=y_j) = \sum_{j=1}^t p_{ij}$$

$$P(Y=y_j) = \sum_{i=1}^s P(X=x_i, Y=y_j) = \sum_{i=1}^s p_{ij}$$

즉, 주변분포란 결합 확률분포에서 하나의 확률변수에 대해서만 고려한 확률분포를 뜻한다.

* 결합 확률분포에 관하여 다음이 성립한다.

정리. 결합 확률분포

- (1) 모든 x_i, y_j 에 대하여 $p(x_i, y_j) \geq 0 \quad i, j = 1, 2, \dots$
- (2) 모든 x_i, y_j 에 대하여 $p(x_i, y_j)$ 의 합은 1이다. 즉, $\sum_{i=1}^{\infty} \sum_{j=1}^{\infty} p(x_i, y_j) = 1$
- (3) 모든 x_i, y_j 에 대하여 $P(a < X < b, c < Y < d) = \sum_{a < x < b} \sum_{c < y < d} p(x_i, y_j)$

예제 15 크기가 같은 파란 색 공 3개와 붉은 색 공 2개와 녹색 공 3개가 한 주머니 안에 들어 있다. 이 주머니에서 임의로 2개의 공을 꺼낸다. 꺼낸 공(ball) 중 파란색 공의 수를 X , 붉은 색 공의 수를 Y 라 할 때 다음 물음에 답하여라.

- ① X 와 Y 의 결합 확률함수를 구하여라.
- ② 결합 확률분포를 작성하여라.
- ③ $P(X+Y \leq 1)$ 을 구하여라.
- ④ X 의 주변분포를 구하여라.
- ⑤ Y 의 주변분포를 구하여라.

풀이.

$$\textcircled{1} \quad p(x_i, y_j) = \frac{{}^3C_x {}_2C_y {}_3C_{2-x-y}}{{}_8C_2} \quad (x=0, 1, 2 : y=0, 1, 2 : 0 \leq x+y \leq 2)$$

$$\textcircled{2} \quad P(X=0, Y=0) = \frac{{}^3C_0 {}_2C_0 {}_3C_2}{{}_8C_2} = \frac{3}{28}$$

$$P(X=0, Y=1) = \frac{{}^3C_0 {}_2C_1 {}_3C_1}{{}_8C_2} = \frac{6}{28} = \frac{3}{14}$$

$$P(X=1, Y=0) = \frac{{}^3C_1 {}_2C_0 {}_3C_0}{{}_8C_2} = \frac{9}{28}$$

$$P(X=1, Y=1) = \frac{{}^3C_1 {}_2C_1 {}_3C_0}{{}_8C_2} = \frac{6}{28} = \frac{3}{14}$$

⋮

$X \backslash Y$	0	1	2	합
0	$\frac{3}{28}$	$\frac{3}{14}$	$\frac{1}{28}$	$\frac{5}{14} = P(X=0)$
1	$\frac{9}{28}$	$\frac{3}{14}$	0	$\frac{15}{28}$
2	$\frac{3}{28}$	0	0	$\frac{3}{28}$
합	$\frac{15}{28} = P(Y=0)$	$\frac{3}{7}$	$\frac{1}{28}$	1

$$\textcircled{3} \quad P(X+Y \leq 1) = p(0,0) + p(0,1) + p(1,0) = \frac{3}{28} + \frac{3}{14} + \frac{9}{28} = \frac{9}{14}$$

$$\textcircled{4} \quad P(X=0) = p(0,0) + p(0,1) + p(0,2) = \frac{3}{28} + \frac{3}{14} + \frac{1}{28} = \frac{5}{14}$$

$$P(X=1) = p(1,0) + p(1,1) + p(1,2) = \frac{9}{28} + \frac{3}{14} + 0 = \frac{15}{28}$$

$$P(X=2) = p(2,0) + p(2,1) + p(2,2) = \frac{3}{28} + 0 + 0 = \frac{3}{28}$$

X	0	1	2	합
$P(X=x_i)$	$\frac{5}{14}$	$\frac{15}{28}$	$\frac{3}{28}$	1

$$\textcircled{5} \quad P(Y=0) = p(0,0) + p(1,0) + p(2,0) = \frac{3}{28} + \frac{9}{28} + \frac{3}{28} = \frac{15}{28}$$

$$P(Y=1) = p(0,1) + p(1,1) + p(2,1) = \frac{3}{14} + \frac{3}{14} + 0 = \frac{3}{7}$$

$$P(Y=2) = p(0,2) + p(1,2) + p(2,2) = \frac{1}{28} + 0 + 0 = \frac{1}{28}$$

Y	0	1	2	합
$P(Y=y_j)$	$\frac{15}{28}$	$\frac{3}{7}$	$\frac{1}{28}$	1

* 여러 개의 연속확률변수에 대하여는 다음과 같은 결합밀도함수를 이용하여 결합 확률분포를 나타낸다. (여기서 중적분의 지식이 필요하다.)

정의. 결합밀도함수(joint density function)

연속확률변수 X 와 Y 의 결합밀도함수(joint density function) $f(x, y)$ 는 다음과 같이 정의된다.

(1) 모든 x, y 에 대하여 $f(x, y) \geq 0$ 이다.

(2) 모든 x, y 에 대하여 $\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) dx dy = 1$ 이다.

(3) 모든 x, y 에 대하여 $P(a < X < b, c < Y < d) = \int_c^d \int_a^b f(x, y) dx dy$ 이다.

(4) (X, Y) 가 xy 평면상의 임의의 영역 A 에 들어갈 확률은

$$P\{(X, Y) \in A\} = \int \int_A f(x, y) dx dy$$

로 주어진다.

(5) X 와 Y 의 주변밀도함수(marginal probability density function)는 각각 다음과 같이 정의된다.

$$f_X(x) = \int_{-\infty}^{\infty} f(x, y) dy, \quad f_Y(y) = \int_{-\infty}^{\infty} f(x, y) dx$$

예제 16 두 확률변수 X, Y 의 결합밀도함수가 다음과 같이 주어져 있다고 하자.

$$f(x, y) = x + y, \quad 0 < x < 1, \quad 0 < y < 1$$

X 와 Y 의 주변밀도함수를 각각 구하여라.

풀이. 정의에 의해 구하면 다음과 같다.

$$f_X(x) = \int_0^1 (x + y) dy = \left[xy + \frac{1}{2} y^2 \right]_0^1 = x + \frac{1}{2}, \quad 0 < x < 1$$

$$f_Y(y) = \int_0^1 (x + y) dx = \left[\frac{1}{2} x^2 + xy \right]_0^1 = y + \frac{1}{2}, \quad 0 < y < 1$$

[Sage code] <http://mathlab.knou.ac.kr:8080/> <http://sage.skku.edu/>

```
var('x, y')
f(x, y) = x + y
print("f_X(x) =", integral(f(x, y), y, 0, 1)) # X의 주변밀도함수
print("f_Y(y) =", integral(f(x, y), x, 0, 1)) # Y의 주변밀도함수

f_X(x) = x + 1/2      # X의 주변밀도함수
f_Y(y) = y + 1/2      # Y의 주변밀도함수
```

■

* 여러 확률변수 사이의 상관관계를 숫자로 나타낸 것이 공분산과 상관계수이다.

* 확률변수 X 가 갖는 분포를 이해하기 위해 사용하는 것이 첫 번째로 평균이고, 두 번째로 분산이다. 평균을 이용하여 분포의 중간부분을 알아내고, 분산으로써 분포가 얼마나 퍼져있는지 알아낸다.

- 그렇다면 확률변수가 2개일 때 이 확률분포들이 어떤 모양으로 되어있는지를 알고 싶으면, 가장 먼저 X 의 평균, 다음이 Y 의 평균을 생각하면 된다. 그 다음으로 얼마나 퍼져있는지 알기위해 분산 특히 각 확률변수들이 어떻게 퍼져있는지를 보여주는 개념이 공분산(Covariance)이다.
- 공분산은 X 의 편차와 Y 의 편차를 곱한 것의 평균이다. 그런데 공분산에도 X 와 Y 의 단위의 크기에 영향을 받는다는 문제점이 있다. 이것을 보완하기 위해 상관계수(Correlation)를 사용한다. 상관계수라는 개념은 확률변수의 절대 크기에 영향을 받지 않도록 표준화 시킨 것이고, 분산의 크기만큼 나누었다고 생각하면 된다.

정의. 공분산, 상관계수

(1) 확률변수 X 와 Y 의 공분산은 다음과 같이 정의된다.

$$Cov(X, Y) = \sigma_{xy} = E[(X - \mu_x)(Y - \mu_y)] = E(XY) - \mu_x\mu_y$$

(2) 확률변수 X 와 Y 사이의 상관계수는 다음과 같이 정의된다.

$$Corr(X, Y) = \rho = \frac{E[(X - \mu_x)(Y - \mu_y)]}{\sqrt{E(X - \mu_x)^2} \cdot \sqrt{E(Y - \mu_y)^2}}$$

■ 공분산 행렬(Covariance Matrix)

행렬을 이용하면 두 변수가 서로 어떤 관계를 가지는지를 쉽게 표현할 수 있다. 특히, 각 데이터의 분산과 공분산을 이용해 만드는 공분산 행렬이 이에 해당한다. n 개의 확률변수 x_1, \dots, x_n 를 하나의 벡터로 나타낸 것을 \mathbf{x} 라 하자. 벡터 \mathbf{x} 의 공분산 행렬(covariance matrix)은 (i, j) 성분이 $i \neq j$ 일 때는 \mathbf{x} 의 i 번째와 j 번째 확률변수의 공분산이고, $i = j$ 일 때는 i 번째 확률변수의 분산인 $n \times n$ 행렬을 말하는 것으로 $\Sigma = Var[\mathbf{x}]$ 를 사용하여 나타낸다. 쉽게 말하면, 정사각행렬의 값을 각 변수의 분산(주대각선)과 공분산으로 채운 것이 바로 공분산 행렬이다.

$$Var[\mathbf{x}] = \begin{bmatrix} var(x_1) & cov(x_1, x_2) & \cdots & cov(x_1, x_n) \\ cov(x_2, x_1) & var(x_2) & \cdots & cov(x_2, x_n) \\ \vdots & \vdots & \ddots & \vdots \\ cov(x_n, x_1) & cov(x_n, x_2) & \cdots & var(x_n) \end{bmatrix} = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1n} \\ \sigma_{21} & \sigma_{22} & \cdots & \sigma_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{n1} & \sigma_{n2} & \cdots & \sigma_{nn} \end{bmatrix} = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1n} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{n1} & \sigma_{n2} & \cdots & \sigma_n^2 \end{bmatrix}$$

▶ 공분산행렬에서 고유벡터와 고윳값을 찾는 것은 데이터의 주성분(principal component)을 찾는 것과 동일하다. 왜냐하면 고유 벡터는 행렬이 어떤 방향으로 힘을 가하는지를 표현하는데, 이것은 데이터가 어떤 방향으로의 분산이 가장 큰지를 구하는 것과 같기 때문이다. 고윳값(Eigenvalues)은 각 고유벡터에 해당하는 상관계수이다. 우리가 다루는 행렬이 공분산 행렬일 경우 고윳값은 각 축에 대한 공분산 값이 된다. 그리고 고윳값이 큰 순서대로 고유벡터를 정렬하면, 결과적으로 중요한 순서대로 주성분을 구하는 것이 된다.

정의. 독립사건, 종속사건

두 개의 사건 A, B 가

$$P(A \cap B) = P(A)P(B) \quad \text{또는} \quad P(B|A) = P(B) \quad \text{이} \text{고} \quad P(A|B) = P(A)$$

를 만족할 때 통계적 독립(Statistically Independent)이라 하고, A 와 B 를 독립사건이라 한다.

정리. **공분산, 상관계수**

(1) X, Y 가 서로 독립이면 $E(XY) = E(X) \cdot E(Y) = \mu_x \mu_y$ 이므로 공분산은 0이다.

(2) X, Y 가 서로 독립이면 $\text{Corr}(XY) = 0$ 이므로 상관계수는 0이다. ■

예제 17 두 확률변수 X, Y 의 결합밀도함수가 다음과 같이 주어져 있다고 하자.

$$f(x, y) = x + y, \quad 0 < x < 1, \quad 0 < y < 1$$

공분산 $\text{Cov}(X, Y)$ 과 상관계수 $\text{Corr}(X, Y)$ 를 구하여라.

풀이. 예제 15에서 얻은 X 와 Y 의 주변밀도함수

$$f_X(x) = x + \frac{1}{2}, \quad 0 < x < 1, \quad f_Y(y) = y + \frac{1}{2}, \quad 0 < y < 1$$

로부터 공분산과 상관계수를 다음의 Sage 코드를 활용하여 구할 수 있다.

[Sage code] <http://sagecell.sagemath.org/> <http://sage.skku.edu/>

```
var('x, y')
f(x, y) = x + y # 결합밀도함수
f_X(x) = integral(f(x, y), y, 0, 1) # X의 주변밀도함수
f_Y(y) = integral(f(x, y), x, 0, 1) # Y의 주변밀도함수
EX = integral(x*f_X(x), x, 0, 1) # E[X]
EY = integral(y*f_Y(y), y, 0, 1) # E[Y]
EXY = integral(integral(x*y*f(x, y), x, 0, 1), y, 0, 1) # E[XY]
Cov = EXY - EX*EY # Cov[X, Y]
print("Cov[X, Y] =", Cov)
VX = integral(x^2*f_X(x), x, 0, 1) - EX^2 # V[X]
VY = integral(y^2*f_Y(y), y, 0, 1) - EY^2 # V[Y]
Corr = Cov/(sqrt(VX)*sqrt(VY)) # Corr[X, Y]
```

```
print("Corr[X, Y] =", Corr)
```

```
Cov[X, Y] = -1/144    # Cov[X, Y]  
Corr[X, Y] = -1/11     # Corr[X, Y]
```



2019 Spring Conference of the Korean Society of Mathematical Education

변화하는 사회, 변화하는 수학교육

<http://matrix.skku.ac.kr/2019-Conference/>

4. 데이터 활용의 실제

- 아래는 웹으로부터 데이터를 불러와서 분석을 진행한 사례로 출처는 다음과 같다.

[출처] <http://matrix.skku.ac.kr/E-math/>
<http://matrix.skku.ac.kr/E-Math/R-Practice-all.txt>

다음은 txt 형식으로 작성된 파일(my_data.txt)을 웹으로부터 불러와서 데이터 프레임을 작성하는 예제이다.

[R code] <http://sagecell.sagemath.org/> <http://sage.skku.edu/> (언어를 R로 바꾸어서 실행)

```
# 텍스트 파일로 부터 데이터 프레임 작성
my_data1      =      read.table(file="http://matrix.skku.ac.kr/E-Math/my_data.txt",
header=T)
my_data1
```

	Walking	Aerobic	Flexibility	Sport	Bike
1	4	3	2	0	3
2	7	5	5	0	2
3	16	11	12	4	5
4	12	11	17	0	3
5	8	6	9	0	1
6	11	6	6	0	2
7	14	13	15	4	6
8	9	4	6	1	2
9	4	3	3	0	1
10	8	7	9	1	4

다음은 my_data의 Walking 변수의 히스토그램을 그린 예시이다.

[R code] <http://sagecell.sagemath.org/> <http://sage.skku.edu/> (언어를 R로 바꾸어서 실행)

```
# 텍스트 파일로 부터 데이터 프레임 작성
my_data = read.table(file="http://matrix.skku.ac.kr/E-Math/다중회귀_최종모형.txt",
```

```
header=T)
# Walking 히스토그램
hist(my_data$Walking, main="Distribution of Walk 분포", xlab="Wk")



---


$breaks
[1] 0 5 10 15 20 25 30 35 40

$counts
[1] 52 114 115 48 21 9 4 2

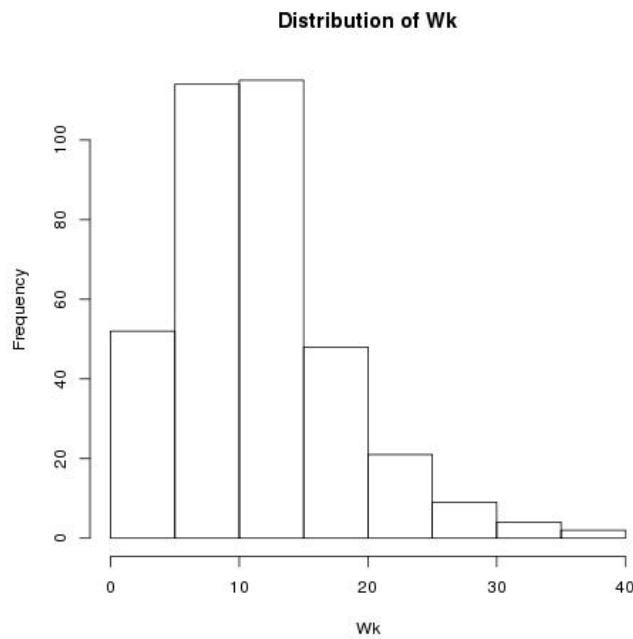
$density
[1] 0.028493151 0.062465753 0.063013699 0.026301370 0.011506849
0.004931507
[7] 0.002191781 0.001095890

$mids
[1] 2.5 7.5 12.5 17.5 22.5 27.5 32.5 37.5

$xname
[1] "my_data$Walking"

$equidist
[1] TRUE

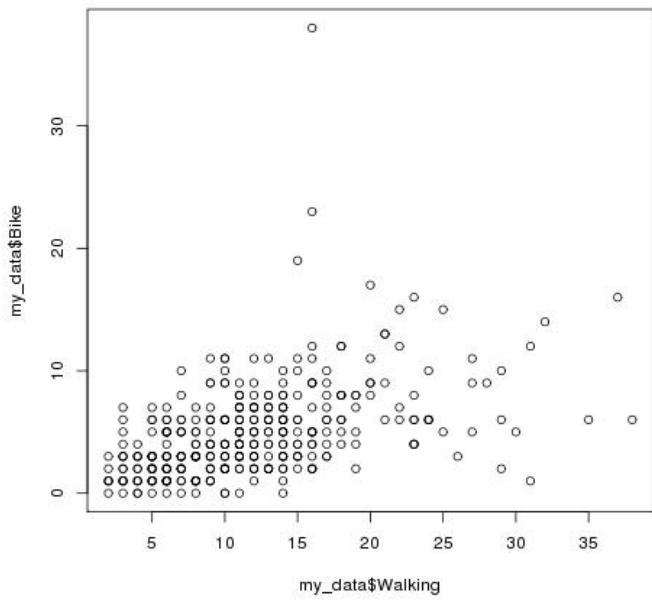
attr("class")
[1] "histogram"
```



다음은 my_data의 Walking 변수와 Bike 변수의 자료를 평면에 그린 예시이다.

[R code] <http://sagecell.sagemath.org/> <http://sage.skku.edu/> (언어를 R로 바꾸어서 실행)

```
# 텍스트 파일로 부터 데이터 프레임 작성
my_data = read.table(file="http://matrix.skku.ac.kr/E-Math/다중회귀_최종모형.txt",
header=T)
# Walking, Bike 데이터 그리기
plot(my_data$Walking, my_data$Bike)
```



다음은 자료의 각 변수에 관하여 표준편차, 평균 등을 계산한 예제이다. attach 명령을 사용하면 직접 변수명으로 접근할 수 있다.

[R code] <http://sagecell.sagemath.org/> <http://sage.skku.edu/> (언어를 R로 바꾸어서 실행)

R 데이터 프레임의 변수 이용 방법 text 파일 사용 2016. 3. 12.

```
my_data = read.table(file="http://matrix.skku.ac.kr/E-Math/다중회귀_최종모형.txt",
header=T)
print(mean(my_data$Walking))      # Walking 변수 자료의 평균
print(sd(my_data$Walking))        # Walking 변수 자료의 표준편차
print(attach(my_data))            # attach
summary(Walking)                 # Walking 변수 자료의 요약
```

```
[1] 11.90411
[1] 6.428312
<environment: 0x95e2d20>
attr("name")
[1] "my_data"
Min. 1st Qu. Median Mean 3rd Qu. Max.
2.0    7.0   11.0  11.9  15.0  38.0
```

■

우리는 Part 3에서 인공지능을 이해하기 위하여 필수적인 <기초 통계 및 확률>을 학습하였다. 통계적 방법은 인공 지능을 구현하는 구체적 접근 방식인 머신러닝 예측 모델링에서 매우 중요한 역할을 한다. 다음은 머신러닝 프로젝트를 수행할 때 따르는 통계적 방법의 적용 절차를 잘 보여준다.

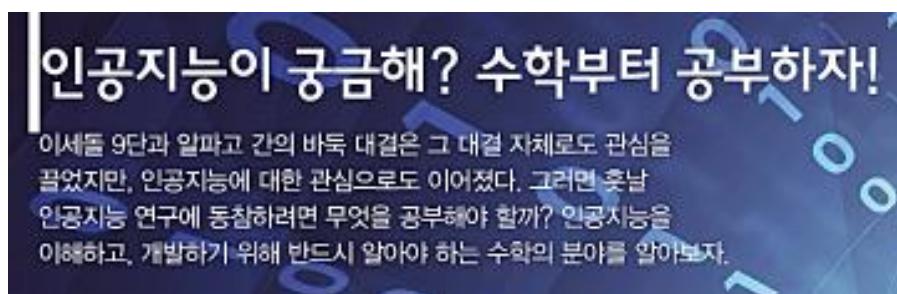
[출처] https://machinelearningmastery.com/statistics_for_machine_learning/

- ① 문제 구성 : 탐색적 데이터 분석 및 데이터 마이닝이 요구됨
- ② 데이터 이해 : 요약 통계 및 데이터 시각화가 필요함
- ③ 데이터 정리 : 이상치(outlier)를 탐지하고 결측값을 대치해야 함.
- ④ 데이터 선택 : 데이터 샘플링 및 특징선택(feature selection) 방법을 사용해야 함.
- ⑤ 데이터 준비 : 데이터를 변환하고, 스케일링, 인코딩 등을 사용해야 함.
- ⑥ 모델 평가 : 실험 설계 및 리샘플링 방법이 필요함.
- ⑦ 모델 구성 : 통계적 가설 검정 및 추정 통계를 사용해야 함.
- ⑧ 모델 선택 : 통계적 가설 검정 및 추정 통계를 사용해야 함.
- ⑨ 모델 제시 : 신뢰 구간과 같은 추정 통계를 사용해야 함.
- ⑩ 모형 예측 : 예측 구간과 같은 추정 통계를 사용해야 함.

■ 머신러닝에서 주로 사용되는 심화 통계 내용은 다음과 같다.

[출처] <https://machinelearningmastery.com/statistics-for-machine-learning-mini-course/>

통계 및 머신러닝, 통계 소개, 가우스 분포 및 기술 통계, 변수 간의 상관 관계, 통계적 가설 검정, 추정 통계, 비모수 통계 ■



■ 머신러닝에서 주로 사용되는 심화 통계 주제 및 응용에 관하여 학습을 원하는 대학원 학생들은 아래의 [도서(Statistical Learning)와 강의 동영상]을 참조하라.

[1] G. James, D. Witten, T. Hastie and R. Tibshirani, An Introduction to Statistical Learning: with Applications in R, Springer, 2017.

(교재) <http://faculty.marshall.usc.edu/gareth-james/ISL/ISLR%20Seventh%20Printing.pdf>

(동영상 강의) <https://www.r-bloggers.com/in-depth-introduction-to-machine-learning-in-15-hours-of-expert-videos/>

Chapter 1: Introduction (slides, playlist)

Chapter 2: Statistical Learning (slides, playlist)

Chapter 3: Linear Regression (slides, playlist)

Chapter 4: Classification (slides, playlist)

Chapter 5: Resampling Methods (slides, playlist)

Chapter 6: Linear Model Selection and Regularization (slides, playlist)

Chapter 7: Moving Beyond Linearity (slides, playlist)

Chapter 8: Tree-Based Methods (slides, playlist)

Chapter 9: Support Vector Machines (slides, playlist)

Chapter 10: Unsupervised Learning (slides, playlist)

Interviews (playlist)

- Interviews with statistics graduate students (7:44)



[부록 4] 기초통계 개념

정의. 표본공간(sample space), 사건(event), 표본점(sample point)

- (1) 같은 조건하에서 몇 번이고 반복할 수 있는 결과가 우연에 의해서 정해지는 실험 또는 관찰을 시행이라 하고 시행의 결과로 일어나는 것을 사건이라 한다.
- (2) 실험 또는 시행에 의하여 일어날 수 있는 모든 가능한 결과들의 집합을 표본공간 (sample space)이라 한다. 표본공간의 부분집합을 사건(event)이라 한다.
- (3) 표본공간을 구성하고 있는 개개의 원소를 표본점(sample point)이다
- (4) 표본공간의 한 원소만으로 이루어진 사건을 단순사건이다.

- n 개의 표본점 w_1, w_2, \dots, w_n 으로 구성되어 있는 표본공간 S 는 다음과 같이 표현된다.

$$S = \{w_1, w_2, \dots, w_n\}$$

- 균원사건(단순사건) : 표본공간의 한 원소만으로 이루어진 사건 $\{w_1\}, \{w_2\}, \dots, \{w_n\}$ 을 균원사건 또는 단순사건이다.

- (5) 반드시 일어나는 사건을 전사건이라 하고 S 로 나타낸다.
표본공간의 모든 원소를 포함하고 있는 사건을 전사건이다
- (6) 아무도 일어나지 않는 사건을 공사건이라 하고 \emptyset 로 나타낸다.
표본점을 하나도 포함하지 않는 사상을 공사건이다.

예제 1 동전을 세 번 던지는 실험에서 앞면을 H, 뒷면을 T 일 때 표본 공간 S 는

$$S = \{\text{HHH}, \text{HHT}, \text{HTH}, \text{THH}, \text{HTT}, \text{THT}, \text{TTH}, \text{TTT}\}$$

이다. 적어도 한 번 앞면이 나오는 사건

$A = \{\text{HHH}, \text{HHT}, \text{HTH}, \text{THH}, \text{HTT}, \text{THT}, \text{TTH}\}$ 는 S 의 부분 집합이다. 두 번 앞면이 나오는 사건 $B = \{\text{HHT}, \text{HTH}, \text{THH}\}$ 는 S 의 부분 집합이다. 한 번 앞면이 나오는 사건 $C = \{\text{HTT}, \text{THT}, \text{TTH}\}$ 는 S 의 부분 집합이다. 앞면이 한 번도 나오지 않는 사건 $D = \{\text{TTT}\}$ 는 S 의 부분 집합이다.

예제 2 한 개의 동전을 앞면이 나올 때까지 던질 때 표본 공간 S 는

$$S = \{H, TH, TTH, TTTH, \dots\}$$

이다. 네 번째 앞면이 나오는 사건 $A = \{\text{TTTH}\}$ 은 S 의 부분 집합이다.

- 표본공간 S 의 임의의 사건 A, B 에 대하여 다음과 같은 사건의 연산이 성립 한다.

① 합사건(union event)

임의의 두 사건 A, B 에 대하여 적어도 한쪽이 일어나는 사건을 합사건

$$A \cup B = \{x \mid x \in A \text{ or } x \in B\}$$

② 곱사건(intersection event)

임의의 두 사건 A, B 에 대하여 양쪽이 동시에 일어나는 사건을 곱사건

$$A \cap B = \{x \mid x \in A \text{ and } x \in B\}$$

③ 여사건(complement event)

임의의 사건 A 가 일어나지 않는 사건을 여사건

$$A^c = \{x \mid x \notin A \text{ and } x \in S\}$$

④ 배반사건(exclusive event, disjoint event)

임의의 두 사건 A 와 B 가 동시에 일어나지 않는 사건, 즉, $A \cap B = \emptyset$ 일 때 즉, 두 사건을 배반사건

예제 3 동전을 세 번 던지는 시행에서 앞면이 나타나면 H, 뒷면이면 T이다. 이 시행에서

- (1) 표본공간 S 를 나타내어라.
- (2) 첫 번째 던진 동전의 결과가 앞면이 되는 사건 A 를 나타내라.
- (3) 두 번째 던진 동전의 결과가 뒷면이 되는 사건 B 를 나타내라.
- (4) 사건 A 와 B 는 서로 배반인가?
- (5) $A \cup B, A \cap B$ 와 A^c 를 구하라.

풀이.

- (1) $S = \{\text{HHH}, \text{HHT}, \text{HTH}, \text{THH}, \text{HTT}, \text{THT}, \text{TTH}, \text{TTT}\}$
- (2) $A = \{\text{HHH}, \text{HHT}, \text{HTH}, \text{HTT}\}$
- (3) $B = \{\text{HTH}, \text{HTT}, \text{TTH}, \text{TTT}\}$
- (4) 사상 A 와 B 는 동일한 근원사건 (H, T)를 가지고 있으므로 서로 배반인 아니다.
- (5) $A \cup B = \{\text{HHH}, \text{HHT}, \text{HTH}, \text{THH}, \text{HTT}, \text{TTH}, \text{TTT}\}$, $A \cap B = \{\text{HTH}, \text{HTT}\}$,
 $A^c = \{\text{THH}, \text{THT}, \text{TTH}, \text{TTT}\}$

예제 4 1에서 16까지 숫자가 하나씩 적힌 16개의 공이 들어 있는 주머니에서 한 개의 공을 꺼내는 시행을 한다. 꺼낸 공에 적힌 숫자가 짝수인 사건을 A , 16의 약수인 사건을 B , 3의 배수인 사건을 C 라고 할 때 다음을 구하여라.

- (1) $A \cup B$
- (2) $A \cap B$
- (3) $B \cap C$
- (4) A^c
- (5) B^c
- (6) $A^c \cap B^c$

풀이. 표본공간 S 는

$$S = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16\}$$

이며, 세 사건 A , B , C 는

$$A = \{2, 4, 8, 10, 12, 14, 16\}, \quad B = \{1, 2, 4, 8, 16\}, \quad C = \{3, 6, 9, 12, 15\}$$

이므로

- (1) $A \cup B = \{1, 2, 4, 6, 8, 10, 12, 14, 16\}$
- (2) $A \cap B = \{2, 4, 8, 16\}$
- (3) $B \cap C = \emptyset$
- (4) $A^c = \{1, 3, 5, 7, 9, 11, 13, 15\}$
- (5) $B^c = \{3, 5, 6, 7, 9\}$
- (6) $A^c \cap B^c = \{3, 5, 7, 9, 11, 13, 15\}$

[Sage code] <http://sagecell.sagemath.org/> <http://sage.skku.edu/>

```
S = Set([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16])    # 표본공간
A = Set([2, 4, 8, 10, 12, 14, 16])    # 짝수 사건
B = Set([1, 2, 4, 8, 16])            # 16의 약수 사건
```

```

C = Set([3, 6, 9, 12, 15])          # 3의 배수
print(A.union(B))                  # 합집합 사건
print(A.intersection(B))           # 교집합 사건
print(B.intersection(C))
Acomp = S.difference(A)           # 여집합 사건
print(Acomp)
Bcomp = S.difference(B)
print(Bcomp)
Acomp.intersection(Bcomp)          # 교집합 사건

```

```

{1, 2, 4, 8, 10, 12, 14, 16}
{2, 4, 8, 16}
{}
{1, 3, 5, 6, 7, 9, 11, 13, 15}
{3, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15}
{3, 5, 6, 7, 9, 11, 13, 15}

```

예제 5 한 개의 주사위를 던질 때 4이하의 눈이 나올 확률을 구해보자.

풀이. 표본공간 S 는 $S = \{1, 2, 3, 4, 5, 6\}$ 이며, 4이하의 눈이 나오는 사건을 A 은

$$A = \{1, 2, 3, 4\} \text{ 이므로 } P(A) = \frac{n(A)}{n(S)} = \frac{4}{6} = \frac{2}{3} \text{이다.}$$

[Sage code] <http://sagecell.sagemath.org/> <http://sage.skku.edu/>

```

S = Set([1, 2, 3, 4, 5, 6])      # 표본공간
A = Set([1, 2, 3, 4])            # 4이하의 짝수 사건
P_A = A.cardinality()/S.cardinality() # A사건이 나타날 확률
P_A

```

2/3

예제 6 주사위 한 개를 던지는 시행을 n 번 반복할 때 등이 나오는 횟수 r_n 을 조사한 결과가 아래 표와 같았다. 주사위의 눈이 1이 나올 통계적 확률을 소수 첫째 자리까지 구하여보자.

n	100	200	300	600
r_n	10	30	48	102

풀이. 상대도수 $\frac{r_n}{n}$ 의 값을 차례로 구해보면 0.1, 0.15, 0.16, 0.17으로 n 의 값이 커질수록 $\frac{1}{6}$ 에 가까워짐을 알 수 있다. 따라서 등이 나올 통계적 확률은 $\frac{1}{6}$ 이다.

[R code] <http://sagecell.sagemath.org/> <http://mathlab.knu.ac.kr/r/> <http://sage.skku.edu/>

```
n <- 10000
dice1 = sample(1:6, n, replace = TRUE) # 주사위 던지기
r <- as.matrix(table(dice1)) # 표로 정리된 것을 행렬로 표현
print(r[1])                  # 1이 나온 갯수
print(r[1]/n)                # 1이 나온 확률
r
```

```
[,1]
1 1676
2 1663
3 1619
4 1714
5 1676
6 1652
[1] 1676
[1] 0.1676
```

예제 7 반지름의 길이가 각각 9, 3인 두 원 A , B 가 있다. 원 내부의 각각의 점을 잡을 가능성은 같은 정도로 기대된다고 하면 원 A 의 내부에서 임의의 한 점을 잡을 때 그 점이 원 B 의 내부의 점일 확률 p 는

$$p = \frac{\text{area of } B}{\text{area of } A} = \frac{\pi \cdot 3^3}{\pi \cdot 9^2} = \frac{1}{9}$$

이다.

예제 8 네 개의 동전을 동시에 던질 때 뒷면이 적어도 한 개 나올 확률을 구하여 보자.

풀이. 네 개의 동전을 동시에 던질 때 뒷면이 적어도 한 개 나오는 사건을 A 라 하면 A 의 여사건은 네 개 모두 앞면이 나오는 사건이므로

$$P(A^c) = \frac{1}{16}$$

$$P(A) = 1 - P(A^c) = 1 - \frac{1}{16} = \frac{15}{16}.$$

정리. 교사건(intersection), 합사건(union)의 확률, 덧셈정리

① 임의의 사건 A, B 가 $A \subset B$ 이면 $P(A) \leq P(B)$

② $P(A \cap \overline{B}) = P(A) - P(A \cap B) = P(A \cup B) - P(B)$

③ 임의의 사건 A, B 에 대하여

$$P(A \cup B) = P(A) + P(B) - P(A \cap B) \quad (\text{덧셈정리})$$

④ 임의의 사건 A, B 가 서로 배반사건일 때

$$P(A \cup B) = P(A) + P(B)$$

⑤ 임의의 사건 A, B 에 대하여 $P(A \cup B) \leq P(A) + P(B)$

⑥ $P(\overline{A} \cup \overline{B}) = P(\overline{A \cap B}) = 1 - P(A \cap B)$

예제 9 어느 학급의 전체 학생들을 대상으로 조사한 결과 야구를 좋아하는 학생은 50%, 축구를 좋아하는 학생은 40%, 야구와 축구를 좋아하는 학생은 20% 이었다. 야구 또는 축구를 좋아하는 학생은 전체의 몇 % 인지 구하여보자.

풀이. $P(A \cup B) = P(A) + P(B) - P(A \cap B) = 0.50 + 0.40 - 0.20 = 0.70$

야구 또는 축구를 좋아하는 학생은 전체의 70%이다.

예제 10 어느 체육 고등학교에서 태권도, 유도, 복싱을 전공하는 학생이 각각 70명, 60명, 20명이라고 한다. 학생들 가운데 임의로 한 명을 뽑을 때, 이

학생이 유도 또는 복성을 전공하는 학생일 확률을 구하여보자. (단 모든 학생은 반드시 하나의 분야를 전공한다고 하자.)

풀이]. $\frac{60}{150} + \frac{20}{150} = \frac{8}{15}$

정의. 조건부 확률(conditional probability)

어떤 사건 A 가 일어났다는 조건하에서 사건 B 가 일어날 확률을 사건 A 에 대한 사건 B 의 조건부 확률(conditional probability)이라 하고 $P(B|A)$ 로 표시하며 다음과 같이 정의한다.

$$P(B|A) = P_A(B) = \frac{P(A \cap B)}{P(A)} \quad (\text{단, } P(A) > 0)$$

정리. 조건부 확률

- ① 임의의 사건 B 에 대하여 $0 \leq P(B|A) \leq 1$
- ② $B_1 \cap B_2 = \emptyset$ 이면, $P(B_1 \cup B_2|A) = P(B_1|A) + P(B_2|A)$
- ③ $P(A|A) = 1$

예제 11 두 개의 주사위를 던지는 실험에서 두 주사위의 눈의 합을 X 라고 하자. X 가 홀수라고 할 때 X 가 8보다 작을 확률을 구하여보자.

풀이]. A 가 $X < 8$ 인 사건이고 B 는 X 가 홀수인 사건이라고 하면 $P(A|B)$ 가 우리가 구하려는 확률이다.

$A \cap B$ 는 X 가 3, 5, 7인 사건이고 표본공간은 36가지이므로

$$P(A|B) = \frac{2}{36} + \frac{4}{36} + \frac{6}{36} = \frac{1}{3}$$

$$P(B) = \frac{2}{36} + \frac{4}{36} + \frac{6}{36} + \frac{4}{36} + \frac{2}{36} = \frac{1}{2}$$

이다. 그러므로

$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{2}{3}.$$

정리. 곱셈정리

조건부 확률의 정의로부터 다음의 곱셈정리를 얻을 수 있다.

$$P(A \cap B) = P(A)P(B|A) = P(B)P(A|B)$$

또한, 일반적으로 사건 A_1, A_2, \dots, A_n 에 대하여 다음이 성립한다.

$$\begin{aligned} P(A_1 \cap A_2 \cap \dots \cap A_n) &= P(A_1)P(A_2|A_1)P(A_3|A_1 \cap A_2) \\ &\quad \dots P(A_n|A_1 \cap A_2 \cap \dots \cap A_{n-1}) \end{aligned}$$

예제 12 7개의 검은 공과 8개의 흰 공이 들어 있는 주머니로부터 2개의 공을 비복원추출한다고 할 때 2개의 공 모두가 흰 공일 확률을 구하여보자.

풀이. A_i 를 i ($i = 1, 2$) 번째 꺼낸 공이 흰 공일 사건이라고 하면

$$P(A_1) = \frac{8}{15}, \quad P(A_2|A_1) = \frac{7}{14} = \frac{1}{2}$$

이다. 그러므로 2개의 공이 모두 흰 공일 사건 $A_1 \cap A_2$ 에 대한 확률은

$$P(A_1 \cap A_2) = P(A_1)P(A_2|A_1) = \frac{8}{15} \cdot \frac{1}{2} = \frac{4}{15}.$$

정의. 독립사건, 종속사건

두 개의 사건 A, B 가

$$P(A \cap B) = P(A)P(B) \text{ 또는 } P(B|A) = P(B)P(B|A) = P(B)$$

를 만족할 때 통계적 독립이라 하고, A 와 B 를 독립사건이라 한다.

A 와 B 가 서로 독립이 아닐 때 A 와 B 는 종속사건이라 한다.

만약 사상 A, B 가 독립이면, 다음 두 사건들도 독립이다.

- ① A 와 B^c
- ② A^c 와 B
- ③ A^c 와 B^c

또한, 3개의 사상 A, B, C 에 대해서 다음 4개의 조건이 성립할 때, 사상 A, B, C 는 서로 독립이라고 한다.

$$P(A \cap B) = P(A)P(B), P(A \cap C) = P(A)P(C), P(B \cap C) = P(B)P(C)$$

$$P(A \cap B \cap C) = P(A)P(B)P(C)$$

정리. 독립사건일 때 교사건(intersection)의 확률

두 사건 A 와 B 가 서로 독립사건일 때 교사건의 확률은 다음과 같다.

$$P(A \cap B) = P(A) \cdot P(B)$$

- A 와 B 가 독립사건이면 A^c 와 B^c 가 독립사건이다.

예제 13 검정색의 주사위와 흰색의 주사위를 던졌다. 검정색 주사위의 눈이 4 이라는 조건에서 두 주사위의 눈의 합이 8 이상일 확률을 구하여라.

풀이. B : 검정색 주사위의 눈이 4 인 사건

W : 검정색 주사위와 흰색 주사위의 두 눈의 합이 8 이상인 사건

$$B = \{(4, 1), (4, 2), (4, 3), (4, 4), (4, 5), (4, 6)\}$$

$$W = \{(2, 6), (3, 5), (3, 6), (4, 4), (4, 5), (4, 6), (5, 3), (5, 4), (5, 5), (5, 6), (6, 2), (6, 3), (6, 4), (6, 5), (6, 6)\}$$

$$B \cap W = \{(4, 4), (4, 5), (4, 6)\}$$

$$P(B) = \frac{6}{36} = \frac{1}{6}, \quad P(W) = \frac{15}{36} = \frac{5}{12}, \quad P(B \cap W) = \frac{3}{36} = \frac{1}{12}$$

$$P(W|B) = \frac{P(B \cap W)}{P(B)} = \frac{\frac{1}{12}}{\frac{1}{6}} = \frac{1}{2}$$

예제 14 주머니 안에 같은 크기로 7개의 흰 구슬과 3개의 검은 구슬이 들어 있다. 여기서, 1개씩 차례로 2개를 비복원으로 추출할 때 처음에는 흰 구슬, 나중에 검은 구슬이 나올 확률을 구하여라.

풀이.

A : 흰 구슬이 나올 사건

B : 검은 구슬이 나올 사건

처음에 흰 구슬이 나올 확률은 $P(A) = \frac{7}{10}$

나머지 9개의 구슬 중에서 검은 구슬이 나올 확률은 $P(B|A) = \frac{3}{9} = \frac{1}{3}$

구하고자 하는 확률은 다음과 같다.

$$P(A \cap B) = P(A)P(B|A) = \frac{7}{10} \times \frac{1}{3} = \frac{7}{30}$$

정리. 독립시행의 정리

확률 P 을 가지는 독립사건 A 가 n 회 반복시행 중 x 회 나타나는 확률을 $P(x : n, p)$ 라고 하면

$$P(x : n, p) = {}_n C_x p^x q^{n-x} \quad (\text{단, } p+q=1, x=0,1,2,\dots,n)$$

로 주어진다.

정의. 분할(partition)

어떤 집합 A 가 공집합이 아닐 때 집합 A 의 부분집합 A_1, A_2, \dots, A_n 을 원소로 하는 집합족 $\{A_1, A_2, \dots, A_n\}$ 이 다음 내용을 만족할 때 $\{A_1, A_2, \dots, A_n\}$ 을 집합 A 의 분할(partition)이라 한다.

- ① $A_i \subset A \quad (i = 1, 2, \dots, n)$
- ② $A_i \cap A_j = \emptyset \quad (i \neq j, i, j = 1, 2, \dots, n)$
- ③ $A_1 \cup A_2 \cup \dots \cup A_n = A$

예제 15 동전 10개를 던질 때 표면이 나오는 개수 X 의 평균과 분산을 구하여라.

풀이. $n = 10, p = \frac{1}{2}, q = \frac{1}{2}$

$$E(X) = np = 10 \times \frac{1}{2} = 5$$

$$V(X) = npq = 10 \times \frac{1}{2} \times \frac{1}{2} = \frac{5}{2}$$

예제 16 주사위 4개를 던질 때 적어도 6이 두 개 나올 확률을 구하여라.

풀이. $p = P(6) = \frac{1}{6}, q = \frac{5}{6}, n = 4.$

적어도 6이 두 번 나오는 경우는 6이 두 번, 세 번, 또는 네 번 나오면 된다. 따라서

$$\begin{aligned} p &= f(2) + f(3) + f(4) = \binom{4}{2} \left(\frac{1}{6}\right)^2 \left(\frac{5}{6}\right)^2 + \binom{4}{3} \left(\frac{1}{6}\right)^3 \left(\frac{5}{6}\right) + \binom{4}{4} \left(\frac{1}{6}\right)^4 \\ &= \frac{1}{6^4} (6 \cdot 25 + 4 \cdot 5 + 1) = \frac{171}{1296} \end{aligned}$$

[Sage code] <http://sagecell.sagemath.org/> <http://sage.skku.edu/>

```
print(sum([binomial(4, t)*(1/6)^t*(5/6)^(4-t) for t in range(2, 5)]))
```

19/144

[R code] <http://sagecell.sagemath.org/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
sum(dbinom(2:4, 4, 1/6))
```

[1] 0.1319444

■ 초기하분포(Hypergeometric distribution) $H(N, r, n)$

(1) (비복원추출) 모집단 N 에서 비복원추출하여 n 개의 표본을 추출할 때, 각 성분의 속성이 A 을 갖는 것을 A 형, A 을 갖지 않는 것을 \bar{A} 형이라 하자. 모집단 가운데 r 개가 A 형이고 $N-r$ 개가 \bar{A} 형일 때, n 개의 표본을 추출하여 A 형의 개수를 X 라 하면 $X=x$ 일 확률은 다음과 같다.

$$P(X=x) = f(x) = \frac{\binom{r}{x} \cdot \binom{N-r}{n-x}}{\binom{N}{n}} \quad (\text{단, } x=0, 1, 2, \dots, n)$$

여기서 $x=0, 1, 2, \dots, n$ 에 대하여 $f(x) \geq 0$ 이므로 $f(x)$ 는 확률질량함수이다.

그리고 확률변수 X 는 초기하분포 $H(N, r, n)$ 를 따른다고 한다.

(2) 확률변수 X 가 초기하분포 $H(N, r, n)$ 를 따르면

$$X \text{의 평균은 } \mu = E(X) = n \cdot \frac{r}{N}$$

$$X \text{의 분산은 } V(X) = \frac{N-n}{N-1} n \frac{r}{N} \left(1 - \frac{r}{N}\right)$$

분산식에서 $\frac{N-n}{N-1}$ 을 유한모집단수정계수(finite population correction coefficient)

라 하며 N 이 커질수록 이 값은 1에 접근한다.

예제 17 카드 더미에는 빨간색 카드 6장과 검은색 카드 4장이 있다. 카드를 복원하지 않고 무작위로 5장을 뽑는다고 할 때, 빨간색 카드가 4장이 뽑힐 확률은 얼마인가?

풀이. x 를 빨간색 카드의 장수라 하면, $5-x$ 는 검은색 카드의 장수이다. 복원하지 않고 5장을 뽑을 때 빨간색 카드를 x 장 뽑을 확률은 다음과 같다.

$$P(X=x) = \frac{{}^6C_x \times {}^4C_{5-x}}{{}^{10}C_5}, \quad x=1, 2, 3, 4, 5$$

$$\therefore P(X=4) = \frac{{}^6C_4 \times {}^4C_1}{{}^{10}C_5}$$

[Sage code] <http://sagecell.sagemath.org/> <http://sage.skku.edu/>

```
print(binomial(6, 4)*binomial(4, 1)/binomial(10, 5))
```

5/21

[R code] <http://sagecell.sagemath.org/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
m = 6  
n = 4  
dhyper(4, m, n, 5)
```

[1] 0.2380952

☞ 각 분포의 의미는 다음과 같다. 예를 들어, 이항분포는 수강생 중 한 명을 뽑았을 때 남자일 확률을 보여주며, 포아송 분포는 주어진 시간 내에 수업에 들어오는 남학생 수를 보여준다. 초기하(Hypergeometric) 분포는 수강생 중 남자와 여자라는 두 그룹이 있는 경우 한명을 뽑았을 때 남자일 확률을 보여준다.

■ 각 분포는 샘플의 크기(N)가 커질수록 서로의 분포로 수렴하기도 하고, 어떤 분포는 다른 분포의 부분집합(특별한 경우)이 되기도 한다. 예를 들어, 초기하 분포는 N (수강생 전체 인원수)과 r (남자의 수)을 고정시키고, $N \rightarrow \infty$, $r \rightarrow \infty$ 으로 보낼 때 $\frac{r}{N} \rightarrow p$ 로 수렴한다고 하면, 직관적으로 <수강생 중 한 명을 뽑았을 때 남자일 확률을 보여주는> 이항분포에 가까워지게 된다. 그리고 이항분포에서는 n (수강생의 수)이 고정되는데, 만일 이항분포에서 $n \rightarrow \infty$, $p \rightarrow 0$ 로 보내고, 평균은 유지된다고 가정하면 ($np \rightarrow \mu$), 이는 포아송 분포에 가까워지게 된다. 이를 도식화하면 다음과 같다.

$$\begin{array}{ccc} \text{Hypergeometric 분포} & \rightarrow & \text{Binomial 분포} & \rightarrow & \text{Poisson 분포} \\ N \rightarrow \infty & & n \rightarrow \infty & & \mu = np \\ r \rightarrow \infty & & p \rightarrow 0 & & \\ \frac{r}{N} \rightarrow p & & np \rightarrow \mu & & \end{array}$$

■ Gamma 분포(Gamma distribution) $G(\alpha, \beta)$

(1) Gamma함수의 정의

$$\begin{aligned} \Gamma(\alpha) &= \int_0^\infty e^{-t} t^{\alpha-1} dt \quad (\alpha > 0) \\ &= [-e^{-t} t^{\alpha-1}]_0^\infty + (\alpha-1) \int_0^\infty e^{-t} t^{\alpha-2} dt = (\alpha-1) \int_0^\infty e^{-t} t^{\alpha-1} dt \end{aligned}$$

$$= (\alpha - 1)\Gamma(\alpha - 1)$$

반복하여

$$\Gamma(\alpha) = (\alpha - 1)\Gamma(\alpha - 1) = (\alpha - 1)(\alpha - 2)\Gamma(\alpha - 2) = \dots$$

α 가 양의 정수 n 이면

$$\Gamma(n) = (n-1)(n-2) \cdots 1 \Gamma(1) \text{ 과 } \Gamma(1) = \int_0^\infty e^{-t} dt = 1 \text{에서}$$

$$\Gamma(n) = (n-1)!$$

$$\Gamma(\alpha) = \int_0^\infty e^{-t} t^{\alpha-1} dt \quad (\alpha > 0)$$

Gamma함수 $\Gamma(\alpha)$ 에서 $t = \frac{x}{\beta}$ 라면 $dt = \frac{1}{\beta} dx$

$$\Gamma(\alpha) = \int_0^\infty e^{-\frac{x}{\beta}} \left(\frac{x}{\beta}\right)^{\alpha-1} \frac{1}{\beta} dx \text{에서 } \int_0^\infty \frac{1}{\Gamma(\alpha)\beta^\alpha} e^{-\frac{x}{\beta}} x^{\alpha-1} dx = 1$$

$\Gamma(\alpha) > 0$ 에서 Gamma분포의 확률밀도함수를 정의한다.

$$\Gamma\left(\frac{1}{2}\right) = \sqrt{\pi}$$

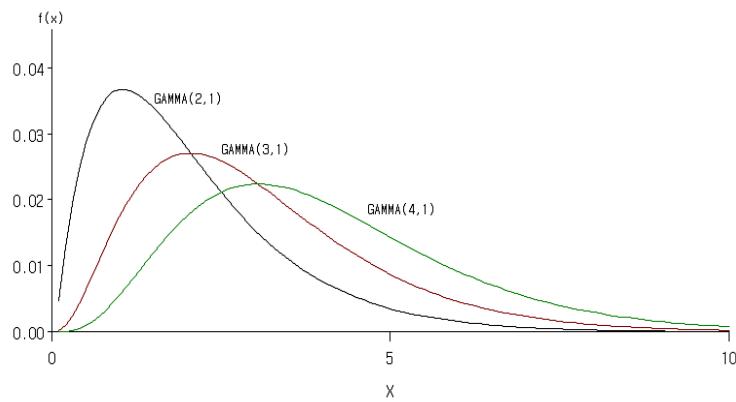
(2) Gamma분포의 확률밀도함수

Gamma함수에서 확률변수 X 의 확률밀도함수가 $\alpha, \beta > 0$ 에 대하여 다음과 같을 때, X 는 Gamma분포를 따른다고 하며, $G(\alpha, \beta)$ 로 표시한다.

$$f(x) = \begin{cases} \frac{1}{\Gamma(\alpha)\beta^\alpha} x^{\alpha-1} e^{-\frac{x}{\beta}} & (\alpha, \beta > 0) \\ 0 & x \leq 0 \end{cases} \quad 0 < x < \infty$$

일 때 $0 < x < \infty$ 에 대하여 $f(x) \geq 0$ 에서 $f(x)$ 는 확률밀도함수이다.

(3) Gamma분포는 신뢰성 이론, 기대 시간, 수명 시간 등의 분포에서 이용된다. α, β 의 특별한 값에 대하여 감마분포의 확률밀도함수를 그래프로 표현하면 다음과 같다.



(4) 확률변수 X 가 Gamma분포 $G(\alpha, \beta)$ 를 따르면

$$X \text{의 평균은 } \mu = E(X) = \alpha\beta$$

$$X \text{의 분산은 } \sigma^2 = V(X) = \alpha\beta^2$$

■ Beta분포 $Be(\alpha, \beta)$

(1) Beta함수

$$B(\alpha, \beta) = \int_0^1 x^{\alpha-1} (1-x)^{\beta-1} dx$$

(2) Beta함수와 Gamma함수의 관계

$$B(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$$

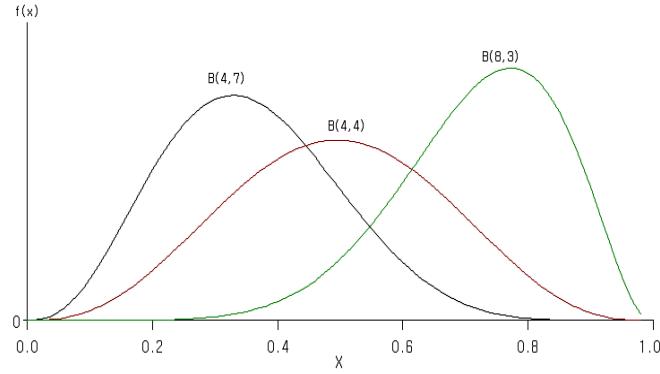
(3) Beta분포 $Be(\alpha, \beta)$ 의 확률밀도함수

확률변수 X 의 확률밀도함수가 $\alpha, \beta > 0$ 에 대하여 다음과 같을 때, X 는 Beta분포를 따른다.

$$f(x) = \begin{cases} \frac{1}{B(\alpha, \beta)} x^{\alpha-1} (1-x)^{\beta-1} (\alpha, \beta > 0) & 0 < x < 1 \\ 0 & x \leq 0, x \geq 1 \end{cases}$$

$$f(x) = \begin{cases} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1} (1-x)^{\beta-1} & (\alpha, \beta > 0) \\ 0 & x \leq 0, x \geq 1 \end{cases}$$

(4) Beta분포는 제품의 불량률이나 총 작업시간 중 휴업한 시간의 비율 등과 같은 비율에 대한 모형으로 사용되는 분포이다. α, β 의 특별한 값에 대하여 베타분포의 확률밀도함수를 그래프로 표현하면 다음과 같다.



(5) 확률변수 X 가 베타분포 $B(\alpha, \beta)$ 를 따를 때, X 의 평균과 분산은 다음과 같다.

$$E(X) = \frac{\alpha}{\alpha + \beta}, \quad Var(X) = \frac{\alpha\beta}{(\alpha + \beta + 1)(\alpha + \beta)^2}$$

■ χ^2 분포

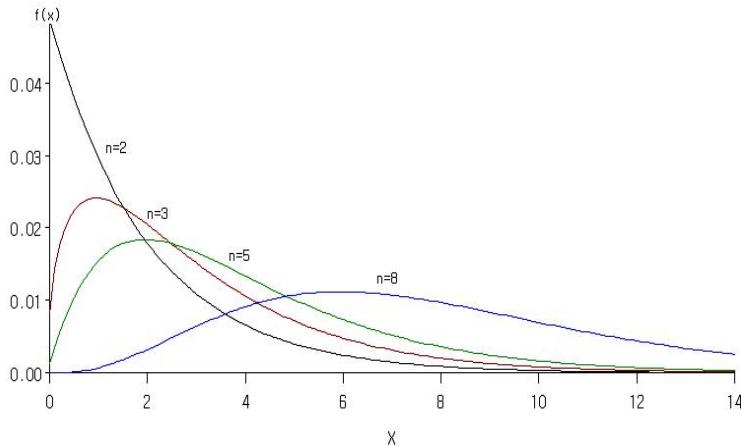
(1) χ^2 분포의 확률밀도함수

Gamma분포에서 $\alpha = \frac{n}{2}, \beta = 2$ 일 때 χ^2 분포의 확률밀도함수이다. 따라서 확률변수 X 의 확률밀도함수가

$$f(x) = \begin{cases} \frac{1}{\Gamma\left(\frac{n}{2}\right)2^{\frac{n}{2}}} x^{\frac{n}{2}-1} e^{-\frac{x}{2}} & 0 < x < \infty \\ 0 & x \leq 0 \end{cases}$$

일 때, X 는 자유도(degree of freedom)가 n 인 χ^2 분포(chi-square distribution)라 한다.

(2) χ^2 분포에서 n 의 크기에 따라 분포의 모양이 변하며, 자유도가 커지면 대칭에 가까워지며 여러 집단들 사이의 독립성 검정과 적합성 검정에 이용된다. χ^2 분포의 확률밀도함수를 그래프로 표현하면 다음과 같다.



(3) 확률변수 X 가 자유도 n 인 χ^2 분포일 때,

$$X \text{의 평균} : \mu = E(X) = n$$

$$X \text{의 분산} : \sigma^2 = V(X) = 2n$$

정리.

χ^2 분포를 따르는 통계량

(1) 확률변수 X 가 $N(\mu, \sigma^2)$ 인 정규분포일 때, $V = W^2 = \frac{(x - \mu)^2}{\sigma^2}$ 은 자유도 1인

χ^2 분포를 따른다.

(2) 확률변수 X 가 $N(\mu, \sigma^2)$ 인 정규모집단에서 서로 독립인 n 개의 확률표본

X_1, X_2, \dots, X_n 을 추출할 때 $Y = \sum_{i=1}^n \left(\frac{X_i - \mu}{\sigma} \right)^2$ 은 자유도 n 인 χ^2 분포를 따른다.

(3) 확률변수 X 가 $N(0,1)$ 인 정규모집단에서 크기가 n 개인 독립인 표본

X_1, X_2, \dots, X_n 에 대한 통계량은 $Y = \sum_{i=1}^n X_i^2$ 은 자유도 n 인 χ^2 분포를 따른다.

(4) 확률변수 X 가 $N(\mu, 1)$ 인 정규모집단에서 크기가 n 개인 확률표본

X_1, X_2, \dots, X_n 의 표본평균 \bar{x} 에 대한 통계량은 $Y = \sum_{i=1}^n (X_i - \bar{x})^2$ 은 자유도 $n-1$ 인 χ^2 분포를 따른다.

(5) 확률변수 X 가 $N(\mu, \sigma^2)$ 인 정규모집단에서 서로 독립인 n 개의 확률표본

X_1, X_2, \dots, X_n 을 추출할 때 $Y = \sum_{i=1}^n \left(\frac{X_i - \bar{x}}{\sigma} \right)^2$ 은 자유도 $n-1$ 인 χ^2 분포를 따른다.

(6) 확률변수 X 가 $N(\mu, \sigma^2)$ 인 정규모집단에서 서로 독립인 n 개의 확률표본 X_1, X_2, \dots, X_n 의 자유도가 n_1, n_2, \dots, n_n 인 χ^2 분포를 하는 확률변수라면 확률변수 $Y = X_1 + X_2 + \dots + X_n$ 의 자유도는 $n = n_1 + n_2 + \dots + n_n$ 인 χ^2 분포를 따른다.

■ *t* 분포

(1) *t* 분포의 확률밀도함수

Z 를 표준정규분포의 확률변수라 하고 V 를 자유도 n 인 χ^2 분포를 하는 확률변수라 하고 Z 와 V 가 독립아라면 확률변수 $X = \frac{Z}{\sqrt{\frac{V}{n}}}$ 의 분포는

$$f(x) = \frac{\Gamma\left(\frac{n+1}{2}\right)}{\sqrt{n\pi} \Gamma\left(\frac{n}{2}\right)} \left(1 + \frac{x^2}{n}\right)^{-\frac{n+1}{2}} \quad -\infty < x < \infty$$

자유도가 n 인 *t* 분포라 하고 $T \sim t(n)$ 로 나타낸다.

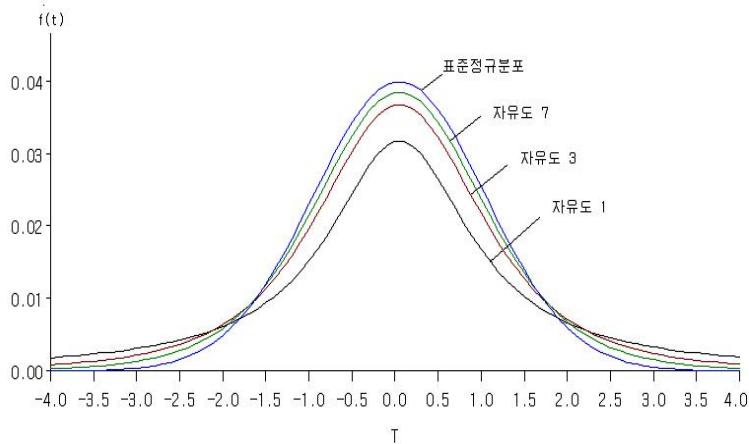
(2) 표본의 크기 n 이 작을 때, 즉 $n < 30$ 일 경우에 주로 *t* 분포를 이용하며, 모평균, 모평균의 차 또는 회귀계수의 추정이나 검정에 사용한다.

(3) 확률변수 X 가 자유도 n 인 *t* 분포일 때

X 의 평균 : $\mu = E(X) = 0$

X 의 분산 : $\sigma^2 = V(X) = \frac{n}{n-2}$

(4) t 분포의 확률밀도함수를 그래프로 표현하면 다음과 같다.



정리. t 분포를 따르는 통계량

(1) 확률변수 Z 가 $N(0, 1)$ 인 정규분포에서 V 가 자유도 n 인 χ^2 분포일 때 Z 와 V 가 독립이면 확률변수 $X = \frac{Z}{\sqrt{\frac{V}{n}}}$ 은 자유도 n 인 t 분포를 따른다.

(2) 확률변수 X 가 $N(\mu, \sigma^2)$ 인 정규모집단에서 크기가 n 개인 임의표본의 표본평균과 표본분산을 \bar{x}, s^2 에 대한 통계량은 $X = \frac{\bar{x} - \mu}{\frac{s}{\sqrt{n}}} \left(s = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2 f_i \right)$ 은 자유도 $n-1$ 인 t 분포를 따른다.

■ F 분포

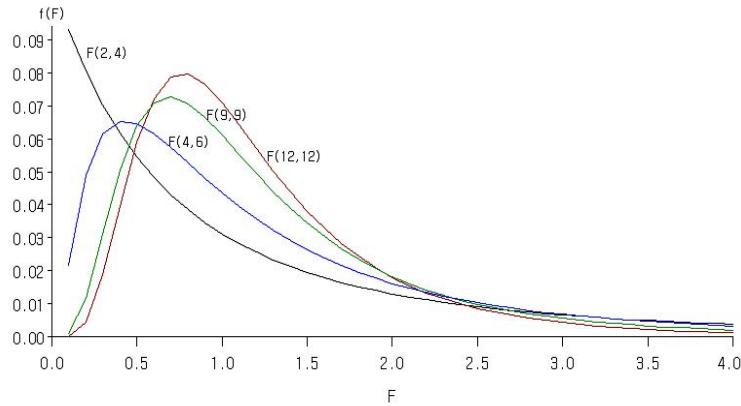
(1) U, V 를 각각 자유도가 n_1 과 n_2 인 χ^2 분포를 하는 독립인 확률변수이라면 확

률변수 $F = \frac{\frac{U}{n_1}}{\frac{V}{n_2}}$ 의 분포는

$$f(x) = \frac{\Gamma\left(\frac{n_1+n_2}{2}\right)}{\Gamma\left(\frac{n_1}{2}\right)\Gamma\left(\frac{n_2}{2}\right)} \left(\frac{n_1}{n_2}\right)^{\frac{n_1}{2}} x^{\frac{n_1}{2}-1} \left(1 + \frac{n_1}{n_2}x\right)^{-\frac{n_1+n_2}{2}} \quad x > 0$$

자유도가 n_1 과 n_2 인 F 분포를 따른다.

- (2) F 분포는 두 모분산의 비를 추론하거나 분산분석, 실험계획법 등에서 사용된다.
- (3) F 분포의 확률밀도함수를 그래프로 그리면 다음과 같다.



- (4) 확률변수 X 가 자유도 n_1 과 n_2 인 F 분포일 때

$$X \text{의 평균} : \mu = E(X) = \frac{n_2}{n_2 - 2} \quad n_2 > 2$$

$$X \text{의 분산} : \sigma^2 = V(X) = \frac{2n_2^2(n_1+n_2-2)}{n_1(n_2-2)^2(n_2-4)} \quad n_2 > 4$$

정리. F 분포를 따르는 통계량

- (1) X_1 와 X_2 의 자유도가 n_1 과 n_2 인 χ^2 분포를 하는 확률변수가 서로 독립인 통

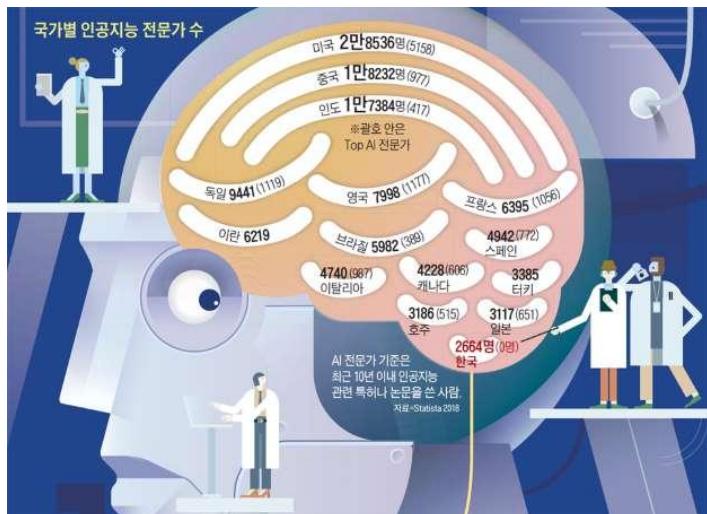
계량 $F = \frac{X_1^2/n_1}{X_2^2/n_2}$ 은 자유도가 (n_1, n_2) 인 F 분포를 따른다.

(2) 자유도가 (n_1, n_2) 인 F 분포의 α 점이 $F_{(n_1, n_2, \alpha)}$ 일 때 $F_{(n_2, n_1, 1-\alpha)} = \frac{1}{F_{(n_1, n_2, \alpha)}}$ 이다.

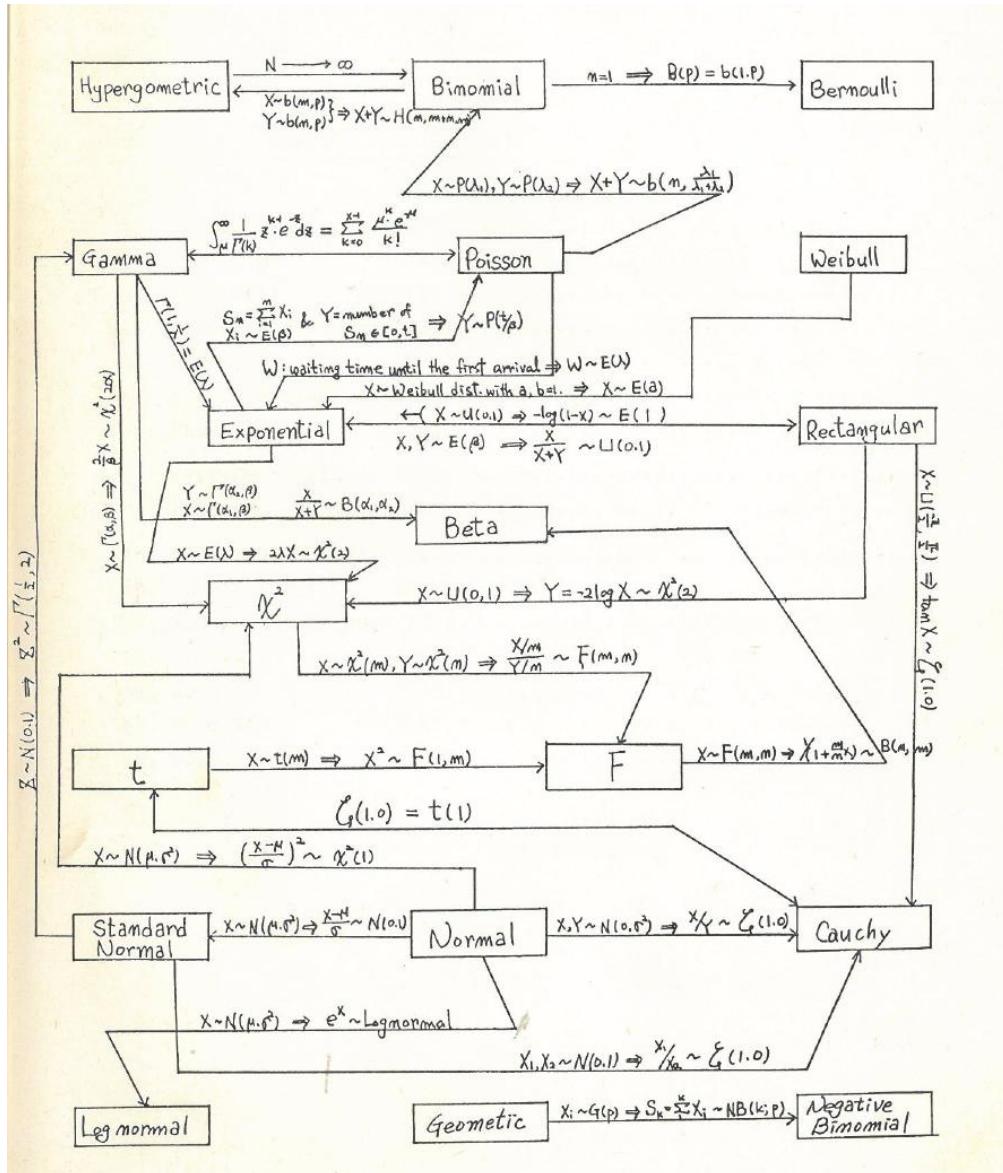
(3) 두 정규모집단 $N(\mu_1, \sigma_1^2)$ 와 $N(\mu_2, \sigma_2^2)$ 에서 추출한 표본의 크기가 n_1, n_2 인 독립확률 표본의 평균의 분산은 $s_1^2 = \frac{1}{n_1-1} \sum_{i=1}^{n_1} (x_i - \bar{x})^2$, $s_2^2 = \frac{1}{n_2-1} \sum_{i=1}^{n_2} (x_i - \bar{x})^2$ 라

면 통계량 $X = \frac{\frac{s_1^2}{\sigma_1^2}}{\frac{s_2^2}{\sigma_2^2}}$ 은 자유도가 (n_1-1, n_2-1) 인 F 분포를 따른다.

(4) X 가 자유도 n 인 t 분포일 때 X^2 은 자유도 $(1, n)$ 인 F 분포를 따른다.



[분포 간의 관계도 by 이우영]



■ [End of Part 3]

[이산수학 내용 설명과 코드가 필요한 학생은 아래 강의록과 실습실을 참고하면 된다.]

○ Discrete Mathematics (이산수학) Lectures:

Ch. 1, Sets and Logic

<http://matrix.skku.ac.kr/2018-DM/DM-Ch-1-Lab.html>

Ch 0 Introduction <https://youtu.be/9ahFnOFTWNQ> Ch 1 <https://youtu.be/J75uuYR-NTs>

Solutions <http://matrix.skku.ac.kr/2018-DM-Sol/Ch1/>

Ch. 2, Proofs

<http://matrix.skku.ac.kr/2018-DM/DM-Ch-2-Lab.html>

Ch 2 Proofs (1) <https://youtu.be/3LelrqADYUo> Ch-2-Proofs (2) <https://youtu.be/XqaleyIod10>

Solutions <http://matrix.skku.ac.kr/2018-DM-Sol/Ch1/>

Ch. 3, Functions, Sequences, and Relations,

<http://matrix.skku.ac.kr/2018-DM/DM-Ch-3-Lab.html>

Ch 3-1 Functions https://youtu.be/Y_hAaUxEpq0

Ch 3-2, 3-3, String, Relation <https://youtu.be/c73e13Otzng>

Ch 3-4, 3-5, Equivalence Relations <https://youtu.be/hwXAleCV4lA>

Solutions <http://matrix.skku.ac.kr/2018-DM-Sol/Ch3/>

Ch 4, Algorithms, Lecture Note

<http://matrix.skku.ac.kr/2018-DM/DM-Ch-4-Lab.html> <https://youtu.be/Dtv-9ykjFFA>

Solutions <http://matrix.skku.ac.kr/2018-DM-Sol/Ch4/>

Ch 5, Number Theory

<http://matrix.skku.ac.kr/2018-DM/DM-Ch-5-Lab.html>

Sec 5 1, 정수론 1 <https://youtu.be/yJL6lP4k0Bg> Sec 5.2, 5.3 정수론 2 https://youtu.be/-3BeY2_CBDk

Solutions <http://matrix.skku.ac.kr/2018-DM-Sol/Ch5/>

Ch 6, Counting Methods and the Pigeonhole Principle

<http://matrix.skku.ac.kr/2018-DM/DM-Ch-6-Lab.html>

Ch 6 일반화된 순열조합과 비둘기집 <https://youtu.be/I6XW6DKLoCU>

Solutions <http://matrix.skku.ac.kr/2018-DM-Sol/Ch6/>

Ch 7, Recurrence Relations

<http://matrix.skku.ac.kr/2018-DM/DM-Ch-7-Lab.html>

Ch 7 https://youtu.be/1n0dC_ICo4U Solutions <http://matrix.skku.ac.kr/2018-DM-Sol/Ch7/>

Ch 8, Graph Theory

<http://matrix.skku.ac.kr/2018-DM/DM-Ch-8-Lab.html>

*Graph Terminology and Lab <http://matrix.skku.ac.kr/2014-Album/Graph-Project.html>

8.1 Graph Theory <https://youtu.be/TSFIJBU2dX8> 8.2 Path and Cycle <https://youtu.be/iqUTT5C1TOs>

8.3 Hamiltonian cycle <https://youtu.be/at7Hx5wxnYk>

8.4, 8.5, 8.6, 8.7 <https://youtu.be/MHDJ3rALtEU>

이산수학 Ch 8 Graph 이론 part2 다익스트라 알고리즘 (한국어 강의) <https://youtu.be/Pe13RXWiuQE>

Solutions <http://matrix.skku.ac.kr/2018-DM-Sol/Ch8/>

Ch 9, Trees

<http://matrix.skku.ac.kr/2018-DM/DM-Ch-9-Lab.html>

Ch 9, Part 1 <https://youtu.be/v6wQeWmMBq8> Ch 9, Part 2 <https://youtu.be/g11cD6-0prs>

Final 학생 PBL 발표:

1. <https://youtu.be/i9FPdZdrFJw> 2. <https://youtu.be/nvwYSFwCoFo> 3. <https://youtu.be/DzMo4cfDLFY>

Midterm: <http://matrix.skku.ac.kr/2018-album/2018-F-DM-Midterm-Exam-Sol-F2-marked.pdf>

Final Exam: <http://matrix.skku.ac.kr/2019-album/2019-S-DM-Final-Sol-Final-5.pdf>

[공학수학 설명과 코드가 필요한 학생은 아래 강의록과 실습실을 참고하면 된다.]

○ Engineering Math with Sage (공학수학)

<http://www.hanbit.co.kr/EM/sage/>

<http://matrix.skku.ac.kr/2019-EM/EM-1-Labs.htm>

<http://matrix.skku.ac.kr/2019-EM/EM-2-Labs.htm>

[수치적 선형대수학과 수학적모델링 지식을 더 알고 싶은 학생은 저자의 아래 강의록과 동영상 강의를 참고하면 된다.]

Numerical Linear Algebra (수치적 선형대수)

Numerical Linear Algebra

Made by SML

<http://matrix.skku.ac.kr/nla/>

Math Modeling (수학적 모델링)

[강의 동영상] <http://matrix.skku.ac.kr/SOCW-Math-Modelling.htm>

Growth Models (exponential and logistic growth model) <http://youtu.be/9a7HumXe9To>

Component Analysis: population, Extended Growth Model <http://youtu.be/f7gjM-AoVhc>

Mathematical Modeling with Computational tools <http://youtu.be/ikXn20pp9Yw>

Physical Models: Projectile Motion(Moon's Orbit) <http://youtu.be/AFuJnIVZIZI>

Projectile Motion with Air Resistance <http://youtu.be/5UMW-tq7Tbk>

Predator–Prey model: Lotka–Volterra model <http://youtu.be/kQtKSOIGSgo>

Equilibrium: Supply–Demand Model, Harvest Model <http://youtu.be/PfBgfSCAhM>

Epidemic Models (SIR, SIS, SEIR) <http://youtu.be/s1jn0dkZd6I>

[실습실]

<http://matrix.skku.ac.kr/SOCW-Math-Modelling.htm>

<http://matrix.skku.ac.kr/MathModeling/Bank–Interest.html>

<http://matrix.skku.ac.kr/MathModeling/Power–method.html>

<http://matrix.skku.ac.kr/MathModeling/Markov–Chain.html>

<http://matrix.skku.ac.kr/MathModeling/Fibonacci–Numbers.html>

<http://matrix.skku.ac.kr/MathModeling/Growth–Model.html>

<http://matrix.skku.ac.kr/MathModeling/Exponential–Growth–Model.html>

<http://matrix.skku.ac.kr/MathModeling/Supply–Demand–Model.html>

<http://matrix.skku.ac.kr/MathModeling/Accelerator–multiplier–Model.html>

<http://matrix.skku.ac.kr/MathModeling/SIR–Model.html>

인공지능 알고리즘에 응용하는 수학, 선형회귀, 자연어 처리, 이미지 인식

24강 동영상, 빅데이터와 인공지능 개론 https://youtu.be/MpWv-U_4fl0 (11:22)

- Part IV에서는 먼저 인공지능(Artificial Intelligence, AI), 기계학습(Machine Learning, 머신러닝), 딥러닝(Deep Learning) 등의 개념과 인공지능의 역사에 대하여 간단히 알아보고, 앞서 배운 수학적 지식이 인공지능에서 어떻게 사용되는지 살펴본다. 구체적으로는 데이터 분석에서 자주 사용되는 주성분 분석(Principal Component Analysis, PCA), 신경망(Neural Network), 그리고 데이터 마이닝 학생프로젝트 사례를 소개한다.
- 인공지능, 기계학습, 딥러닝 등의 용어는 최근 뉴스와 광고 등에서 많이 사용되고 있음에도 그 명확한 구분은 쉽지 않다. Carnegie Mellon대 전산학부 Andrew Moore는 “인공지능은 최근까지 우리가 필요하다고 생각한 방식으로 컴퓨터가 행동하도록 만드는 과학 및 공학”이라고 정의한 바 있다.
- **기계 학습**은 컴퓨터 과학자이자 기계 학습의 선구자인 Tom M. Mitchell이 정의한 바와 같이 “컴퓨터 프로그램이 경험을 통해 자동으로 개선 될 수 있도록 하는 컴퓨터 알고리즘에 대한 연구”이다. 즉 기계학습은 인공지능과 동의어라기보다는 인공지능을 달성하기 위하여 주어진 데이터를 조사하고 비교하여 공통 패턴을 찾고, 뉘앙스를 탐색하여 특정한 과제를 수행하도록 돋는 한 방법론이라고 할 수 있다.
- 기계학습은 크게 **지도학습(Supervised Learning)**, **비지도학습(Unsupervised Learning)**, **강화학습(Reinforcement Learning)**의 세 가지 유형으로 분류할 수 있다. 지도학습은 데이터에 대한 레이블(Label)이 주어진 상태에서 컴퓨터를 학습시키는 방법으로 앞으로 2절에서 살펴볼 “MNIST 데이터셋을 활용한 손글씨 숫자 인식”이 대표적인 예라고 할 수 있다. 그리고 비지도 학습은 데이터에 대한 레이블이 주어지

지 않은 상태에서 컴퓨터를 학습시키는 방법으로 데이터의 숨겨진 특징이나 구조를 발견하는데 사용된다. 1절에서 살펴볼 주성분 분석(PCA)이 데이터의 차원을 줄이는 비지도학습의 대표적인 예이다. 마지막으로 강화학습은 환경과의 상호 작용에서 수집한 관측치를 사용하여 보상을 극대화하거나 위험을 최소화하는 조치를 취하는 것을 목표로 하는 것으로, 어떤 환경 안에서 정의된 에이전트(Agent)가 현재의 상태를 인식하여, 선택 가능한 행동들 중 보상(Reward)을 최대화하는 행동 혹은 행동 순서를 선택하는 방법이다.

- 신경망(Neural Network)은 기계학습의 모델 중의 하나로, 신경계의 기본 단위인 뉴런(신경세포, Neuron)을 모델화 한 것이다. 인공 뉴런(Artificial Neural Network)은 1943년에 Warren McCulloch와 Walter Pitts가 처음 소개하였고, 1950년대 초반에는 문제 해결과 기호법 등의 주제를 주로 탐구했다. 1950년대 후반에 이르러 Frank Rosenblatt과 여러 연구자들이 **퍼셉트론(Perceptron)이라 불리는 신경망**을 고안하였다. 퍼셉트론의 경우, 문제를 해결하는 가중치(weight)가 있는 경우에 학습 규칙이 정확한 네트워크의 가중치로 수렴한다는 것을 입증했지만, 많은 한계를 가지고 있어서, 오랫동안 신경망 연구가 중단되었다. 1960년대로 접어들면서 이러한 연구에 관심을 보인 미 국방부는 인간의 기본적인 추론 방식을 흉내 낼 수 있도록 컴퓨터를 훈련하기 시작하였고, 1970년 국방 고등연구기획국(DARPA)이 수행한 도로 지도화 프로젝트가 그 사례라고 할 수 있다. 1980년대에 이르러 다중 신경망(Multi-layer Networks)을 학습하는 오차 역전파법(Back Propagation Algorithm)이 개발되면서 신경망 연구가 다시 활력을 얻게 되었다. 또한 국방 고등 연구 기획국은 Siri, Alexa나 Cortana와 같은 인공지능이 개발되기 한참 전인 2003년에 지능형 개인 비서를 개발하기도 하였다.
- **딥러닝(Deep Learning)**은 음성을 인식하고, 이미지나 패턴을 확인하고, 다음 상황을 예측하는 일과 같이 인간이 하는 작업을 수행하도록 컴퓨터를 교육하는 일종의 기계 학습(머신 러닝)이다. 사전에 정의 된 수식을 통해 실행되도록 데이터를 구성하는 대신, 1. 주어진 데이터에 초기 파라미터(parameter)를 설정해주고, 2. 여러 층의 처리 계층을 사용하여 패턴을 인식하면서, 컴퓨터가 스스로 학습하도록 훈련(train)하는 것이다. 이러한 노력들은 오늘날 우리가 일상에서 접하고 있는 자동화와 형식 추론 기술의 기틀을 다지는 역할을 하였으며, 이러한 기술에는 사람의 능력을 보완하고 확장할 수 있는 의사 결정 지원 시스템과 스마트 검색 시스템이

포함된다. Amazon과 Netflix가 머신러닝을 이용한 추천 시스템의 개념을 대중화한 것과 같이, 앞으로 산업 전반적인 분야에서 인공지능(AI)이 활용될 것으로 사료된다.

[참고] 김성필, 딥러닝 첫걸음, 한빛미디어, 2016.

https://www.sas.com/en_gb/insights/analytics/deep-learning.html

https://www.sas.com/ko_kr/insights/analytics/what-is-artificial-intelligence.html

<https://www.forbes.com/sites/bernardmarr/2016/12/06/what-is-the-difference-between-artificial-intelligence-and-machine-learning/#77a428c52742>

[우리는 4장에서 주성분분석, 인공신경망, 그리고 실제 프로젝트의 예를 중심으로 소개한다.]

1. 주성분분석 (Principal Component Analysis, PCA)
2. 신경망 (Neural Network)
3. 학생 프로젝트 사례

읽을거리 1: 선형대수학과 구글(Google) 검색엔진

읽을거리 2: 인공지능, 인간을 뛰어넘은 비밀

읽을거리 3: 인공지능과 AI를 활용한 안면 인식기술

1. 주성분분석

25강 동영상, 빅데이터와 인공지능 PCA https://youtu.be/ukIttphmM_4 (34:59)

1.1 주성분 분석(Principal Component Analysis, PCA)

- 아래 1974년 Motor Trend magazine에 실린 1973년~74년도 자동차 32대의 특성을 기술하고 정량화하는 11가지 변수에 관한 데이터를 보자. 자료의 출처는 다음과 같다.

[출처] Robert Reris and J. Paul Brooks, *Principal Component Analysis and Optimization: A Tutorial*, 14th INFORMS Computing Society Conference, Richmond, Virginia, January 11-13, 2015, pp. 212-225 <http://www.people.vcu.edu/~jpbrooks/pcatutorial/>

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1
...											

따라서 자동차 한 대에 관한 데이터는 11차원의 벡터로 표현된다. 이처럼 고차원의 데이터는 계산과 시각화가 어려워 분석하기가 쉽지 않다. 따라서 원 데이터의 분포를 가능한 유지하면서 데이터의 차원을 줄이는 것이 필요하다. 이를 차원 축소 (dimensionality reduction)라 한다. 물론 필요한 일부 변수의 데이터만 뽑아서 사용할 수도 있지만(Feature Selection), 변수들 사이에 어떤 밀접한 관계가 있는지는 미리 알 수가 없어서 이 경우 원 데이터의 분포를 잘 반영한다고 볼 수는 없다.

- 주성분 분석(Principal Component Analysis, PCA)은 가장 널리 사용되는 차원 축소 기법 중 하나로, 원 데이터의 분포를 최대한 보존하면서 서로 직교하는 새 기저(축)를 찾아, 고차원 공간의 데이터들을 선형 연관성이 없는 저차원 공간으로 변환한다. 이때 계산은 주로 행렬의 고윳값 분해 또는 특이값 분해(SVD)를 사용한다. [Key Idea 4]

- PCA는 기준의 변수를 일차 결합하여 서로 선형 연관성이 없는 새로운 변수, 주 주성분(principal component, PC)들을 만들어 낸다. 첫 번째 주성분 PC1이 원 데이터의 분포를 가장 많이 보존하고, 두 번째 주성분 PC2가 그 다음으로 원 데이터의 분포를 많이 보존한다. 예를 들어, PC1, PC2, PC3가 원 데이터의 분포(성질)의 약 90%를 보존한다면, 10% 정도의 정보는 잃어버리더라도, 합리적인 분석에 큰 무리가 없으므로, PC1, PC2, PC3만 택하여 3차원 데이터로 차원을 줄일 수 있다. 이 경우 계산과 시각화가 용이하여 데이터를 쉽게 분석할 수 있다.

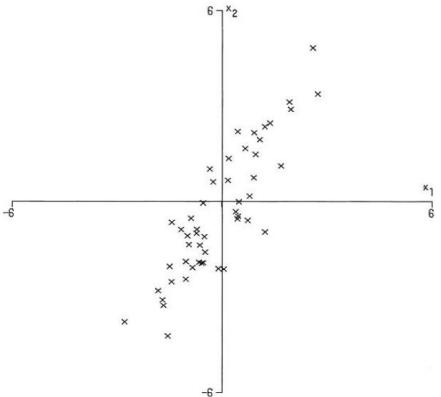


Figure 1.1. Plot of 50 observations on two variables x_1, x_2 .

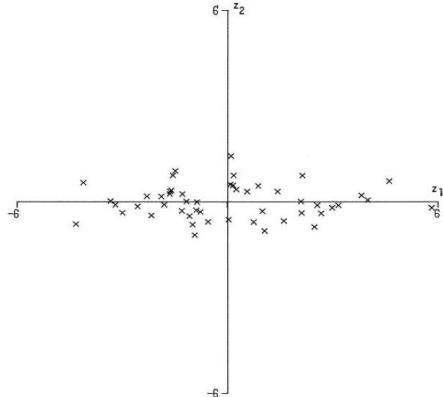


Figure 1.2. Plot of the 50 observations from Figure 1.1 with respect to their PCs

[그림 출처] Jolliffe, I.T. Principal Component Analysis, Springer, 2002

1.2 Principal Component의 유도

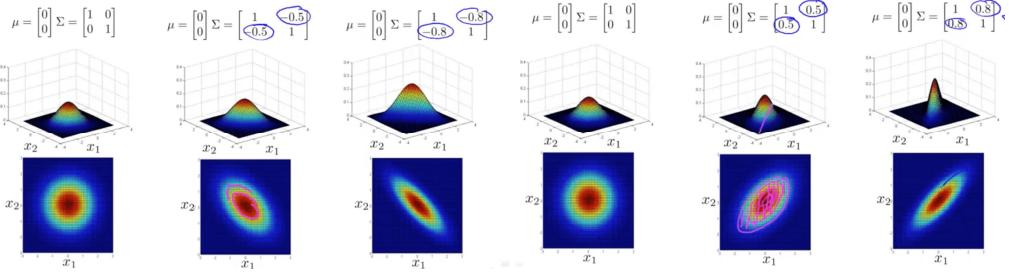
- 앞서 두 개의 확률변수 X 와 Y 에 대하여 공분산을

$$\text{Cov}(X, Y) = \sigma_{xy} = E[(X - \mu_x)(Y - \mu_y)] = E(XY) - \mu_x\mu_y$$

로 정의하였다. 그리고 p 개의 확률변수 $\{x_1, \dots, x_p\}$ 에 대한 공분산 행렬 (covariance matrix)은 (i, j) 성분이 $i \neq j$ 일 때는 i 번째 확률변수 x_i 와 j 번째 확률변수 x_j 사이의 공분산 σ_{ij} 으로, $i = j$ 일 때는 i 번째 확률변수의 분산 $\sigma_{ii} = \sigma_i^2$ 으로 하는 $p \times p$ 행렬로 정의하고 Σ 로 표기하였다. 쉽게 말하면, 정사각행렬의 성분을 각 변수의 분산(주대각선)과 공분산으로 채운 것이 바로 공분산 행렬이다.

$$\Sigma = \begin{bmatrix} \text{Var}(x_1) & \text{Cov}(x_1, x_2) & \dots & \text{Cov}(x_1, x_p) \\ \text{Cov}(x_2, x_1) & \text{Var}(x_2) & \dots & \text{Cov}(x_2, x_p) \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}(x_p, x_1) & \text{Cov}(x_p, x_2) & \dots & \text{Var}(x_p) \end{bmatrix} = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \dots & \sigma_{1p} \\ \sigma_{21} & \sigma_2^2 & \dots & \sigma_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{p1} & \sigma_{p2} & \dots & \sigma_p^2 \end{bmatrix}$$

공분산 행렬은 아래 그림과 같이 데이터의 분포를 나타낸다고 볼 수 있다.



[그림 출처] <https://www.ritchieng.com/machine-learning-anomaly-detection/>

- 이제 X 를 $n \times p$ 의 데이터 행렬(Data matrix)이라 하자. 여기서 n 은 표본(sample)의 개수이고, p 는 데이터의 특성(feature)을 나타내는 확률변수 $\{x_1, \dots, x_p\}$ 의 개수이다. 데이터 행렬 X 의 (i, j) 성분 x_{ij} 는 i 번째 표본의 j 번째 확률변수 x_j 에 대한 하나의 데이터를 의미하고, j 열 $X^{(j)}$ 는 확률변수 x_j 가 갖는 모든 데이터를 의미한다.

$$X = \begin{bmatrix} \text{확률변수} & x_1 & x_2 & \dots & x_p \\ & \boxed{x_{11}} & x_{12} & \dots & x_{1p} \\ & \vdots & \vdots & & \vdots \\ & x_{n1} & x_{n2} & \dots & x_{np} \end{bmatrix}_{n \times p}$$

x_1 의 모든 데이터

- 위의 X 를 센터링(centering)하여, \tilde{X} 를 정의하자. 이를 위해
 - X 의 각 열(하나의 확률변수가 갖는 데이터)의 평균을 구한다. 확률변수 x_j 의 평균은 \bar{x}_j 로 표기한다.

$$\bar{x}_1 = \frac{1}{n} (x_{11} + x_{21} + \dots + x_{n1}) , \dots , \bar{x}_p = \frac{1}{n} (x_{1p} + x_{2p} + \dots + x_{np}) .$$

- X 의 각 열(column)별로 데이터에서 (열의) 평균을 뺀다. 이 행렬을 센터링된 행렬 \tilde{X} 라 한다.

$$\tilde{X} = X - \begin{bmatrix} \bar{x}_1 & \bar{x}_2 & \dots & \bar{x}_p \\ \vdots & \vdots & & \vdots \\ x_1 & x_2 & \dots & x_p \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \dots & x_{np} \end{bmatrix} - \begin{bmatrix} \bar{x}_1 & \bar{x}_2 & \dots & \bar{x}_p \\ \vdots & \vdots & & \vdots \\ x_1 & x_2 & \dots & x_p \end{bmatrix}$$

여기서 센터링된 행렬 \tilde{X} 의 각 열의 평균은 0이 된다. 이런 이유로 센터링된 행렬을 mean-centered 행렬이라 부른다. 앞으로 주어진 데이터 행렬 X 는 이미 센터링 되어 있는 행렬이라고 가정하자.

- 데이터 행렬 X 가 센터링(mean-centered) 되어 있는 행렬이면, $p \times p$ 공분산 행렬은 다음과 같이 계산된다.

$$\begin{aligned} & \begin{bmatrix} \text{Var}(x_1) & \text{Cov}(x_1, x_2) & \cdots & \text{Cov}(x_1, x_p) \\ \text{Cov}(x_2, x_1) & \text{Var}(x_2) & \cdots & \text{Cov}(x_2, x_p) \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}(x_p, x_1) & \text{Cov}(x_p, x_2) & \cdots & \text{Var}(x_p) \end{bmatrix} = \boldsymbol{\Sigma} = \frac{1}{n} X^T X \\ & = \frac{1}{n} \begin{bmatrix} x_{11} & x_{21} & \cdots & x_{n1} \\ \vdots & \vdots & & \vdots \\ x_{1p} & x_{2p} & \cdots & x_{np} \end{bmatrix} \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} = \begin{bmatrix} \frac{1}{n} \sum_{i=1}^n x_{i1}^2 & \frac{1}{n} \sum_{i=1}^n x_{i1}x_{i2} & \cdots & \frac{1}{n} \sum_{i=1}^n x_{i1}x_{ip} \\ \frac{1}{n} \sum_{i=1}^n x_{i2}x_{i1} & \frac{1}{n} \sum_{i=1}^n x_{i2}^2 & \cdots & \frac{1}{n} \sum_{i=1}^n x_{i2}x_{ip} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{n} \sum_{i=1}^n x_{ip}x_{i1} & \frac{1}{n} \sum_{i=1}^n x_{ip}x_{i2} & \cdots & \frac{1}{n} \sum_{i=1}^n x_{ip}^2 \end{bmatrix} \end{aligned}$$

- 우리의 목표는 ‘공분산 행렬로 부터 얻은 정보’를 최대한 보존하는 ‘더 적은 개수($\ll p$)의 새로운 변수들’을 찾으려는 것이다. *[dimension reduction]*

▶ [단계 1]

확률변수 $\{x_1, \dots, x_p\}$ 의 일차결합으로 만든 첫 번째 주성분

$$z_1 = v_{11}x_1 + v_{21}x_2 + \cdots + v_{p1}x_p$$

의 분산이 최대가 되도록 하고자 한다. 여기서 p 개의 상수 $v_{11}, v_{21}, \dots, v_{p1}$ 는 적재 계수(loadings)라 부르기도 한다. z_1 의 평균은 0이므로, z_1 의 분산은 다음과 같이 계산된다. ($\because E(z_1) = E(v_{11}x_1 + \cdots + v_{p1}x_p) = v_{11}E(x_1) + v_{p1}E(x_p) = 0$)

$$\begin{aligned} \text{Var}(z_1) &= \text{Var}(v_{11}x_1 + v_{21}x_2 + \cdots + v_{p1}x_p) \\ &= E((v_{11}x_1 + v_{21}x_2 + \cdots + v_{p1}x_p)^2) \quad (\because E(X) = 0 \Rightarrow V(X) = E(X^2)) \\ &= E\left(\sum_{i=1}^p \sum_{j=1}^p v_{i1}v_{j1}x_i x_j\right) \end{aligned}$$

$$\begin{aligned}
&= \sum_{i=1}^p \sum_{j=1}^p v_{i1} v_{j1} E(x_i x_j) \quad (\because E(aX+bY) = aE(X)+bE(Y) \text{ 이므로}) \\
&= \sum_{i=1}^p \sum_{j=1}^p v_{i1} v_{j1} \sigma_{ij} \quad (\because E(x_i x_j) \text{은 } x_i \text{와 } x_j \text{의 공분산})
\end{aligned}$$

위 식에 Part I에서 학습한 **이차형식** $\left(\mathbf{x}^T A \mathbf{x} = \sum_{i=1}^n \sum_{j=1}^n a_{ij} x_i x_j \right)$ 을 이용하면 z_1 의 분산은 다음과 같이 나타낼 수 있다.

$$Var(z_1) = \mathbf{v}_1^T \Sigma \mathbf{v}_1$$

여기서 \mathbf{v}_1 은 적재계수 $v_{11}, v_{21}, \dots, v_{p1}$ 을 하나의 (열) 벡터로 나타낸 것이다.

따라서 [단계 1]은 결국 아래 문제를 푸는 것이다. 이때 \mathbf{v}_1 의 방향에만 관심이 있으므로 \mathbf{v}_1 의 크기를 1로 제한한다.

$$\begin{aligned}
&\text{maximize } Var(z_1) = \mathbf{v}_1^T \Sigma \mathbf{v}_1 \quad (= f(\mathbf{v}_1)) \\
&\text{subject to } \mathbf{v}_1^T \mathbf{v}_1 = 1 \quad (g(\mathbf{v}_1) = \mathbf{v}_1^T \mathbf{v}_1 - 1 = 0)
\end{aligned}$$

- 위 문제에 **라그랑주의 승수법**(Method of Lagrange multipliers)을 사용한다.
([참고] <https://bit.ly/36MRU1O>)

라그랑주의 승수법	
문제	maximize(또는 minimize) $f(x, y)$ subject to $g(x, y) = 0$
[1단계]	라그랑주 함수 $L(x, y, \lambda) = f(x, y) - \lambda g(x, y)$ 를 만든다.
[2단계]	$\nabla L(x, y, \lambda) = 0$, 즉 $\begin{cases} \nabla f(x, y) - \lambda \nabla g(x, y) = 0 \\ g(x, y) = 0 \end{cases}$ 을 만족하는 점 (x, y) 를 모두 구한다.
[3단계]	[2단계]에서 구한 모든 점에서 함수 f 의 값을 구한다. 이 중 가장 큰 값이 최댓값, 가장 작은 값이 최솟값이 된다.

λ 를 라그랑주 승수라 할 때, $f(\mathbf{v}_1) = \mathbf{v}_1^T \Sigma \mathbf{v}_1$, $g(\mathbf{v}_1) = \mathbf{v}_1^T \mathbf{v}_1 - 1 = 0$ 이고

$$L(\mathbf{v}_1, \lambda) = f(\mathbf{v}_1) - \lambda g(\mathbf{v}_1) = \mathbf{v}_1^T \Sigma \mathbf{v}_1 - \lambda (\mathbf{v}_1^T \mathbf{v}_1 - 1)$$

의 \mathbf{v}_1 에 관한 그래디언트가 $\mathbf{0}$ 이 되어야 하므로, $2(\Sigma \mathbf{v}_1 - \lambda \mathbf{v}_1) = \mathbf{0}$, 즉 $(\Sigma - \lambda I_p) \mathbf{v}_1 = \mathbf{0}$ 이다. 따라서 λ 는 Σ 의 고윳값이고, \mathbf{v}_1 은 그에 대응하는 고유벡터가 된다. Σ 의 가장 큰 고윳값을 λ_1 이라 하고 그에 대응하는 단위고유벡터를 \mathbf{v}_1 이라 하면, $Var(z_1) = \mathbf{v}_1^T \Sigma \mathbf{v}_1 = \lambda_1 \mathbf{v}_1^T \mathbf{v}_1 = \lambda_1$ 이 된다.

▶ [단계 2]

마찬가지로 두 번째 주성분 $z_2 = v_{12}x_1 + v_{22}x_2 + \dots + v_{p2}x_p$ 의 분산이 최대가 되도록 하되, z_1 과는 상관관계가 없도록 z_2 를 찾아야 한다. 이는 다음 문제를 푸는 것과 같다. 여기서 \mathbf{v}_2 는 적재계수 $v_{12}, v_{22}, \dots, v_{p2}$ 을 하나의 벡터로 나타낸 것이다.

$$\begin{aligned} & \text{maximize} \quad Var(z_2) = \mathbf{v}_2^T \Sigma \mathbf{v}_2 \quad (= f(\mathbf{v}_2)) \\ & \text{subject to} \quad \mathbf{v}_2^T \mathbf{v}_2 = 1 \quad (g_1(\mathbf{v}_2) = \mathbf{v}_2^T \mathbf{v}_2 - 1 = 0) \\ & \quad \mathbf{v}_2^T \mathbf{v}_1 = 0 \quad (g_2(\mathbf{v}_2) = \mathbf{v}_2^T \mathbf{v}_1 = 0) \end{aligned}$$

이때 z_1 과 z_2 는 상관관계가 없어야 하므로

$$\begin{aligned} \text{Cov}[z_1, z_2] &= \mathbf{v}_1^T \Sigma \mathbf{v}_2 = \mathbf{v}_2^T \Sigma \mathbf{v}_1 = \mathbf{v}_2^T \lambda \mathbf{v}_1 = \lambda \mathbf{v}_2^T \mathbf{v}_1 = \lambda \mathbf{v}_1^T \mathbf{v}_2 = 0 \\ \Rightarrow \mathbf{v}_1^T \Sigma \mathbf{v}_2 &= 0, \quad \mathbf{v}_2^T \Sigma \mathbf{v}_1 = 0, \quad \mathbf{v}_1^T \mathbf{v}_2 = 0, \quad \mathbf{v}_2^T \mathbf{v}_1 = 0 \end{aligned}$$

으로 부터 $\mathbf{v}_2^T \mathbf{v}_1 = 0$ 을 만족해야 한다.

[단계 1]과 마찬가지로 라그랑주의 승수법을 사용하면 다음을 얻는다. 즉 λ 와 ϕ 를 라그랑주 승수라 할 때,

$$L(\mathbf{v}_2, \lambda, \phi) = f(\mathbf{v}_2) - \lambda g_1(\mathbf{v}_2) - \phi g_2(\mathbf{v}_2) = \mathbf{v}_2^T \Sigma \mathbf{v}_2 - \lambda (\mathbf{v}_2^T \mathbf{v}_2 - 1) - \phi \mathbf{v}_2^T \mathbf{v}_1$$

의 \mathbf{v}_2 에 관한 그래디언트가 $\mathbf{0}$ 이 되어야 하므로 같은 방식으로

$$\boldsymbol{\Sigma}\mathbf{v}_2 - \lambda\mathbf{v}_2 - \frac{1}{2}\phi\mathbf{v}_1 = \mathbf{0}$$

이고, 이 식의 양변에 [단계 1]에서 얻은 \mathbf{v}_1 을 내적하면

$$\langle \mathbf{0}, \mathbf{v}_1 \rangle = \langle \boldsymbol{\Sigma}\mathbf{v}_2 - \lambda\mathbf{v}_2 - \frac{1}{2}\phi\mathbf{v}_1, \mathbf{v}_1 \rangle = \mathbf{v}_1^T \boldsymbol{\Sigma}\mathbf{v}_2 - \lambda\mathbf{v}_1^T \mathbf{v}_2 - \frac{1}{2}\phi\mathbf{v}_1^T \mathbf{v}_1 = 0$$

이다. 그러면 $\mathbf{v}_1^T \boldsymbol{\Sigma}\mathbf{v}_2 = 0$ 이고 $\mathbf{v}_1^T \mathbf{v}_2 = 0$ 인데 $\mathbf{v}_1^T \mathbf{v}_1 \neq 0$ 이므로 $\phi = 0$ 이어야 한다.

따라서 $\boldsymbol{\Sigma}\mathbf{v}_2 - \lambda\mathbf{v}_2 - \frac{1}{2}\phi\mathbf{v}_1 = \mathbf{0} \Rightarrow \boldsymbol{\Sigma}\mathbf{v}_2 - \lambda\mathbf{v}_2 = \mathbf{0} \Leftrightarrow (\boldsymbol{\Sigma} - \lambda I_p)\mathbf{v}_2 = \mathbf{0}$ 이다. 이를 통해 λ 는 $\boldsymbol{\Sigma}$ 의 두 번째로 큰 고윳값(λ_2)이고, \mathbf{v}_2 는 그에 대응하는 (단위)고유벡터가 됨을 알 수 있다. 역시 $\text{Var}(z_2) = \mathbf{v}_2^T \boldsymbol{\Sigma} \mathbf{v}_2 = \lambda_2 \mathbf{v}_2^T \mathbf{v}_2 = \lambda_2$ 이 성립한다.

▶ [단계 3]

같은 방법으로 k 번째 주성분 $z_k = v_{1k}x_1 + v_{2k}x_2 + \dots + v_{pk}x_p$ 의 분산이 최대가 되도록 하되 z_1, z_2, \dots, z_{k-1} 과는 상관관계가 없도록 z_k 를 찾아야 한다. 이는 다음 문제를 푸는 것과 같다.

$$\begin{aligned} & \text{maximize} \quad \text{Var}(z_k) = \mathbf{v}_k^T \boldsymbol{\Sigma} \mathbf{v}_k \\ & \text{subject to} \quad \mathbf{v}_k^T \mathbf{v}_k = 1 \\ & \quad \mathbf{v}_k^T \mathbf{v}_1 = 0, \quad \mathbf{v}_k^T \mathbf{v}_2 = 0, \quad \dots, \quad \mathbf{v}_k^T \mathbf{v}_{k-1} = 0 \end{aligned}$$

마찬가지로 라그랑주의 승수법을 사용하면, λ 는 $\boldsymbol{\Sigma}$ 의 k 번째로 큰 고윳값(λ_k)이고, \mathbf{v}_k 가 그에 대응하는 (단위)고유벡터가 됨을 알 수 있다. 따라서 $\text{Var}(z_k) = \mathbf{v}_k^T \boldsymbol{\Sigma} \mathbf{v}_k = \lambda_k \mathbf{v}_k^T \mathbf{v}_k = \lambda_k$ 이 성립한다.

1.3 PC score

- X 를 센터링(mean-centered)된 $n \times p$ 의 데이터 행렬이라 하자. 여기서 n 은 표본(sample)의 개수이고, p 는 확률변수(variable, feature)의 개수이다. 공분산 행렬

$\Sigma = \frac{1}{n} X^T X$ 는 $p \times p$ 대칭행렬이므로, 다음과 같이 직교대각화가 가능하다.

$$\Sigma = V \Lambda V^T$$

여기서 Λ 는 Σ 의 고윳값 λ_i 를 크기 순서대로, 주대각선 성분에 배열한 대각선행렬이고, V 는 그에 대응하는 정규직교인 고유벡터를 열벡터로 하는 직교행렬이다. 또한 $\mathbf{v}^T (X^T X) \mathbf{v} = (X\mathbf{v})^T (X\mathbf{v}) = \langle X\mathbf{v}, X\mathbf{v} \rangle \geq 0$ 이므로 Σ 는 양의 준정부호(positive semidefinite) 행렬이고, 고윳값은 $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_p \geq 0$ 이다. 그리고 고유벡터는 주축(principal axes) 또는 주방향(principal directions)이라고도 부른다.

- 주성분(PC) z_1, z_2, \dots, z_p 를 벡터로 나타내면 다음과 같다.

$$[z_1, z_2, \dots, z_p] = [x_1, x_2, \dots, x_p] [\mathbf{v}_1 \mid \mathbf{v}_2 \mid \dots \mid \mathbf{v}_p] = [x_1, x_2, \dots, x_p] V$$

그리고 각 확률변수 x_i 에 원 데이터를 대입하면 행렬 $Z = XV$ 를 얻는다.

주성분	원 확률변수
$[z_1, z_2, \dots, z_p]$	$= [x_1, x_2, \dots, x_p] V$
$\begin{bmatrix} z_{11} & z_{12} & \cdots & z_{1p} \\ \vdots & \vdots & & \vdots \\ z_{n1} & z_{n2} & \cdots & z_{np} \end{bmatrix}$	$= \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} V$
1번째 PC에서의 score	

그러면 Z 의 i 번째 데이터의 좌표는 XV 의 i 번째 행벡터로 주어지고, j 번째 PC score (주성분점수, principal component score)는 XV 의 j 번째 열벡터로 주어진다.

■ 이제 X 의 특이값분해(SVD)가 다음과 같이 주어져 있다고 하자.

$$X = [\mathbf{u}_1 \ \mathbf{u}_2 \ \cdots \ \mathbf{u}_k \ \mathbf{u}_{k+1} \ \cdots \ \mathbf{u}_n] = \begin{bmatrix} s_1 & 0 & \cdots & 0 & | & 0 & \cdots & 0 \\ 0 & s_2 & \cdots & 0 & | & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & | & \vdots & & \vdots \\ 0 & 0 & \cdots & s_k & | & 0 & \cdots & 0 \\ \hline 0 & 0 & \cdots & 0 & | & s_{k+1} & \cdots & 0 \\ \vdots & \vdots & & \vdots & | & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & | & 0 & \cdots & s_p \\ \hline 0 & 0 & \cdots & 0 & | & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & | & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & | & 0 & \cdots & 0 \end{bmatrix} = U S V^T = U \begin{bmatrix} S_1 \\ O \end{bmatrix} V^T = \sum_{i=1}^p s_i \mathbf{u}_i \mathbf{v}_i^T$$

여기서 U, V 는 직교행렬이고, S_1 는 크기 순서대로 배열된 특이값(singular value) $s_1 \geq s_2 \geq \cdots \geq s_p \geq 0$ 을 주대각선 성분으로 하는 $p \times p$ 대각선행렬 $S_1 = \text{diag}(s_1, \dots, s_p)$ 이다. 그러면 관계식

$$\Sigma = V \Lambda V^T = \frac{1}{n} X^T X = \frac{1}{n} V [S_1^T \ O] U^T U \begin{bmatrix} S_1 \\ O \end{bmatrix} V^T = V \left(\frac{S_1^2}{n} \right) V^T$$

으로부터 다음을 얻는다.

- (1) $X = USV^T$ 라 하면 V 의 열벡터가 주축(principal axes)이 된다.
- (2) $Z = XV = USV^T V = US$ 이므로, US 의 열벡터들이 PC score (주성분점수, principal component score)가 된다.
- (3) 특이값 s_i 는 Σ 의 고윳값 λ_i 와 관계식 $\lambda_i = \frac{s_i^2}{n}$ 을 만족하며, λ_i 는 대응하는 PC의 분산이 된다.
- (4) 데이터를 p 차원에서 k ($\ll p$) 차원으로 줄이기 위하여, U 의 처음 k 개의 열벡터(U_k)와 S 의 k 번째 선행 주 부분행렬(leading principal submatrix)(S_k)을 택

하면, $U_k S_k$ 는 처음 k 개의 PC를 포함하는 $n \times k$ 행렬이 된다.

$$Z_k = U_k S_k = [\mathbf{u}_1 \ \mathbf{u}_2 \ \cdots \ \mathbf{u}_k] \begin{bmatrix} s_1 & & & \\ & s_2 & & \\ & & \ddots & \\ & & & s_k \end{bmatrix}.$$

(5) (4)에서 얻은 Z_k 에 s_1, s_2, \dots, s_k 에 대응되는 주축으로 이루어진 행렬 V_k^T 을 곱하면, $X_k = U_k S_k V_k^T$ 는 처음 k 개의 PC로부터 원 데이터를 복원하도록 해주는 rank가 k 인 $n \times p$ 행렬이 된다. *[rank reduction]*

$$X_k = U_k S_k V_k^T = [\mathbf{u}_1 \ \mathbf{u}_2 \ \cdots \ \mathbf{u}_k] \begin{bmatrix} s_1 & 0 & \cdots & 0 \\ 0 & s_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & s_k \end{bmatrix} \begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_k^T \end{bmatrix} = \sum_{i=1}^k s_i \mathbf{u}_i \mathbf{v}_i^T.$$

이는 X 의 SVD에서 크기 순서대로 k 개의 큰 특이값 $s_1 \geq s_2 \geq \cdots \geq s_k > 0$ 과 이에 대응되는 U 와 V 의 처음 k 개의 열벡터를 택하는 것과 같다. 이를 [truncated SVD](#)라고 한다. 실제로 X_k 는 rank k 인 행렬 중에서 X 에 가장 가까운 행렬이 된다. 즉 다음이 성립한다.

$$X \approx X_k := \sum_{i=1}^k s_i \mathbf{u}_i \mathbf{v}_i^T = \operatorname{argmin}_{\operatorname{rank}(W) \leq k} \|X - W\|_F^2.$$

1.4 PCA 예제

이제 PCA 예제를 살펴보자. 자료의 출처는 다음과 같다.

[출처] <http://yatani.jp/teaching/doku.php?id=hcistats:PCA>

이를 Sage에서 R 코드로 구현한 자료는 <http://math1.skku.ac.kr/home/pub/212/> 에서 확인할 수 있다.

사람들이 새 컴퓨터를 선택할 때 관심을 갖는 아래 사항에 관하여 척도가 7점인 4문항의 리커트(Likert) 설문 조사를 16명에게 실시하였다.
(1: 매우 그렇지 않다 - 7: 매우 그렇다)

Price	가격이 저렴하다.
Software	운영체제가 사용하려는 소프트웨어와 호환된다.
Aesthetics	디자인이 매력적이다.
Brand	유명 브랜드의 제품이다.

- 아래와 같이 설문조사를 통하여 얻은 데이터를 입력하여 데이터 행렬(data)을 생성한다.

[R code] <http://mathlab.knou.ac.kr:8080/> <http://mathlab.knou.ac.kr/r/> <http://sage.skku.edu/>

```
# 데이터 입력, R 코드
Price <- c(6, 7, 6, 5, 7, 6, 5, 6, 3, 1, 2, 5, 2, 3, 1, 2)
Software <- c(5, 3, 4, 7, 7, 4, 7, 5, 5, 3, 6, 7, 4, 5, 6, 3)
Aesthetics <- c(3, 2, 4, 1, 5, 2, 2, 4, 6, 7, 6, 7, 5, 6, 5, 7)
Brand <- c(4, 2, 5, 3, 5, 3, 1, 4, 7, 5, 7, 6, 6, 5, 5, 7)
data <- data.frame(Price, Software, Aesthetics, Brand)

# 데이터가 잘 입력되었는지 확인한다.
data
```

	Price	Software	Aesthetics	Brand
1	6	5	3	4
2	7	3	2	2
3	6	4	4	5
4	5	7	1	3
5	7	7	5	5
6	6	4	2	3
7	5	7	2	1
8	6	5	4	4
9	3	5	6	7
10	1	3	7	5
11	2	6	6	7
12	5	7	7	6

13	2	4	5	6
14	3	5	6	5
15	1	6	5	5
16	2	3	7	7

- 데이터 행렬을 센터링 하여 mydata 행렬을 얻는다. (본 예제에서는 센터링 한 후에 다시 각 열을 해당 확률변수의 표준편차로 나누었다. 이를 scaling이라 하는데, 필요에 따라 scaling도 진행할 수 있다.)

```
# 표준화(centering, scaling) 작업
```

```
mydata <- scale(data, center = T, scale = T) # 데이터 행렬을 센터링, scaling 한다.
```

	Price	Software	Aesthetics	Brand
[1,]	0.8485483	-0.04217745	-0.75	-0.3865961
[2,]	1.3167129	-1.39185589	-1.25	-1.5112392
[3,]	0.8485483	-0.71701667	-0.25	0.1757255
[4,]	0.3803837	1.30750098	-1.75	-0.9489176
[5,]	1.3167129	1.30750098	0.25	0.1757255
[6,]	0.8485483	-0.71701667	-1.25	-0.9489176
[7,]	0.3803837	1.30750098	-1.25	-2.0735607
[8,]	0.8485483	-0.04217745	-0.25	-0.3865961
[9,]	-0.5559454	-0.04217745	0.75	1.3003686
[10,]	-1.4922746	-1.39185589	1.25	0.1757255
[11,]	-1.0241100	0.63266177	0.75	1.3003686
[12,]	0.3803837	1.30750098	1.25	0.7380470
[13,]	-1.0241100	-0.71701667	0.25	0.7380470
[14,]	-0.5559454	-0.04217745	0.75	0.1757255
[15,]	-1.4922746	0.63266177	0.25	0.1757255
[16,]	-1.0241100	-1.39185589	1.25	1.3003686

attr("scaled:center")

Price	Software	Aesthetics	Brand
4.1875	5.0625	4.5000	4.6875 # 각 확률변수의 평균을 의미한다.

attr("scaled:scale")

Price	Software	Aesthetics	Brand
2.136001	1.481834	2.000000	1.778342 # 각 확률변수의 표준편차를 의미한다.

- mydata 행렬에 PCA를 진행한다. (R의 prcomp 명령어를 이용하였다.)

```
# PCA 진행. 앞에서 이미 센터링 scaling 하였음. retx는 PC score를 제공할지 여부
mypca <- prcomp(mydata, center = F, scale = F, retx = T)
summary(mypca)
```

Importance of components:

	PC1	PC2	PC3	PC4
Standard deviation	1.5589	0.9804	0.6817	0.37926

주성분의 표준편차를 나타낸다. 따라서 위 값의 제곱이 공분산행렬의 고유값, 분산이 된다.

Proportion of Variance 0.6076 0.2403 0.1162 0.03596

주성분이 보존하는 데이터의 비율을 의미한다. 이는 분산의 비율 $\lambda_i / (\sum \lambda_i)$ 에 대입하여 계산한다. 예를 들어 0.6076의 경우, 다음과 같이 계산된다.

$$1.5589^2 / (1.5589^2 + 0.9804^2 + 0.6817^2 + 0.37926^2) = 0.607556518168168$$

Cumulative Proportion 0.6076 0.8479 0.9640 1.00000

위 Proportion of Variance의 누적된 값이다. 예를 들어, $0.6076 + 0.2403 = 0.8479$ 이므로 PC1, PC2가 원 데이터의 84.8% 정도를 보존한다고 이해할 수 있다.

- 주성분을 찾을 때 필요한 적재계수는 다음 명령어를 이용하여 확인할 수 있다. 이는 직교행렬 V 에 해당한다.

```
# loadings (V를 의미한다.)
mypca$rotation
```

	PC1	PC2	PC3	PC4
Price	-0.5229138	0.00807487	-0.8483525	0.08242604
Software	-0.1771390	0.97675554	0.1198660	0.01423081
Aesthetics	0.5965260	0.13369503	-0.2950727	0.73431229
Brand	0.5825287	0.16735905	-0.4229212	-0.67363855

- 1~3번째 PC에서의 score는 다음 명령어를 이용하여 확인할 수 있다. 이는 행렬 $Z = XV$ 의 1~3번째 열벡터로 이루어진 행렬이다.

```
# PC1~PC3 score. 1열부터 3열까지
mypca$x[, 1:3]
```

	PC1	PC2	PC3
[1,]	-1.1088441	-0.19931676	-0.34011951
[2,]	-2.0679730	-1.76890900	-0.27589687
[3,]	-0.3634723	-0.69751259	-0.80636384
[4,]	-2.0272096	0.88740412	0.75172002
[5,]	-0.6686402	1.35057412	-1.10839793
[6,]	-1.6151352	-1.01942683	-0.03565573
[7,]	-2.3840835	0.76603244	1.07981912
[8,]	-0.8105812	-0.13246925	-0.48765585
[9,]	1.5030793	0.27221348	-0.30467593
[10,]	1.8749056	-1.17502483	0.65597978
[11,]	1.6283487	0.92758606	0.17338293
[12,]	0.7450737	1.57081802	-0.84695116
[13,]	1.2415980	-0.55167695	0.39695643
[14,]	0.8479424	0.08399428	0.17095950
[15,]	0.9197585	0.66873897	1.19372328
[16,]	2.2852328	-0.98302526	-0.21682424

한 예로 mydata의 1행(붉은색)과 mypca\$rotation의 1열(붉은색)을 곱하면 mypca\$x[, 1:3]의 (1, 1) 성분인 -1.1088441(붉은색)를 얻는다.

- 앞서 PC1, PC2가 원 데이터의 84.8% 정도를 보존한다고 이해할 수 있으므로, 15% 정도의 정보는 잃어버리더라도, PC1, PC2만 택하여 2차원 데이터로 차원을 줄일 수 있다. 이를 활용하여 데이터를 시각화하거나 다른 데이터 분석기법을 적용할 수 있다. 예를 들어, 다음과 같이 주어진 데이터를 k 개의 클러스터로 묶는 알고리즘인 **K-means clustering**을 진행할 수 있다(아래 왼쪽 그림 참조). 그리고 주성분의 고윳값에 대한 선 그림(line plot)인 **scree plot**을 활용하여 데이터의 차원을 줄일 때, 주성분을 몇 개로 선택할지 결정할 수 있다(아래 오른쪽 그림 참조). scree plot은 고윳값을 가장 큰 값에서부터 크기 순서대로 정렬하므로, 대개 기울기가 꺾이는 부분의 “elbow point” 왼쪽에 있는 성분까지 선택한다.

```

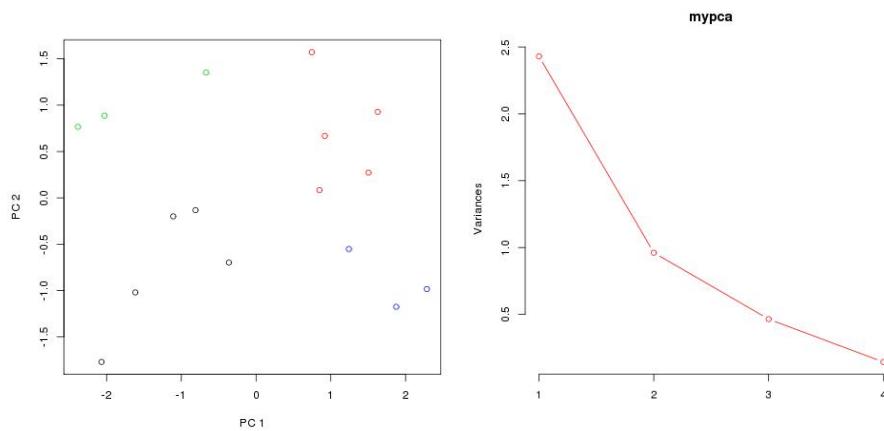
# 2차원으로 차원 축소 후(PC1과 PC2만 취하여) k-means clustering 진행
PC1 = mypca$x[, 1] # 1열을 PC1라 함
PC2 = mypca$x[, 2] # 2열을 PC2라 함

# PC1, PC2를 이용하여 클러스터의 개수(centers)를 4개로 하고
# 임의로 10번 시행(nstart)하여 가장 잘 clustering 하는 것을 찾음
myclust <- kmeans(mypca$x[, 1:2], centers = 4, nstart = 10)

# clustering 후 시각화 결과 보여줌
plot(PC1, PC2, xlab = "PC 1", ylab = "PC 2", type="p", col = myclust$cluster)

# scree 그래프 (분산을 나타낸다)
screeplot(mypca, type = "lines", col = 2)
dev.off()

```



- 앞서 언급한 1974년 Motor Trend magazine에 실린 1973년~74년도 자동차 32대의 자료에 관하여 PCA를 적용한 결과는 <http://www.people.vcu.edu/~jpbrooks/pctutorial/> 를 참조하라.

2. 신경망

26강 동영상, 빅데이터와 인공지능, 신경망 https://youtu.be/D_oCT-tEW_4 (13:06)

26강 part 2, 인공신경망(은닉망의 수학) https://youtu.be/d4WercT_OnU (18:12)

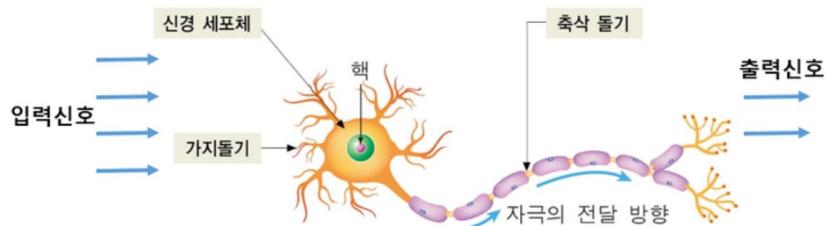


2.1 신경망(Neural Network)

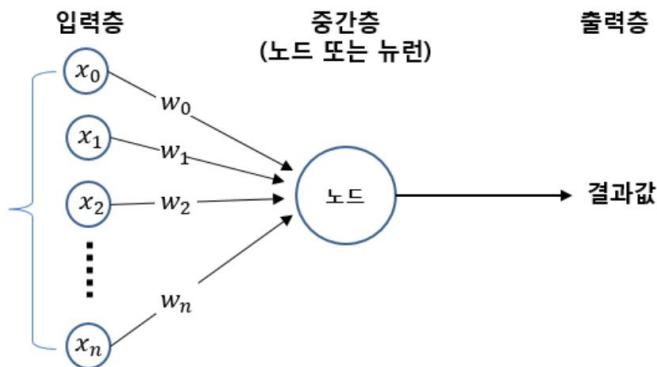
- 알고리즘에서 기계학습으로 변하는 과정은 수학적모델링을 통하여 함수를 구하는 방식에서, 수많은 데이터를 기반으로 기계학습을 통하여 행렬을 구하는 과정으로 변한 것으로 이해하면 쉽다. 이 과정에서 신경망을 이용하는 것이다.



- 신경망(neural network)은 신경계의 기본 단위인 뉴런(신경세포, neuron)을 모델화 한 것으로, 인간의 뇌는 약 1000억 개의 뉴런을 가지고, 컴퓨터에게는 어려운 이미지 인식과 같은 업무를 잘 수행한다. 아래 그림은 각각 실제 뉴런과 이를 모델화한 인공 뉴런이다.



[그림 출처] 블로그 <https://m.blog.naver.com/PostList.nhn?blogId=samsjang>
파셉트론 - 인공신경망의 기초개념

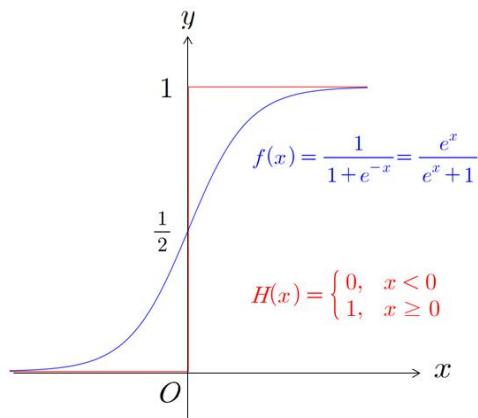


- 하나의 인공 뉴런에서는 다수의 입력 신호 x_i 를 받아서 하나의 신호를 출력한다. 이는 실제 뉴런에서 전기신호를 내보내 정보를 전달하는 것과 비슷하다. 이때 뉴런의 돌기가 신호를 전달하는 역할을 인공 뉴런에서는 가중치(weight) w_i 가 그 역할을 한다. 각 입력신호에는 고유한 가중치가 부여되며 가중치가 클수록 해당 신호가 중요하다고 볼 수 있다.
- 뉴런에서 임계값(threshold) 이상의 자극이 주어져야 감각이 반응하는 것처럼, 인공 뉴런에서도 다수의 입력 신호가 주어지면, 미리 부여된 가중치와 계산을 한 후 그 총합이 정해진 임계값을 넘으면, 1을 출력하고 넘지 못하면 0 (또는 -1)을 출력한다. 이때 출력을 결정하는 함수를 활성화 함수(activation function)라 한다. 주로 사용되는 활성화 함수로는 유계이며 미분 가능한 실함수인 sigmoid 함수가 있다.

$$f(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1}$$

이는 임계값을 기준으로 활성화 되거나 혹은 비활성화 되는 step function 또는 Heaviside function을 근사화 한 것이다.

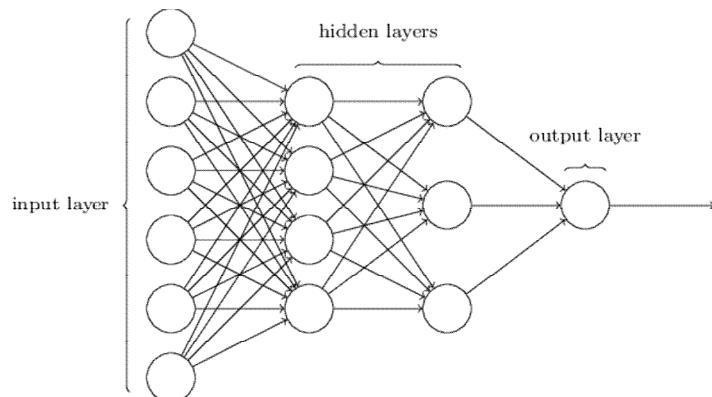
$$H(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$



- sigmoid 함수는 또한 다음과 같은 좋은 성질을 갖는다.

$$f'(x) = -(1+e^{-x})^{-2}(-e^{-x}) = \frac{e^{-x}}{(1+e^{-x})^2} = \frac{1}{1+e^{-x}} \cdot \frac{e^{-x}}{1+e^{-x}} = f(x)(1-f(x))$$

- 신경망은 순환하지 않는 그래프로 연결된 뉴런의 모음을 모델화 한 것으로 다음은 신경망을 그림으로 표현한 것이다. 왼쪽부터 입력층(Input layer), 은닉층(Hidden layers), 출력층(Output layer)으로 구성되어 있다. 따라서 입력층에서 신호를 받으면, 미리 부여된 가중치와 계산 후 그 총합이 은닉층으로 전파되고, 주어진 활성화 함수에 따라 그 다음 층으로 전해질 신호가 계산된다. 이런 식으로 전파되어 출력층에서 해당하는 결과를 내보낸다.



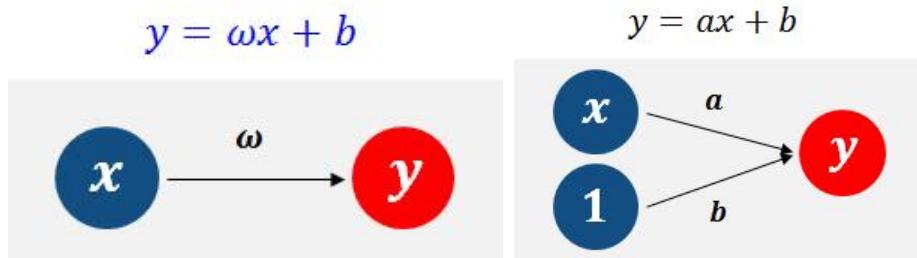
[그림 출처] <http://neuralnetworksanddeeplearning.com/chap1.html>

- 활성화 함수가 일차함수인 간단한 경우를 통해 신경망의 원리를 살펴보도록 하자. 아래와 같이 하나의 노드는 입력 신호를 받아 결과를 전달해주는 하나의 합수로 볼 수 있다.

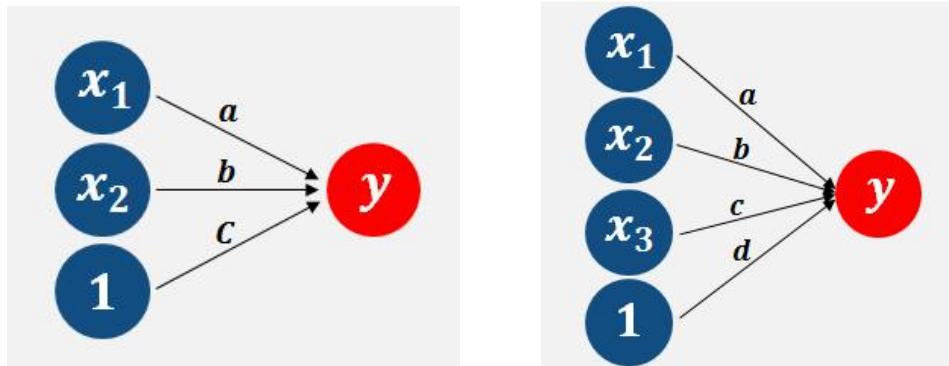
따라서 입력 x 를 받아 y 를 출력하는 경우 $y = xw + b$ 와 같이 나타낼 수 있다. (여기서 b 는 편향(bias)을 의미한다. 예를 들어, 입력층에서 신호 x 를 받으면, 미리 부여된 가중치 w 와 계산 후 그 결과값이 임계값 θ 를 넘으면 1을 출력, 즉

$$wx > \theta \Rightarrow 1을 출력$$

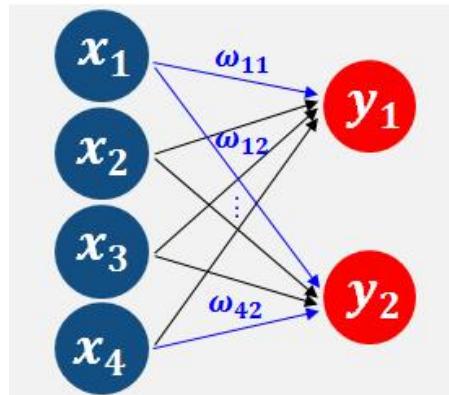
할 때 θ 를 왼쪽으로 이항하면 $b = -\theta$ 를 얻는다.)



마찬가지로 입력이 두 개라면, $y = x_1 w_1 + x_2 w_2 + b$, 입력이 세 개라면 $y = x_1 w_1 + x_2 w_2 + x_3 w_3 + b$ 와 같이 표현할 수 있을 것이다.

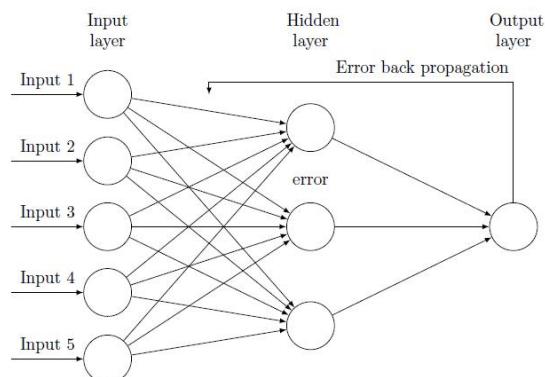


- 만일 출력이 두 개라면, 다음과 같이 Part 1에서 학습한 행렬의 곱을 이용하여 나타낼 수 있다. 이때, 편의상 편향은 0이라고 하고, w_{ij} 는 입력층의 i 번째 노드에서 출력층의 j 번째 노드로 연결된 가중치를 의미한다.



$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} w_{11} & w_{21} & w_{31} & w_{41} \\ w_{12} & w_{22} & w_{32} & w_{42} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \quad \text{또는} \quad \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix} \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \\ w_{41} & w_{42} \end{bmatrix} = \begin{bmatrix} y_1 & y_2 \end{bmatrix}$$

- 입력값과 정답을 알고 있는 데이터셋(dataset)이 있다고 하자. 우리의 목적은 주어진 데이터셋을 잘 판단하도록 신경망의 가중치를 찾는 것이다. 이때 어떤 공식에 의해서 가중치를 계산하는 것이 아니라, 초기에 임의로 가중치를 부여해놓고, 주어진 데이터로부터 신경망을 이용하여 얻은 예측값과 미리 알고 있는 정답간의 오차를 줄이는 방향으로 가중치를 점차 갱신해 나간다. 이때 계층 간의 각각의 연결이 오차에 영향을 주는 정도에 비례해서 오차를 전달해준다. 대표적인 방법으로 오차 역전파(back propagation)법이 있다.



[그림 출처] <https://sacko.tistory.com/19?category=632408>

- 오차 역전파법에 의해 각 계층에 전달된 오차를 바탕으로 가중치를 갱신하는 방법은 Part 2에서 이미 학습한 경사하강법(gradient descent method)을 오차함수를 최소화하는 문제에 적용한 것과 같다. 예를 들어, 각 계층에서 입력신호 \mathbf{x}_i ($i = 1, \dots, N$)를 받아 미리 부여된 가중치와 계산 후 주어진 활성화 함수로부터 출력된 값을 $\hat{\mathbf{y}}_i$ 라 하고, 실제 정답은 \mathbf{y}_i 라 하자. 그러면 제곱 오차(squared error)는 다음과 같이 주어진다.

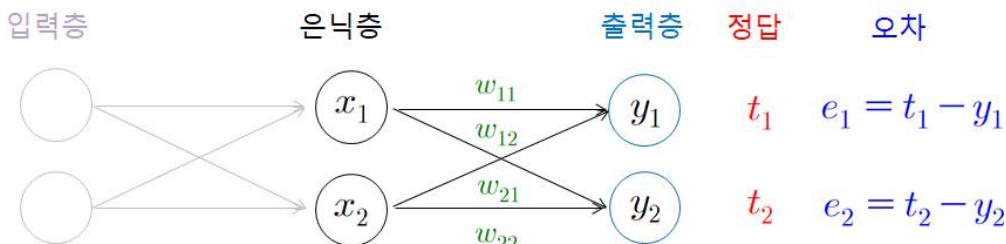
$$E = \frac{1}{2} \sum_{i=1}^N \| \mathbf{y}_i - \hat{\mathbf{y}}_i \|^2$$

그러면 각 가중치를 갱신하는 공식은 경사하강법으로부터 다음과 같이 주어진다.

$$w_{ij}^{(\text{NEW})} = w_{ij}^{(\text{OLD})} - \alpha \frac{\partial E}{\partial w_{ij}}.$$

- 이제 아래와 같이 간단한 신경망의 각 계층에 전달된 오차로부터 가중치를 갱신하는 방법에 관하여 살펴보자.

- ① 각 계층에 전달된 오차를 계산한다. 먼저 출력층에서의 오차를 계산하기 위하여, 은닉층에서 입력 신호 x_1 과 x_2 를 받아 가중치 $w_{11}, w_{12}, w_{21}, w_{22}$ 와 계산한 후, sigmoid 함수 $f(x)$ 를 거쳐 출력 y_1 과 y_2 를 얻었다고 가정하자.



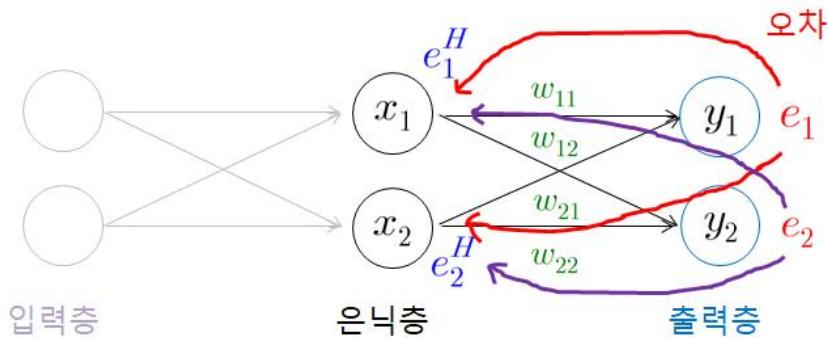
$$\text{제곱오차 } E = \frac{1}{2} (e_1^2 + e_2^2) = \frac{1}{2} ((t_1 - y_1)^2 + (t_2 - y_2)^2)$$

그러면 출력층에서의 제곱오차는 다음과 같다.

$$\begin{aligned}
E &= \frac{1}{2}(e_1^2 + e_2^2) = \frac{1}{2}((t_1 - y_1)^2 + (t_2 - y_2)^2) \\
&= \frac{1}{2}((t_1 - f(x_1 w_{11} + x_2 w_{21}))^2 + (t_2 - f(x_1 w_{12} + x_2 w_{22}))^2)
\end{aligned}$$

이제 은닉층에서의 오차를 계산해보자. 은닉층에는 출력층과 달리 목표하고자 하는 정답이 따로 없으므로 출력층에서의 오차 $e_j = t_j - y_j$ 에 해당하는 것이 없다. 이때 앞서 언급한 오차 역전파법을 활용한다.

신경망에서 오차가 일어났다는 것은 결국, 입력신호가 입력층으로부터 은닉층을 거쳐 최종 출력층으로 전파될 때, 은닉층에서의 오차가 반영된 결과라고 볼 수 있으므로 계층 간의 각각의 연결이 오차에 영향을 주는 정도, 즉 가중치에 비례해서 오차를 역으로 전달해주면 된다.



출력층의 첫 번째 노드에서 오차 e_1 을 얻었다고 가정하자. 이 노드는 은닉층의 첫 번째 노드와 가중치 w_{11} , 두 번째 노드와 가중치 w_{21} 으로 연결되어 있다. 가중치가 크면 오차에 미치는 영향도 클 것이므로, 가중치에 비례하여 오차를 은닉층에 전달해준다. 따라서 예를 들어, e_1^H 를 은닉층의 첫 번째 노드에 전파된 오차라 하면 다음과 같이 계산할 수 있다.

$$e_1^H = \frac{w_{11}}{w_{11} + w_{21}} e_1 + \frac{w_{21}}{w_{12} + w_{22}} e_2$$

마찬가지로 e_2^H 를 구할 수 있다.

$$e_2^H = \frac{w_{21}}{w_{11} + w_{21}} e_1 + \frac{w_{22}}{w_{12} + w_{22}} e_2$$

위의 두 식에서 분모를 제외하면 다음과 같이 행렬을 이용하여 나타낼 수 있다.

$$\begin{bmatrix} e_1^H \\ e_2^H \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \end{bmatrix}$$

이때, 분모를 없애게 되면 원 식과 비교했을 때, 일정 부분의 비율이 사라지게 되는데, 다음 웹사이트에 따르면, 분모가 있는 경우와 실제로는 거의 차이가 없이 잘 작동하는 것이 알려져 있다.

[참고] <http://makeyourownneuralnetwork.blogspot.com/2016/07/error-backpropagation-revisted.html>

② 출력층에서 얻은 오차로부터 은닉층과 출력층 사이의 가중치를 갱신한다.

sigmoid 함수의 성질과 연쇄법칙으로부터 $\frac{\partial E}{\partial w_{11}}$ 을 구하면 다음과 같다.

$$\begin{aligned} \frac{\partial E}{\partial w_{11}} &= - (t_1 - f(x_1 w_{11} + x_2 w_{21})) f'(x_1 w_{11} + x_2 w_{22}) (1 - f(x_1 w_{11} + x_2 w_{21})) x_1 \\ &= -(t_1 - y_1) y_1 (1 - y_1) x_1 \end{aligned}$$

다른 가중치에 대해서도 마찬가지로 다음과 같이 계산한다.

$$\frac{\partial E}{\partial w_{12}} = - (t_2 - y_2) y_2 (1 - y_2) x_1$$

$$\frac{\partial E}{\partial w_{21}} = - (t_1 - y_1) y_1 (1 - y_1) x_2$$

$$\frac{\partial E}{\partial w_{22}} = - (t_2 - y_2) y_2 (1 - y_2) x_2$$

즉, 가중치 w_{ij} 는 은닉층의 i 번째 노드와 출력층의 j 번째 노드만 연결되어 있으므로, E 를 w_{ij} 에 대하여 편미분하면 다음과 같이 간단하게 표현됨을 알 수 있다.

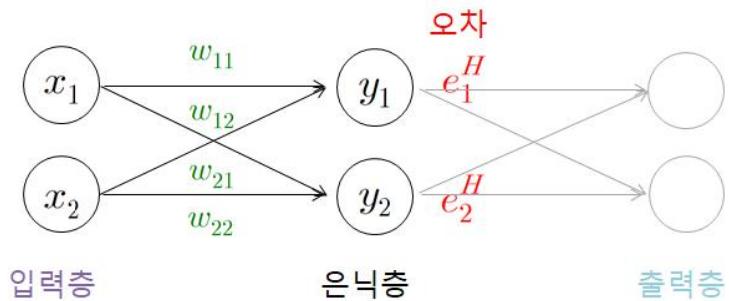
$$\frac{\partial E}{\partial w_{ij}} = -(t_j - y_j)y_j(1 - y_j)x_i = -e_j y_j(1 - y_j)x_i$$

여기서 x_i 는 은닉층의 i 번째 노드에서의 입력, y_j 는 출력층의 j 번째 노드에서의 출력, $e_j = t_j - y_j$ 는 출력층의 j 번째 노드에서의 오차이다.

이제 경사하강법을 적용하여 다음과 같이 은닉층과 출력층 사이의 모든 가중치를 갱신한다.

$$w_{ij}^{(\text{NEW})} = w_{ij}^{(\text{OLD})} - \alpha \frac{\partial E}{\partial w_{ij}}$$

③ 은닉층에 전달된 오차로부터 입력층과 은닉층 사이의 가중치를 갱신한다. 기본적으로는 은닉층과 출력층 사이의 가중치를 갱신하는 방식으로 계산하면 된다. 논의의 편의상 기호를 그대로 사용하였다.



여기서 x_i 는 입력층의 i 번째 노드에서의 입력, y_j 는 은닉층의 j 번째 노드에서의 출력, e_j^H 는 은닉층의 j 번째 노드에 전파된 오차이다. 따라서 다음을 얻을 수 있다.

$$\frac{\partial E}{\partial w_{ij}} = -e_j^H y_j(1 - y_j)x_i$$

이제 경사하강법을 적용하여 다음과 같이 입력층과 은닉층 사이의 모든 가중치를 갱신할 수 있다.

$$w_{ij}^{(\text{NEW})} = w_{ij}^{(\text{OLD})} - \alpha \frac{\partial E}{\partial w_{ij}}$$

- 신경망은 제대로 예측할 때 까지 많은 데이터를 필요로 하며, 또한 입력층과 출력층 사이에 많은 은닉층을 둘 수도 있다. 이와 같이 은닉층이 여러 개 있는 인공신경망을 심층신경망(deep neural network)이라고 부르며, 심층 신경망을 학습하기 위한 기계학습 기법을 딥러닝(deep learning)이라고 부른다. 은닉층이 여러 개 있는 경우에도 마찬가지로 오차 역전파법을 이용하여 그 이전 층에서 전파된 오차로부터 경사하강법을 적용하여 가중치를 갱신할 수 있다. 기타 자세한 사항은 신경망, 딥러닝과 관련된 도서를 참조하라.

1. SKKU 선형대수학 Lectures

<http://matrix.skku.ac.kr/LA-K/> <http://matrix.skku.ac.kr/LA/>

2. SKKU 미적분학 Lectures

<http://matrix.skku.ac.kr/Cal-Book1/>

<http://matrix.skku.ac.kr/Cal-Book/>

Part I [Single Variable Calculus](#) Part II [Multivariate Calculus](#)

3. 기초 통계학

R을 활용한 기초 통계학 실습실 Lab 1

<http://matrix.skku.ac.kr/2018-album/R-Sage-Stat-Lab-1.html>

R을 활용한 기초 통계학 실습실 Lab 2

<http://matrix.skku.ac.kr/2018-album/R-Sage-Stat-Lab-2.html>

4. Mathematics for BigData

<http://matrix.skku.ac.kr/e-math/>

5. SKKU Discrete Mathematics(이산수학) Lectures

<http://matrix.skku.ac.kr/2018-DM/DM-Labs.htm>

3. 학생 프로젝트 사례

28강 동영상, 빅데이터와 인공지능, Project 발표 <https://youtu.be/EKkr3EkDV3M> (21:06)

아래는 2016년도 1학기 개설된 ‘빅데이터를 위한 수학’ (담당: 이상구 교수)에서 수행된 학생 프로젝트 사례이다.

제목 : Statistical analysis on features of actual energy consumption
of office buildings in the South Korea

프로젝트 수행 학생 : 안기연, 추한경, 지도 : 이상구 교수

*본 내용은 보강 연구를 거쳐 Big-data Analysis on Energy Consumption of Office Buildings in Seoul, Korea라는 제목으로 저자(Ki Uhn Ahn, Han Sol Shin, Cheol Soo Park, Kwang Woo Kim)에 의해 아래 학회

Proceedings of the 15th IBPSA Conference, San Francisco, CA, USA, Aug. 7–9, 2017
http://www.ibpsa.org/proceedings/BS2017/BS2017_411.pdf

에 발표되었고, 그 후 다음과 같은 논문으로 게재되었다. 본 프로젝트 사례는 대표저자인 안기연 교수의 동의를 아래와 같이 얻어 사용한다.

Ki Uhn Ahn, Han Sol Shin and Cheol Soo Park, Energy Analysis of 4625 Office Buildings in South Korea, *Energies* 2019, 12(6), 1114; <https://doi.org/10.3390/en12061114>

[안녕하세요 이상구 교수님,

항상 교수님께 감사한 마음을 갖고 있으며 이를 꼭 전해 드리고 싶었 ... 습니다.

교수님께 수강한 PBL 수업 덕분에,

<다른 강좌에서 배울 수 없었던, 기계학습 및 인공지능에 활용되는 수학의 지식을 얻고 이를 해하는데 큰 도움이 되었으며, Python-Sage, R 등의 다양한 프로그래밍 언어를 실습하여, 이를 연구에 활용할 수 있는 소중한 계기가 되었고, 수업에서 진행하였던 프로젝트를 통해 제 전공분야의 학술적 의의를 도출하고, 이를 연구 결과로 발표할 수 있었습니다.> 많은

지식과 경험을 얻게 가르쳐 주시고, 교수님 수업에서 소개되는 큰 영광을 주셔서 정말 감사합니다.

교수님의 수업을 통하여 성균관대학교 후배들 또한 학문적 지적 호기심이 고취되고, 큰 배움과 발전이 있을 것이라 생각합니다. 건강하시고 항상 좋은 일 가득하시길 바랍니다.

안기언 드립. (2019/09/04 (수) 12:50)]

3.0 역할 분담

○ 안기언

- 데이터마이닝 분석 수행 위한 데이터 가공, - 자료의 시각화 및 분석, - 통계기반 데이터마이닝, - 보고서 작성

○ 추한경

- 건물정보 데이터베이스 수집, - 데이터 필터링

3.1 Outline of4 project

○ 목적 : 국내 건축물 에너지 소비 현황 및 특징을 분석하고, 에너지 사용량에 영향을 미치는 요소를 확인하여, 향후 정책 및 에너지 절감 기술 개발 방향 논의

○ 배경 : 전 세계적으로, 건축물에서 소비되는 에너지를 절감하기 위해, 에너지 절감 정책을 제정하고, 신기술 개발에 주력하고 있음. 하지만, engineering based approach(e.g. building energy simulation analysis) 기반의 건물 에너지 성능 분석 및 예측은 건축물의 실제 에너지 소비와 상당한 차이를 보임. 따라서, 전술한 노력들이 실효성을 지니기 위해서는, 기존 건축물의 현황 및 에너지 소비 패턴에 대한 분석이 선행되어야함.

○ 비고 : 본 수업시간을 통해 접하게 된 R을 실제 데이터 마이닝 프로젝트에 활용하면서, R을 학습하고 익히기 위한 개인적인 목표가 있었음. 따라서 본 프로젝트는 데이터의 편집을 제외하고 모든 과정을 R을 통해서 진행함

3.2 Data collection and matching

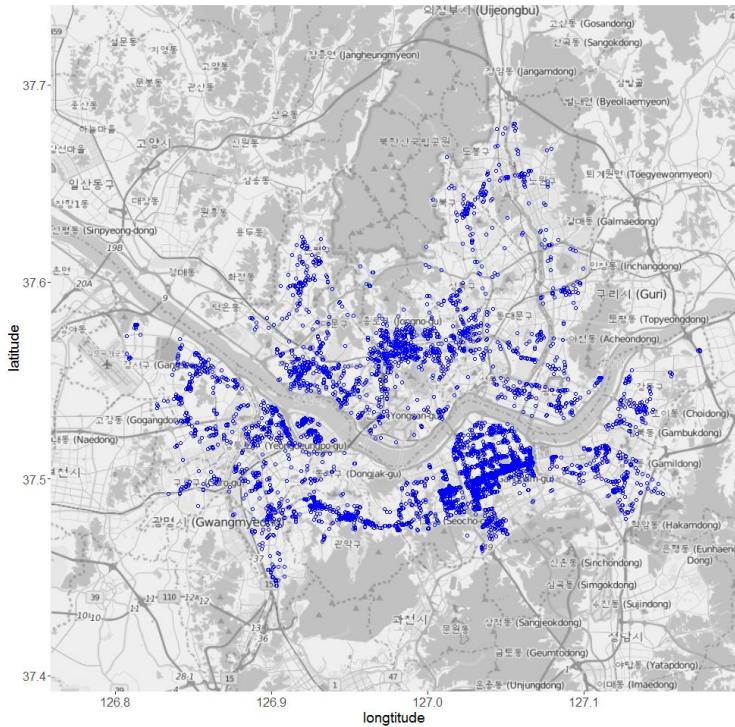
- 개념 : 데이터마이닝을 통해 자신이 원하는 정보를 획득하기 위해서는, 원본 데이터를 자신의 목적 및 데이터마이닝 방법에 맞춰 변형시키는 작업이 필요함 (Data generalization or specialization).
- 개요 : 최근 정부는 건축물 특징에 대한 정보가 담긴 데이터베이스와 건축물에서 소비되고 있는 에너지 사용량 데이터베이스를 각각 공개하였으며, 본 프로젝트는 두 데이터베이스를 수집하고, 각 데이터베이스가 포함하고 있는 건축물 주소를 기준으로 하나의 데이터베이스로 통합하는 작업을 수행함.
- 데이터베이스 목록: 정부 3.0에 따라 공개된 건축물 데이터베이스 2가지 활용
 - 건물 특징 정보 데이터베이스: <http://open.eais.go.kr/>
 - 데이터 목록: 주소, 연면적, 용적율, 주용도, 기타용도, 세대 수, 층 수, 준공년도
 - 건물 에너지 사용량 데이터베이스: <https://open.eais.go.kr>
 - 데이터 목록: 주소, 전기/가스 월별 에너지 사용량

3.3 Description of database(DB)

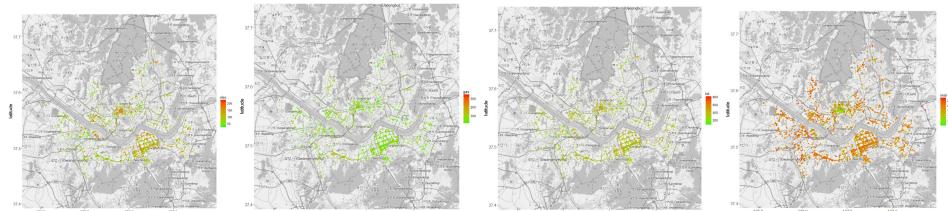
- 대상 : 서울시 업무용 건축물, 총 4,603동
- 데이터 목록
 - 주소 : 구(서울시 소재 25개 구), 동, 지번 [문자 형태]
 - 층수 : 지상 층수, 지하 층수 [숫자 형태]
 - 연면적 [숫자형태]
 - 엘리베이터 대수 [숫자 형태]
 - 용도 : 주용도, 부용도 [문자형태]
 - 준공년도 [숫자형태]
 - 에너지 사용량: 월별 전기, 가스 에너지 사용량 [숫자 형태]

○ DB의 건축물 위치 가시화

- 방법: 데이터 목록이 포함하고 있는 주소를 기준으로, 위도 및 경도 정보를 획득하고, 이를 통해 지도에 위치를 표현함



3.4 Visualization of energy consumption



<단위 면적당 전기 에너지 사용량 기준 건축물 분포 현황>

<단위 면적당 가스 에너지 사용량 기준 건축물 분포 현황>

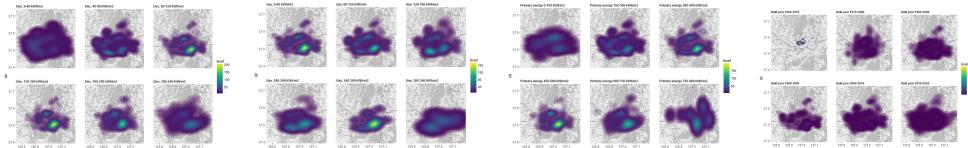
<단위 면적당 일자 에너지 사용량 기준 건축물 분포 현황>

<준공년도 기준 건축물 분포 현황>

3.5 Statistics of database

<지역구별 월별 일차 에너지 사용 현황>

3.6 Estimation of energy consumption



<단위 면적당 전기 에너지 사용량 기준 건축물 분포 추정>

<단위 면적당 가스 에너지 사용량 기준 건축물 분포 추정>

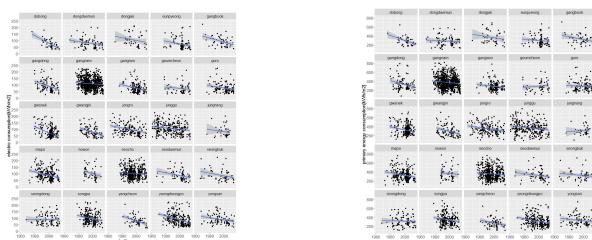
<단위 면적당 일차 에너지 사용량 기준 건축물 분포 추정>

<준공년도 기준 건축물 분포 추정>

○ 결과

- 전기 에너지 사용량이 높은 건축물은 강남구 및 서초구에 주로 밀집해 있으며, 중구 및 종로구도 다른 지역구에 비해 전기 에너지 사용량이 높음 ...
- 노후화된 건축물은 종로구에 밀집해 있으며, 그 외의 건축물은 전 지역구에 걸쳐 균등한 수준으로 분포하고 있음

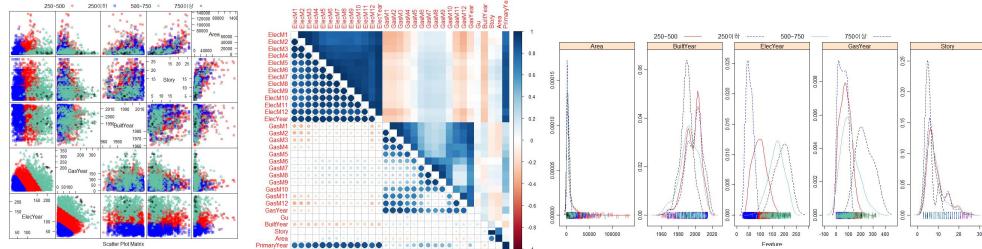
3.7 Correlation between energy consumption and building year



<전기 에너지 사용량과 준공년도의 scatter plot>

<일차 에너지 사용량과 준공년도의 scatter plot>

3.8 Detailed analysis on correlation between energy consumption and building features



<건축물 특징 사이의 scatter plot matrix 분석 결과>

<건축물 특징 사이의 correlation plot> <건축물 특징 별 에너지 사용량 확률 밀도>

3.9 Estimation of energy basis and building operation hour

3.10 Conclusion

- 노후화된 건축물의 에너지 성능 개선을 통해, 에너지 사용량을 절감할 수 있다는 직접적인 근거를 확인하기 어려움(노후 건축물의 에너지 사용량이 적기도 하며, 신식 건축물의 에너지 사용량이 많기도 함)
- 건물의 에너지 사용량에 결정적 영향을 미치는 요인은 건물의 특징 및 속성보다도 건물의 에너지 사용시간이며, 이로 인해 건축물의 에너지 성능과 사용량 사이의 차이가 발생함(performance gap)
- 건물의 성능이 에너지 사용량에 영향을 미칠 수 있지만, 건물 설비 시스템의 효율적인 운영이 에너지 사용량을 절감시킬 수 있는 결정적인 요인이 될 것으로 추정됨

[참고 문헌]

건물에너지 정보공개시스템, <http://open.eais.go.kr/> (accessed by 2016.06.13.)

건축데이터 민간개방 시스템, <https://open.eais.go.kr> (accessed by 2016.06.13.)

3.11 Final comments

○ 안기언

- 3월 기계학습 관련 수학적 배경지식의 학습과정에서는 어려움이 있었고, 처음 경험해 보는 PBL 수업이기에, 학기 초에는 본 수업의 프로세스가 다소 낯설고 적응하기 힘들기도 했습니다. 하지만, 다른 학생들이 수업에 임하는 태도와 성의, 교수님의 적극적인 수업 환경조성이 본 수업 방식에 익숙해지는데 많은 도움이 되었습니다.
- 본 수업을 통해, 한 학기동안 기계학습 관련 수학적 배경지식을 습득하고, 관련 예제를 직접 다뤘으며, 주제를 선정하여 프로젝트를 진행하였습니다. 수업을 수강하기 전과 후의 차이라면, 기계학습 관련 문헌을 읽더라도 수학적 알고리즘 설명에 거부감이 확연히 줄어들었으며, 수업시간에 학습한 개념들이 이론을 이해하는데 많은 도움이 되고 있습니다. 더불어, 프로젝트의 성과는 논문으로 발전시킬 계획입니다. 특히, 프로젝트에서 R 프로그램을 이용하였으며, 본 수업을 통해 R 프로그램을 학습하는 계기를 마련해주셔서 너무나 감사하게 생각하고 있습니다.
- 마지막으로, 본 기계학습 PBL 수업은 지식의 습득 및 활용에 최적화된 수업이라고 생각합니다. 이와 같은 환경을 조성해주신 교수님과 수업에 적극적으로 참여하여 많은 도움을 준 학생들에게 감사의 인사를 전합니다.

[참고 문헌]

* 프로젝트의 구체적인 내용 전체를 보려면 아래 주소를 참고하시기 바랍니다.

<http://matrix.skku.ac.kr/math4ai/part4/>

http://www.ibpsa.org/proceedings/BS2017/BS2017_411.pdf

빅 데이터 분석 기법을 활용한 기존 건축물 데이터베이스 분석, Big-data Analysis on Energy Consumption of Existing Buildings in South Korea, 안기언(Ahn, Ki-Uhn), 박철수 (대한건축학회 학술 발표대회 논문집, Vol.36 No.2, [2016])



부록: [심화내용인 Math for AI (2nd Course) 강의 동영상 주소]

<http://matrix.skku.ac.kr/e-math/>

Math for Big Data, Lecture 1, Introduction, <https://youtu.be/EURJnLppzKc>

Math for Big Data, Lecture 2, LU Decomposition, <https://youtu.be/bzhTnoN3atk>

Math for Big Data, Lecture 3, Schur Decomposition, https://youtu.be/F2kZON0oS_w

Math for Big Data, Lecture 4, Power Method, https://youtu.be/n4KD4aq_jxw

Math for Big Data, Lecture 5, QR Decomposition, <https://youtu.be/gQ7gxTx5f9k>

Math for Big Data, Lecture 6, Google's PageRank algorithm, <https://youtu.be/tp6B7s43jAI>

Math for Big Data, Lecture 7, SVD, <https://youtu.be/AxL4Q83IdAA>

Math for Big Data, Lecture 8, Least Square Solutions, <https://youtu.be/GwHh5lh5wEs>

Math for Big Data, Lecture 9, Polar Decomposition, NMF, <https://youtu.be/FqkMP9lBtaE>

Math for Big Data, Lecture 10, Finding JCF using Dot Diagram, <https://youtu.be/1E3wXN1oZyc>

Math for Big Data, Lecture 11, 일반화된 고유벡터와 행렬함수, https://youtu.be/lK4_Kp6P_N4

Math for Big Data, Lecture 12, Principal Componant Analysis 1, <https://youtu.be/0IKbslNH7xk>

Math for Big Data, Lecture 13, Principal Componant Analysis 2, https://youtu.be/j8PAt_Ai180

Math for Big Data, Review 1, Intro. Calculus, Team 4, <https://youtu.be/qALN6OAwNUo>

Math for Big Data, Review 2, Intro. Linear Algebra, Team 3, <https://youtu.be/xrFqBe8Rhs4>

Math for Big Data, Review 3, Intro. Statistics, Team 2, <https://youtu.be/sOx74EntB0I>

Math for Big Data, Review 4, Intro. Engineering Math, <https://youtu.be/LRHN5swQW4E>

Math for Big Data, Midterm PBL, S. Sun, <https://youtu.be/CSdciSMPm-8>

Math for Big Data, Midterm PBL, Naguib, https://youtu.be/k9_Ie8bMAY0

Math for Big Data, Midterm PBL, KEAhn, https://youtu.be/xFJmI1_uynk

Math for Big Data, Midterm PBL, Choo, https://youtu.be/TIC78z_LErQ

Math for Big Data, Midterm PBL, Naeem, <https://youtu.be/8xo5UOP1tu8>

Math for Big Data, Midterm PBL, Lkhagva, <https://youtu.be/pPtO1rNdLs0>

Math for Big Data, Midterm PBL, Sudip, https://youtu.be/5md49_RG74Q

Math for Big Data, Midterm PBL, Jeongwon Pyo, <https://youtu.be/u5zDWtmx9P0>

Math for Big Data, Midterm PBL, ESJang, <https://youtu.be/cHYvWBuBrFA>

Math for Big Data, Lecture 14, Graph and Matrix, <https://youtu.be/Z89XvKXIYeg>

Math for Big Data, Lecture 15, Laplacian Matrix and Big Data,

<https://youtu.be/4VuaOFRGm1g>

Math for Big Data, Lecture 16, Intro. Math for Machine Learning 1,

<https://youtu.be/P24A1fkpX-Y>

Math for Big Data, Lecture 17, Intro. Math for Machine Learning 2,

<https://youtu.be/bY3nfAHc6Qk>

Math for Big Data, Lecture 18, (Team 4) Intro. Data Mining, Ahn& Choo,

<https://youtu.be/Dq2G8ReeEcY>

Math for Big Data, Lecture 19, (Team 1) Pattern Classification 1, Naguib &Naeem,

<https://youtu.be/ieOUI6pc18A>

Math for Big Data, Lecture 20, (Team 1) Pattern Classification 2, Naguib &Naeem,

<https://youtu.be/9kxyu0e-nfQ>

Math for Big Data, Lecture 21, (Team 2) Statistical Learning,

<https://youtu.be/5dQO2Z3PgPU>

Math for Big Data, Lecture 22, (Team 3) Cluster Analysis, <https://youtu.be/LPyFO8jFHD8>

Math for Big Data, Lecture 23, (Team 3) Project Draft 1, <https://youtu.be/TZJrU7S1Q0o>

Math for Big Data, Lecture 24, (Team 3) Project Presentation, Spectral Cluster Analysis by Shaowei-Lkhagva, <https://youtu.be/476HgeBM8AE>

Math for Big Data, Lecture 25, (AV) Project Presentation, Restricted Boltzmann Machine

Training of Perceptron for Clustering by Naguib-Naeem <https://youtu.be/QLKIgUCVLIY>

Math for Big Data, Lecture 25, (Team 1) Project Presentation, Restricted Boltzmann

Machine Training of Perceptron for Clustering by Naguib-Naeem <https://youtu.be/vZ613MEWin4>

Math for Big Data, Lecture 26, (Team 2) Project Presentation, Hand Gesture Recognition

with Convolutional Neural Network by Pyo-Sudip-Jang <https://youtu.be/FK-ANqohVlo>

Math for Big Data, Lecture 27, Final PBL Presentation by Sudip, <https://youtu.be/cOwWZcVb1AU>



<인공지능을 위한 기초수학> (Basic Mathematics for Artificial Intelligence)

<http://matrix.skku.ac.kr/math4ai/>

인공지능을 위한 기초수학 1강 (강의운영 소개), Basic Math for AI 1 <https://youtu.be/lkQxelizELM>

인공지능을 위한 기초수학 2강 (선형대수학 1: Big Picture, 벡터, 정사영, 최단거리), Math 4 AI 2 <https://youtu.be/UdCJClk2MWDU>
인공지능을 위한 기초수학 3강 (선형대수학 2: 선형연립방정식, 행렬과 행렬식), Basic Math for AI 3 https://youtu.be/qwQX_zPiICU

인공지능을 위한 기초수학 4강 (선형대수학 3: 기저, 차원), Basic Math for AI 4 <https://youtu.be/UHqhrN38ps>

인공지능을 위한 기초수학 5강 (선형대수학 4: 최소제곱해, QR분해), Basic Math for AI 5 <https://youtu.be/5r2KghYFw2w>

인공지능을 위한 기초수학 6강 (선형대수학 5: 선형변환), Basic Math for AI 6 https://youtu.be/_t871V2CDSw

인공지능을 위한 기초수학 7강 (선형대수학 6: 행렬의 대각화), Basic Math for AI 7 <https://youtu.be/d8KE1QpKiDo>

인공지능을 위한 기초수학 8강 (선형대수학 7: SVD), Basic Math for AI 8 <https://youtu.be/e0loDqjLB8U>

인공지능을 위한 기초수학 9강 (선형대수학 8: 이차형식), Basic Math for AI 9 <https://youtu.be/rCNBWT0r5mA>

인공지능을 위한 기초수학 10강 (미적분 1: 극한과 도함수), Basic Math for AI 10 <https://youtu.be/rsItppMbtBQ> (New)

인공지능을 위한 기초수학 11강 (미적분 2: 미분의 응용), Basic Math for AI 11 <https://youtu.be/O4IN5zEZnMA>

인공지능을 위한 기초수학 12강 (미적분 3: 적분), Basic Math for AI 12 <https://youtu.be/62OxYG7VMsE>

인공지능을 위한 기초수학 13강 (미적분학 4: 다변수함수), Basic Math for AI 13 다변수함수와 미적분 https://youtu.be/XQW8_8k9GjE

인공지능을 위한 기초수학 14강 (미적분학 5: 편도함수와 그래디언트), Basic Math for AI 14 <https://youtu.be/cvsBYZT4SZg>

인공지능을 위한 기초수학 15강 (미적분학 6: 함수의 극대, 극소), Basic Math for AI 15 <https://youtu.be/nR9it9cBjDk>

인공지능을 위한 기초수학 16강 (미적분 7: Gradient Descent Algorithm), Math for AI 16 경사하강법 <https://youtu.be/XWDPAAdKhq-Q>

인공지능을 위한 기초수학 17강 (미적분 8: 중적분), Basic Math for AI 17 https://youtu.be/T1z_GYt85rl

인공지능을 위한 기초수학 18강 (통계 1: 통계학과 R), Basic Math for AI 18 <https://youtu.be/u82BC1Rj0JA>

인공지능을 위한 기초수학 19강 (통계 2: 순열, 조합, 확률), Basic Math for AI 19 <https://youtu.be/KQXO-XbjauU>

인공지능을 위한 기초수학 20강 (통계 3: 확률변수), Basic Math for AI 20 <https://youtu.be/SUsZHarQqqg>

인공지능을 위한 기초수학 21강 (통계 4: 이산확률분포), Basic Math for AI 21 https://youtu.be/Fq7D7bGG_cE

인공지능을 위한 기초수학 22강 (통계 5: 연속확률분포), Basic Math for AI 22 <https://youtu.be/4wx1raETI8o>

인공지능을 위한 기초수학 23강 (통계 6: 공분산과 상관계수, 데이터 활용의 실제), Math for AI 23 <https://youtu.be/oUSPhkyEWp4> (50:05)

인공지능을 위한 기초수학 24강 (빅데이터와 인공지능 1. 개론), Basic Math for AI 24 https://youtu.be/MpWv-U_4fI0

인공지능을 위한 기초수학 25강 (빅데이터와 인공지능 2. PCA 주성분분석), Basic Math for AI 25 https://youtu.be/NVd6_Orz0ng (56:21)

인공지능을 위한 기초수학 26강 (빅데이터와 인공지능 3. ANN 신경망 (Math of Hidden Layer)), https://youtu.be/d4WercT_OnU (18:14)

인공지능을 위한 기초수학 27강 (빅데이터와 인공지능 4. MNIST), Basic Math for AI 27 <https://youtu.be/UqmV4wEzK1Y>

인공지능을 위한 기초수학 28강 (빅데이터와 인공지능 5. Project), Basic Math for AI 28 <https://youtu.be/EKkr3EkDV3M>

인공지능을 위한 기초수학, Midterm PBL 학생 발표 1, 최진호 <https://youtu.be/0iBPdFEXzqM>

인공지능을 위한 기초수학 29강 1학년 염지민 Final PBL 학생 발표 <https://youtu.be/kldV0YcuYu8> (13:39)

인공지능을 위한 기초수학 29강 화공과 최진호 Final PBL 학생 발표 <https://youtu.be/zj4PTgnWyac> (15:35)

인공지능을 위한 기초수학 29강 최준오 Final PBL 학생 발표 <https://youtu.be/kWrgij3oDE> (9:39)

인공지능을 위한 기초수학 29강 황태식 Final PBL 학생 발표 <https://youtu.be/JSbcOaupkEo> (10:07)

인공지능을 위한 기초수학 29강 1팀 최진호 Final Project <http://matrix.skku.ac.kr/math4ai/Project-1/>

인공지능을 위한 기초수학 29강 2팀 Final PBL Project <http://matrix.skku.ac.kr/math4ai/Project-2/>

[창의재단 Talk] 수학교육과 인공지능(AI) by 이상구 <https://youtu.be/-emrYxyde-M>

[한양대 Talk] Math for Big Data / Machine Learning / AL <https://youtu.be/yZlloZvYIOo>



읽을거리1

“선형대수학과 구글(Google) 검색엔진”

– 페이지랭크 알고리즘

<https://www.scienceall.com/미래를-여는-수학-⑦-Google-검색의-비밀은/>

글 | 이상구 성균관대 수학과 교수

영국 파이낸셜타임스(FT)와 컨설팅그룹 밀워드브라운이 공동 조사한 ‘2010 글로벌 100대 브랜드 기업’을 보면 랭킹 1위는 미국의 인터넷 기업 구글로, 브랜드 가치가 1,143억 달러를 기록해 2007년부터 4년 연속 최고의 자리를 지켰다.¹⁾ 최근 구글의 모토로라 인수가 삼성과 한국의 경제에 큰 영향을 줄 것이라는 언론 보도를 접하였듯이 격변하는 환경에서 ‘모든 학생은 왜 수학을 배워야 하는가?’에 대한 답의 하나로 본 원고에서는 경제와 생활에 막대한 영향을 미치는 “구글 검색 엔진 속의 수학이론”에 대하여 소개한다.

구글(Google)의 역사는 Matrix Computation의 저자 Gene Golub이 강의하던 스탠퍼드대의 20대 초반의 대학원생 래리 페이지(Larry Page)와 세르게이 브린(Sergey Brin)이 1995년 기준의 검색엔진들이 보여주는 정리되지 않는 결과물에 불만을 가지고 새로운 검색기법 연구에 몰두하여 마침내 페이지랭크(PageRank) 기술과 링크(link)의 매칭기술을 개발하는 데 성공하면서 시작된다. 이들은 ‘이 사이트가 링크를 만들어 연결시켜 줄 얼마만큼의 가치를 가지고 있을까?’라는 기준을 가지고 검색된 사이트들의 순위를 매기는 구글 특유의 차별화된 검색방법을 확립한다.

페이지와 브린이 인터넷의 광대한 정보를 구글이 모두 담겠다는 의지를 담고, 미국인 수학자 Edward Kasner의 조카인 Milton Sirota가 10의 100승을 뜻하는 말로 만든 'googol'이라는 신조어를 변형시켜 처음 구글이란 말을 쓴 것은 1997년이었다.

1998년 4월 개최된 월드와이드웹 컨소시엄(W3C)에서 페이지와 브린은 자신들의 연구 결과를 발표하게 되고, 이때부터 검색엔진 업계와 학계에서 관심을 끌기 시작했다. 구글은 사이트가 가지고 있는 메타태그나 키워드에만 의지하지 않고 페이지랭크라는 독특한 기법을 사용하여 웹페이지의 공정한 순위를 매긴다.

순수하게 수학적으로 접근한 구글 검색엔진의 새로운 아이디어와 수학적 알고리즘인 구글 행렬과 페이지랭크를 계산하는 방법은 인터넷 검색엔진 시장에 획기적인 새 패러다임을 제공하였다. 이를 통하여 학생들이 대학에서 배우는 선형대수학의 기본적인 내용이 생활 속에 널리 이용되고 있음을 확인 할 수 있다.

1. 구글 검색엔진의 기본 아이디어

우리는 구글 검색엔진에 사용된 페이지랭크를 통한 웹페이지의 외부인기도가 검색 순위에 반영될 때 페이지랭크 점수가 어떻게 구해지는지 알아보고 그 뒤에 숨어있는 수학적 아이디어의 핵심을 찾아 설명하도록 한다.

웹페이지에 대한 객관적인 순위를 만들어 내기 위하여 구글은 인터넷의 광범위한 구조를 직접 이용한다. 대부분의 다른 검색엔진들은 관계있는 사이트를 결정하기 위해 웹페이지의 제목과 내용을 체크한다. 그러나 구글은 한 웹페이지에서 다른 웹페이지로 연결하는 링크가 있으며, 그 링크를 일종의 투표로 본다. 많이 투표된 웹페이지를 중심으로 구글이 평가를 하게 된다. 구글의 페이지랭크는 어떤 웹페이지가 다른 웹페이지와 밀접한 관계가 있는지를 결정하고 관련된 구조를 표현하기 위해 하이퍼링크(Hyperlink)²⁾ 행렬을 만들고 행렬의 가장 큰 고윳값을 이용해서 검색에서 가장 근접한 사이트들을 찾아내는 것이다. 이 과정에서 구글은 Power Method(거듭제곱법)과 페이지랭크 연산법을 이용한다.

구글과 같은 검색엔진이 수행하는 기본적인 업무 중 첫 번째는 인터넷에 상주하며 사용자들이 접근하는 웹페이지를 파악하는 것이다. 두 번째 작업은 파악된 웹페이지에 대한 데이터를 수집하는 것이다. 수집된 데이터들은 검색어와 관련된 단어나 문구를 검색하는데 효과적으로 사용하게 된다. 세 번째 단계는 데이터로 수집한 웹페이지에 중요도(중요도: 웹페이지가 가지고 있는 중요성을 다른 웹페이지와 비교하여 점수로 표현한 것)를 기록하고 사용자가 검색을 할 때 수집하여 가지고 있던 웹페이지들의 데이터 중에 좀 더 중요한 자료를 검색결과의 상단에 보여주는 것이다.

우리는 이 세 번째 과정에서 수집된 웹페이지 데이터들의 집합에서 각각의 웹페이지에 대한 중요도를 어떠한 방법으로 결정하며 그 비율을 어떻게 정했는지에 대해 알아본다.

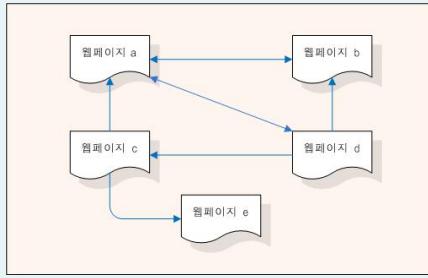
2. 페이지랭크(PageRank)공식의 건설

구글의 페이지랭크라는 개념은 쉽게 이야기해서 ‘사람들이 링크를 많이 거는 웹페이지는 사람들이 많이 찾아가는 곳일 테고, 그 만큼 정확한 정보가 있는 웹페이지일 것이므로 웹페이지의 페이지랭크 점수를 높게 주자.’라는 이론이다.

페이지랭크는 임의의 방문자가 어떤 웹페이지를 방문하는 빈도로 이해될 수도 있다. 새로운 웹페이지는 자신이 담고 있는 링크의 적절성에 따라 페이지랭크를 증가시키기도 하고, 감소시키기도 한다. (Avrachenkov & Litvak, 2005)

이제 구글 검색엔진의 수학적 모델에 대하여 살펴보자.

1단계: 인접(adjacency) 행렬의 건설



[그림 1] 5개 웹페이지로 이루어진 간단한 웹

위의 그림은 간단한 웹을 표현한 것이다. 주어진 웹페이지들로부터 얻어진 위의 연결관계를 다음과 같이 $n \times n$ 인접행렬 A 를 사용하여 나타낼 수 있다. (Page; Brin; Motwani & Winograd, 1998)

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \quad (2)$$

이 경우 [그림 1]의 웹페이지 e는 다른 웹페이지로의 링크를 가지고 있지 않으므로 행렬 A 의 마지막 열은 0의 값을 갖는다. dangling node³)라 불리는 외부로의 링크를 가지지 않는 웹페이지는 특별한 케이스이다. 게다가 인접행렬을 만들 때 각 웹페이지의 자기 자신으로의 링크는 무시하므로 행렬 A 의 대각선 성분은 모두 0의 값을 갖는 것을 알 수 있다.

2단계: 열정규화된 인접행렬 H 의 건설

행렬 A 의 각각의 열 A_j 를 A_j 의 성분들의 합으로 나누어 새로운 행렬 H 를 얻는다. 이 행렬을 열정규화된 인접행렬이라고 한다. 행렬 H 의 열을 H_j 라 하면 다음과 같은 공식에 의해 얻어진다.

$$H_j = \frac{A_j}{\sum_{k=1}^n A_{kj}}, \quad j = 1, \dots, n$$

이 과정을 식 (2)의 행렬 A 에 적용하면

1) 2011년에는 애플이 구글을 제치고 브랜드 가치 세계 1위를 차지했다고 보도했다. 구글은 2위 (1,724억 달러, 약 180조원)로 밀려났다. 유일하게 포함된 국내 기업 삼성은 67위를 차지했다.

$$H = \begin{bmatrix} 0 & 1 & \frac{1}{2} & \frac{1}{3} & 0 \\ \frac{1}{2} & 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & \frac{1}{3} & 0 \\ \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 & 0 \end{bmatrix} \quad (3)$$

를 얻는다.

3단계: 행렬 H 의 열 stochastic화

전 단계에서 얻어진 행렬 H 가 바로 페이지랭크 알고리즘에 이용되는 것은 아니다. 그 이유는 페이지랭크 알고리즘의 수렴성을 확보하기 위해서는 stochastic 행렬이 필요한데, 행렬 H 는 dangling node에 의해 열 stochastic 행렬이 아닐 수 있기 때문이다. 열stochastic은 ‘행렬의 모든 성분이 음수가 아니어야 하고 열의 합이 모두 1이어야 한다.’ 일단 위의 행렬 H 는 dangling node인 웹페이지에 의해 마지막 열의 합이 0 이므로 열 stochastic 행렬이 아니다. 그래서 행렬 H 로부터 모든 열의 합을 1이 되는 열 stochastic 행렬 S 를 다음과 같이 정의 한다.

$$S = H + \frac{\mathbf{e}\mathbf{a}^T}{n}$$

여기서 벡터 \mathbf{e} 는 모든 성분이 1인 열벡터이고, \mathbf{a} 는 $\sum_{i=1}^n H_{ij} = 0$ 이면 $a_j = 1$ 이고 $\sum_{i=1}^n H_{ij} \neq 0$ 이면 $a_j = 0$ 인 열벡터이다. 그러면 행렬 H 로 부터 열 stochastic 행렬 S 를 건설할 수 있다. Gershgorin 정리에 의해 열 stochastic 행렬(혹은 Markov 행렬) S 의 가장 큰 고윳값의 크기 $\rho(S)$ 는 1보다 작거나 같다. 또한 $S - I$ 의 행벡터(혹은 열벡터)의 합은 0이 되는데, 이것은 trivial sum이 아니다. 따라서 $\det(S - I) = 0$ 으로 부터 어떤 $i \in 1, \dots, n$ 에 대하여 $\lambda_i = 1$ 인 고윳값이 존재함을 알 수 있다. 그런데 문제점은 링크행렬이 크기가 매우 큰 sparse 행렬이라는 것이다. 따라서 일반적인 계산을 통해 고유벡터를 구하기보다는 거듭제곱법(Power method)을 사용하는 것이 효율적이다. 우선식 (3)의 행렬 H 를 사용하여 행렬 S 를 만들면 다음과 같이 된다.

-
- 2) 하이퍼링크(Hyperlink): <컴퓨터 전문용어> 하이퍼텍스트 문서 내의 단어나 구, 기호, 화상과 같은 요소를 그 하이퍼텍스트 내의 다른 요소 또는 다른 하이퍼텍스트 내의 다른 요소와 연결한 것. 하이퍼링크 행렬을 링크행렬이라 부르기로 한다.
 - 3) dangling node: 다른 웹페이지들과 연결이 끊겨 버린 상태
 - 4) http://matrix.skku.ac.kr/sglee/perron_frobenius/perron_frobenius.html
 - 5) http://en.wikipedia.org/wiki/Google_Books_Library_Project
-

$$S = \begin{bmatrix} 0 & 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{5} \\ \frac{1}{2} & 0 & 0 & \frac{1}{3} & \frac{1}{5} \\ 0 & 0 & 0 & \frac{1}{3} & \frac{1}{5} \\ \frac{1}{2} & 0 & 0 & 0 & \frac{1}{5} \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{5} \end{bmatrix} \quad (4)$$

4단계: 구글 행렬 G 의 건설

행렬 S 가 유일한 stationary distribution 벡터를 갖는다는 것을 보증할 수 없기 때문에 아직 끝난 것이 아니다. (즉, 정확한 하나의 페이지랭크 벡터가 없을 수도 있다는 의미) 따라서 수렴하는 하나의 stationary distribution 벡터 \mathbf{x} 가 있다는 것을 보장하기 위해 행렬 S 가 irreducible인 동시에 stochastic임을 확실히 해야 한다. 정의에 의하여 “주어진 정사각행렬 A 에 대하여, $P^T A P = \begin{bmatrix} X & Y \\ O & Z \end{bmatrix}$ 인 치환 행렬 P 와 정사각행렬 X, Z 가 존재하지 않을 때 행렬 A 는 irreducible이다.” 행렬이 irreducible인 것은 대응하는 그래프가 strongly connected하다는 것과 동치이다.(Horn & Johnson, 1985) 또 A 가 primitive 행렬이면, 언제나 A 는 irreducible 행렬이다. 이에 보태서 주어진 정사각행렬 A 가 primitive일 필요충분조건은 어떤 자연수 k 에 대하여 A^k 가 양의 행렬이 되는 것이다.⁴⁾ 그래서 우리는 식 (4)의 행렬 S 를 이용하여 irreducible한 열 stochastic 행렬 G 를 다음과 같이 정의한다.

$$G = mS + (1-m)E \quad (5)$$

여기서 m 은 $0 \leq m \leq 1$ 이고 $E = \frac{\mathbf{e}\mathbf{e}^T}{n}$ 이다. 구글검색에서는 보통 m 을 0.85로 이용한다.

이 행렬 G 가 바로 우리가 필요로 했던 구글행렬이 된다.

이제 페이지랭크를 구하는 기본 알고리즘인 식(1) $\mathbf{x}_{k+1} = H\mathbf{x}_k$ 에 행렬 H 대신 행렬 G 를 대입하여 새 페이지랭크 알고리즘을 다음과 같이 정의한다. $\mathbf{x}_{k+1} = G\mathbf{x}_k$

그러나 크기가 1인 고윳값이 2개 이상 존재하면 거듭제곱법의 수렴성이 보장되지 않는다. 또한 행렬의 주위에 블록 O 행렬이 있다면, 페이지 랭킹에서 아예 제외되는 웹페이지가 생기는데, 이런 경우 현실 문제를 적절히 반영한다고 볼 수 없다. 따라서 우리가 다루기를 원하는 행렬의 의미를 모두 담고 있으면서 가장 큰 고윳값이 1개뿐인(대수적 중복도가 1인)행렬인 primitive 행렬로 변환시킨 것이 구글 행렬이다. 이 부분의 이론에 대하여 소개하고 다음 단계로 넘어간다.

집합 $\sigma(G)$ 를 행렬 G 의 고윳값들의 집합, 정사각행렬 G 의 spectral radius를 $\rho(G) = \max_{\lambda \in \sigma(G)} |\lambda|$ 라 하면 Perron–Frobenius정리에 의해 구글 행렬 $G = mS + (1-m)E$ 에

대하여 $\rho(G)=1$ 인 고윳값 1은 G 의 Gershgorin circle 상의 유일한 고윳값이고, 고윳값 1의 대수적 중복도는 1임을 알게 된다. 그리고 $\lambda=\rho(G)$ 는 행렬 G 의 유일한 가장 큰 고윳값이 된다. 더구나 대응하는 그래프는 strongly connected이고 어떤 자연수 k 에 대하여 G^k 가 양의 행렬이 되는 irreducible 행렬 G 는 primitive 행렬이다. 즉, 구글 행렬 $G = mS + (1-m)E$ 가 primitive 행렬이라는 의미이다.(Horn & Johnson, 1985)

여기서 $\lambda > 0$ 이고 $\lambda \in \sigma(G)$ 인 대수적 중복도가 1인 가장 큰 고윳값 $\lambda = \rho(G)$ 를 G 의 Perron 근(root)라 하고, $G\mathbf{x} = \lambda\mathbf{x}$ 인 양의 고유벡터 $\mathbf{x} > 0$ 가 존재한다. 그리고 $G\mathbf{x} = \lambda\mathbf{x} \circ$ 면서 $\mathbf{x} > 0$ 이고 $\|\mathbf{x}\|_1 = 1$ 인 G 의 고유벡터를 G 의 Perron 벡터라 한다.

위의 전설 과정에서 우리가 건설한 구글 행렬이 수렴하는 유일한 페이지랭크 벡터를 갖는다는 것을 보인 셈이다.

이제 수렴성을 분석해 보자. 만일 행렬 P 가 (G 와 마찬가지로) primitive 행렬이라하자. 그러면 $\lim_{k \rightarrow \infty} P^k$ 가 존재한다. 특히, $r = \rho(P)$ 가 이 행렬 P 의 Perron 근이고, \mathbf{p} 와 \mathbf{q} 를 각각 primitive 행렬 P 와 P^T 의 Perron 벡터 라 할 때 $\lim_{k \rightarrow \infty} (P/r)^k = \frac{\mathbf{p}\mathbf{q}^T}{\mathbf{q}^T \mathbf{p}}$ 임을 알고 있다.(Meyer, 2000) 이를 이용하면 \mathbf{x} 가 P 의 (right) Perron 벡터라 하고, \mathbf{e}/n 를 P^T 의 Perron 벡터라 하면, $\lim_{k \rightarrow \infty} P^k = \frac{(\mathbf{e}/n)\mathbf{x}^T}{\mathbf{x}^T (\mathbf{e}/n)} = \frac{(\mathbf{e})\mathbf{x}^T}{\mathbf{x}^T (\mathbf{e})} = \mathbf{e}\mathbf{x}^T > 0$ 를 얻는다. 벡터 \mathbf{e} 는 모든 성분을 1로 갖는 열 벡터이다.

이것으로부터 $P\mathbf{x} = \mathbf{x}$ 즉, \mathbf{x} 가 P 의 (right) Perron 벡터임이 확인되고, 따라서 $\mathbf{x}^T P^T = \mathbf{x}^T$ 가 되어 \mathbf{x}^T 는 P^T 의 (left) Perron 벡터임이 확인된다. 따라서 $\|\mathbf{x}\|_1 = 1$ 인 임의의 초기벡터 \mathbf{x}_0 에 대하여 $P^k \mathbf{x}_0 = \mathbf{x}_k \circ$ 며, primitive 행렬 P 에 대하여 $\lim_{k \rightarrow \infty} P^k = \mathbf{e}\mathbf{x}^T \circ$ 이고 $\lim_{k \rightarrow \infty} \mathbf{p}_k = \mathbf{x} \circ$ 이다. $P\mathbf{x} = 1\mathbf{x}$ 인 유일한 단위 벡터 \mathbf{x} 가 존재한다는 의미이다.

수렴성을 보여주는 위의 설명은 구글행렬 G 가 벡터 \mathbf{x} 를 (right) Perron 벡터로 가지는 것과 그에 대응하는 대수적 중복도가 1인 고윳값 $\lambda = \rho(G) = 1$ 을 가진다는 것을 확인해 준다. 그리고 이 알고리즘을 이용하면 반복연산을 통하여 $\mathbf{x}_{k+1} \approx G\mathbf{x}_k$ 를 만족하는 근사값을 페이지랭크 벡터 \mathbf{x} 로 언제나 구할 수 있다는 것이다. 이 \mathbf{x} 가 구글행렬 G 의 Perron root라 하고, $G\mathbf{x} = \lambda\mathbf{x}$ 인 양의 고유벡터 $\mathbf{x} > 0$ 가 존재한다. 그리고 $G\mathbf{x} = \lambda\mathbf{x} \circ$ 면서 $\mathbf{x} > 0$ 이고 $\|\mathbf{x}\|_1 = 1$ 인 행렬 G 의 고유벡터는 행렬 G 의 페이지랭크 벡터이고, Perron 벡터이다. 따라서 행렬 G 의 페이지랭크 벡터를 찾는 것은 행렬 G 의 (dominant right) 고유벡터 또는 G^T 의 (dominant left) 고유벡터, 즉, Perron 벡터를 찾는 것과 동치이다.

이제 구글 행렬 G 의 페이지랭크 벡터를 구하는 문제를 생각하자, 구글은 수십억 대의 홈페이지를 다루므로, 실제로 그런 크기의 행렬 G 를 만들어 Perron 벡터를 구하는 것은 현실과 다르다. 인터넷상의 웹페이지는 적어도 80억 이상의 개수를 가지므로 간단하게 계산되어질 수 없다. 따라서 실제로는 페이지랭크 벡터를 아래에 설명하는 거듭제곱법을 이용하여 구한다. (Langville & Meyer, 2004)

단계 5: 거듭제곱법(Power method)

$$\begin{aligned}
\mathbf{x}_k &= G\mathbf{x}_{k-1} \\
&= \left(mS + (1-m) \frac{\mathbf{e}}{n} \mathbf{e}^T \right) \mathbf{x}_{k-1} \\
&= mS\mathbf{x}_{k-1} + (1-m) \frac{\mathbf{e}}{n} \mathbf{e}^T \mathbf{x}_{k-1} \\
&= mS\mathbf{x}_{k-1} + (1-m) \frac{\mathbf{e}}{n} \\
&= m \left(H + \frac{\mathbf{e}\mathbf{a}^T}{n} \right) \mathbf{x}_{k-1} + (1-m) \frac{\mathbf{e}}{n} \\
&= mH\mathbf{x}_{k-1} + \frac{\mathbf{e}}{n} (m\mathbf{a}^T \mathbf{x}_{k-1} + (1-m))
\end{aligned}$$

거듭제곱법(이상구, 2009)을 이용한 수렴속도는 τ 를 $\|\mathbf{x}_{(k)} - \mathbf{x}_{(k-1)}\|$ 라고 할 때 $\log_{10} \tau / \log_{10} m$ 임과 페이지와 브린이 $m = 0.85$ 로 놓으면 대개 50번에서 100번의 연산을 통하여 원하는 페이지랭크 벡터를 구할 수 있음을 보였다. (Langville & Meyer, 2003)

이제 어떻게 페이지랭크 벡터가 계산되는지를 확인해 보자. 첫째 웹페이지 i 의 페이지랭크는 페이지랭크 벡터 \mathbf{x} 의 i 번째 성분인 값 x_i 이다. 여기서 벡터 \mathbf{x} 는 $\|\mathbf{x}\|_1 = 1$ 인 Perron 벡터이다. 이 벡터 \mathbf{x} 를 통해 웹상에서의 페이지 i 의 순위를 가늠해 볼 수 있는 점수를 구한다.

위의 전 과정을 요약하면 다음과 같다. 웹의 연결성으로부터 얻은 행렬 A 의 각 성분을 $H_j = \frac{A_j}{\sum_{k=1}^n A_{kj}}$ 를 이용하여 정규화한 행렬 H 를 만들게 된다. 그리고 행렬 H 로부터

열 stochastic 행렬인 S 를 $S = H + \frac{\mathbf{e}\mathbf{a}^T}{n}$ 로 정의하고, 이 때 벡터 \mathbf{e} 는 모든 성분이 1인 열(column) 벡터이고, 벡터 \mathbf{a} 는 $\sum_{i=1}^n H_{ij} = 0$ 이면 $a_j = 1$ 이고 $\sum_{i=1}^n H_{ij} \neq 0$ 이면 $a_j = 0$ 인 열벡터이다. 이로부터 여기서 m 은 $0 \leq m \leq 1$ 의 값인 0.85로 하고 $E = \frac{\mathbf{e}\mathbf{e}^T}{n}$ 로 정의하여 구글행렬 $G = mS + (1-m)E$ 를 얻은 것이다. m 을 0.85로 선택한 것은 임의로 한 것이 아니다. 이와 관련된 자세한 내용은 (Haveliwala & Kamvar, 2003)에 소개 되었다.

이와 같이 문제 표현형식을 바꾸면, 웹페이지의 순위를 매기기 위한 문제가 정사각행렬에서 고유벡터를 찾는 우리가 잘 아는 문제로 바뀌게 된다.(정의에 의해 행렬 G 의 고윳값 λ 와 고유벡터 \mathbf{x} 는 방정식 $G\mathbf{x} = \lambda\mathbf{x}$ 를 만족한다.) 위 행렬 G 를 구글 행렬이라 한다. 이제는 열 stochastic 행렬 G 의 가장 큰 고윳값 1에 대응하는 고유벡터 \mathbf{x} 를 구하는 일이 남는다.

위의 전 과정을 예를 들어 설명하면 다음과 같다. 미리 언급하였듯이 구글에서는 $m = 0.85$ 을 사용한다. 그래서 구글에서 사용하는 $m = 0.85$ 의 값을 사용하여 얻은 식 (5)

의 행렬 S 를 이용하여 행렬 G 를 구하면 아래와 같다.

$$G = \begin{bmatrix} \frac{3}{100} & \frac{22}{25} & \frac{91}{200} & \frac{47}{150} & \frac{1}{5} \\ \frac{91}{200} & \frac{3}{100} & \frac{3}{100} & \frac{47}{150} & \frac{1}{5} \\ \frac{3}{100} & \frac{3}{100} & \frac{3}{100} & \frac{47}{150} & \frac{1}{5} \\ \frac{91}{200} & \frac{3}{100} & \frac{3}{100} & \frac{3}{100} & \frac{1}{5} \\ \frac{3}{100} & \frac{3}{100} & \frac{91}{200} & \frac{3}{100} & \frac{1}{5} \end{bmatrix}$$

페이지랭크가 담고 있는 내용을 정리하면 웹페이지 i 의 페이지랭크는 $\|x\|_1 = 1$ 인 Perron 벡터 x 의 i 번째 x_i 성분의 값이고 페이지랭크 벡터는 각 웹페이지에 접근하게 될 가능성을 반영하는 Probability 벡터이다.(Langville & Meyer, 2003)

우리가 사용하고 있는 인터넷의 모든 사이트(웹페이지)가 n 개의 웹페이지로 만들어진 웹이라 하자. 그러면 인터넷 사용자들은 어떠한 웹페이지를 시작페이지로 설정하여 인터넷을 이용하게 될 것이다.

인터넷을 이용하던 중에 r 개의 링크를 가지고 있는 어느 특정 웹페이지를 열어보게 된다면 그 웹페이지에서 $\frac{m}{r}$ 의 확률(구글에서는 m 을 0.85를 사용하였다.)을 가지고 웹페이지가 가지고 있는 임의의 링크를 선택을 하게 된다. 웹페이지에 있는 링크를 선택하지 않을 확률은 $\frac{1-m}{n}$ 이 되고 인터넷 사용자가 선택할 수 있는 모든 경우의 수는 두 확률로 합으로 계산이 된다. 즉, $r\frac{m}{r} + n\frac{(1-m)}{n} = 1$ (여기서 r 은 웹페이지가 가진 링크의 개수, n 은 인터넷 웹상의 모든 웹페이지의 개수)가 된다.

구글의 페이지랭크 알고리즘에서 기본 전제로 한 ‘중요한 웹페이지는 다른 웹페이지로부터 더 많이 연결되어 있다.’에 따라 인터넷 사용자는 자주 페이지랭크 점수가 높은 웹페이지로 연결되는 링크를 더 자주 선택하게 되고 페이지랭크 점수가 높은 페이지에 많은 인터넷 사용자가 접근을 하게 된다.

접근을 하는 횟수가 다른 웹페이지에 비하여 많다는 것은 바꿔 생각하면 그만큼 인터넷 사용자들이 해당 웹페이지에 머물게 된다는 것이다. 즉, 식 $x = Gx$ 에서 정규화 된 벡터 x 의 성분 x_i 는 인터넷 사용자들이 웹페이지 i 에서 머무르는 짧은 시간을 의미하는 것으로 판단 생각할 수 있게 된다. 링크구조를 데이터베이스로 가지는 검색엔진의 설계에서 링크구조는 고정적이다. 고정된 링크구조를 가지는 웹에서는 한번 정의된 x 는 계속해서 사용할 수 있다. 하지만 인터넷에서의 웹페이지는 단 하나의 구조로 고정되어 있지 않는다. 수많은 사이트와 웹페이지는 생성되었다가 사라지기도 하고 그들 사이를 잇는 링크들 또한 필요성에 따라 계속해서 생겨나고 사라지는 동적인 공간이기 때문에 웹페이지의 페이지랭크 계산을 통한 고유벡터 x 의 개선은 매우 중요한 문제가 된다. 페이지랭크 기술은 키워드별로 인터넷 사용자가 클릭하는 웹페이지의 경

로를 알고리즘화해 웹페이지들 간의 상호 관련성을 계산해내고, 웹페이지간의 관련성과 웹페이지내의 관련어 배치 등을 고려해 고객이 원하는 결과를 가장 빠르고 정확하게 제공하는 기술이다. 구글 검색엔진의 경우 어떤 내용을 검색창에 넣어도 0.5초면 자신이 찾는 가장 근접한 검색결과를 제시해준다. 이 기능은 아직 어떤 검색엔진도 따라올 수 없는 독보적 기술로 평가받고 있다.

현재 구글은 다양한 분야로 사업을 확장하고 있다. 수학자는 ‘구글 북스 라이브러리 프로젝트’⁵⁾에 대하여도 관심을 가져야 한다. 구글의 목표 중 하나가 20세기에 발간된 모든 책을 전자화 한다는 것이라고 들었다. 구글은 이미 일본 게이오대학과 미국 하버드대학 등 세계 40곳 이상의 도서관 및 3만 곳 이상의 출판사와 연계해 지금까지 1500만 권 이상을 전자화했다. 2011년 3월에는 영국의 대영도서관 소장 장서 25만 권을 전자화하기로 합의했다고 발표했다. 이런 변화가 수학을 하는 우리에게 조만간 또 다른 큰 영향을 미칠 것이라는 것을 느낄 수 있다.

3. 결 론

인류 역사의 혁명적 발전 시기마다 그 사회가 안고 있던 문제를 해결하는 과정의 중심에 수학이 있었다. 우리나라의 학생들은 보통 수학공부를 하는 이유를 ‘입학시험을 잘 보기위해 서’라고 생각하고 있다. 따라서 수학은 우리의 삶에 불필요한 과목이라고 생각하기도 한다. 본 원고에서는 구글 검색 엔진 속의 존재하는 선형대수학 이론의 일부를 간략하게 소개했다. 이는 수학의 발전이 인류 사회의 발전 역사와 함께한다는 것을 보여주는 좋은 예가 될 수 있다.

4. 참고문헌

- 이상구 (2009). 현대선형대수학 3판, 서울 : 경문사
- Avrachenkov K. & Litvak N. (2005). The Effect of New Links on Google PageRank
<http://wwwhome.math.utwente.nl/~litvakn/StochModels06.pdf>
- Brin S.; Page L.; Motwami R. & Winograd T. (1998). The PageRank Citation Ranking: Bringing Order to the Web, Technical report, Computer Science Department, Stanford University, Stanford, CA
- Haveliwala T. H. & Kamvar S. D. (2003). The Second Eigenvalue of the Google Matrix, Stanford University Technical Report
<http://www.stanford.edu/~sdkamvar/papers/secondeigenvalue.pdf>
- Horn R. & Johnson C.R. (1985). Matrix Analysis, Cambridge, 1985 ISBN 0521305861
- Langville A. N. & Meyer C. D. (2003). Fiddling with PageRank
http://meyer.math.ncsu.edu/Meyer/PS_Files/FiddlingPageRank.pdf
- Langville A. N. & Meyer C. D. (2004). The Use of the Linear Algebra by Web Search Engines
http://meyer.math.ncsu.edu/Meyer/PS_Files/IMAGE.pdf
- Langville A. N. & Meyer C. D. (2005). Deeper inside PageRank, Internet Math.
- Meyer C. D. (2000). Matrix Analysis and Applied Linear Algebra, SIAM, Philadelphia. ■



읽을거리3

인공지능과 AI를 활용한 안면 인식기술

https://www.sas.com/ko_kr/solutions/ai-mic/blog/face-recognition-technology.html

스티븐 스필버그 감독과 톰 크루즈 주연의 영화 ‘마이너리티 리포트’는 예술적 측면뿐 아니라 사회적으로도 다양한 이슈를 만들었다. 톰 크루즈가 맡은 주인공 존 앤더슨은 도시 곳곳에 설치돼 신원을 확인하는 홍채 인식 시스템으로부터 숨기 위해 안구 이식 수술을 받는다.

안면 인식 시스템의 원리

안면 인식 시스템은 대부분 ‘협력 시스템’ 개념을 기반으로 구축된다. 미리 정해진 거리에서 적절한 조명과 함께 카메라를 똑바로 바라보고 찍은 이미지를 데이터베이스에 저장된 검증된 이미지와 대조하는 방식이다. 이런 유형의 시스템은 국경 통제와 물리 보안 시스템에서 광범위하게 사용되고 있다. ‘이미저스 테크놀로지(Imagus Technology)’라는 호주 기업이 군중 속 특정 얼굴을 인식하는 ‘비협력 시스템(non-cooperative systems)’이라는 작업을 수행하는 군중 속 안면 인식 시스템을 개발했다.

안면 인식 시스템의 다양한 활용

안면 인식 시스템은 신원 파악뿐 아니라 다양한 마케팅과 인구 통계에 활용되고 있다. 예를 들어 상점에서 광고나 디스플레이를 보는 사람 수를 세거나 성별과 연령대를 구분하고, 광고를 본 시간 등 다양한 데이터를 창출할 수 있다. 이 같은 데이터는 타깃 마케팅과 광고에 효과적으로 활용될 수 있다. 실제 호주에서 디지털 옥외 광고 장치인 ‘DOOH(Digital Out-of-Home)’가 이미 사용되고 있는데, 이 장치는 특정인을 대상으로 최적의 광고 시간을 결정해 광고비 낭비를 최소화한다. 예를 들어 아줌마가 다가올 때에만 가족용 파이 광고를 노출하는 방식이다.

안면 인식 기술은 이미 확산 단계에 접어들었다. 그 예로 편재형 보안 카메라(pervasive security cameras)는 광대역 통신망을 통해 방대한 클라우드 컴퓨팅 리소스와 연결되어 있으며, 안정적으로 이용 가능한 소프트웨어가 개발된다면 군중 속 얼굴 인식은 시간 문제일 것이다.

<인공지능을 위한 기초수학> <http://matrix.skku.ac.kr/math4ai/>

Part I. 행렬과 데이터분석 <http://matrix.skku.ac.kr/math4ai/part1/>

Part II. 다변수 미적분학과 최적화 <http://matrix.skku.ac.kr/math4ai/part2/>

Part III. 확률통계와 빅데이터 <http://matrix.skku.ac.kr/math4ai/part3/>

Part IV. 빅데이터와 인공지능 <http://matrix.skku.ac.kr/math4ai/part4/>

『인공지능을 위한 기초수학』은 모든 대학생과 대학원생을 대상으로 인공지능에 필요한 기초수학을 선별하여 한 학기 강의용으로 만든 강좌와 교재이다. 인공지능 이해에 필수적인 수학 콘텐츠인 선형대수학, 다변수 미적분학, 기초 통계와 확률, SVD, 주성분 분석(PCA) 및 그래디언트 알고리즘과 파이썬 및 R 코드 및 실습실을 포함하는 내용으로 구성한다. 본 강의는 처음부터 끝까지 인공지능(머신러닝)을 이해하는데 필요한 수학적 개념을 소개하는데 집중하고 있다. 먼저, 인공지능의 알고리즘을 이해하는 데 필요한 최소한의 수학 개념을 고교, 대학 수학 과정의 수준으로 설명하고, 이어서 배운 개념들이 실제로 인공지능을 개발할 때 어떻게 쓰이는지, 이어서 잘 알려진 예와 알고리즘을 이용하여 쉽게 설명한다. 특히 수학적인 이론과 함께 수학적 풀이와 관련 파이썬(Python) 및 R 코드를 교재에 제공하고 마련된 클라우드 컴퓨팅 실습실을 활용하여, 어렵지 않게 복잡한 계산과 데이터 처리 및 시각화를 직접 하면서 학습할 수 있도록 한다.



Copyright @ 2019 SKKU Matrix Lab. All rights reserved.,

Made by Prof. Sang-Gu Lee (이상구) [sglee at skku.edu](mailto:sglee@skku.edu) with Dr. Jae Hwa Lee



[온드림빅북] 인공지능을 위한 기초수학

발행일 2019년 12월 09일

저작권자 빅북운동본부

대표자 조영복

저자 이상구, 이재화

주소 부산광역시 금정구 구서2동 248-10 현대빌딩 4F

문의처 051-517-0268

홈페이지 <http://bigbook.or.kr/>

발행처 교보문고 퍼플

출판등록 2012년 09월 07일 제3-2012-167호

주소 서울시 종로구 종로1가 1번지

대표전화 1544-1900

홈페이지 www.kyobobook.co.kr

ISBN 978-89-24-06612-8 (93560)

© 빅북운동본부 2019

인공지능을 위한 기초수학

(Basic Mathematics for Artificial Intelligence)

이상구 with 이재화

* 첫 강의 [Dummy를 위한 인공지능] <https://youtu.be/VZbv6BG-xIY> (12:50)
Math for AI 홈페이지 <http://matrix.skku.ac.kr/math4ai/>

 **AI Big Book** Chung Mong-Koo Foundation

현대차
정몽구 재단



빅북은 전자책으로도 볼 수 있으며 홈페이지에서 무료로 다운 받을 수 있다.

공유와 협력의 교과서만들기 운동본부
www.bigbook.or.kr

교보문고 POD pod.kyobobook.co.kr

