최적설계Optimum Design

- 주요 참고 문헌 : 다음 참고문헌에서 예제 및 그림 인용, 최적설계입문Introduction to Optimum Design, Jasbir S. Arora[1]

경사도 벡터Gradient

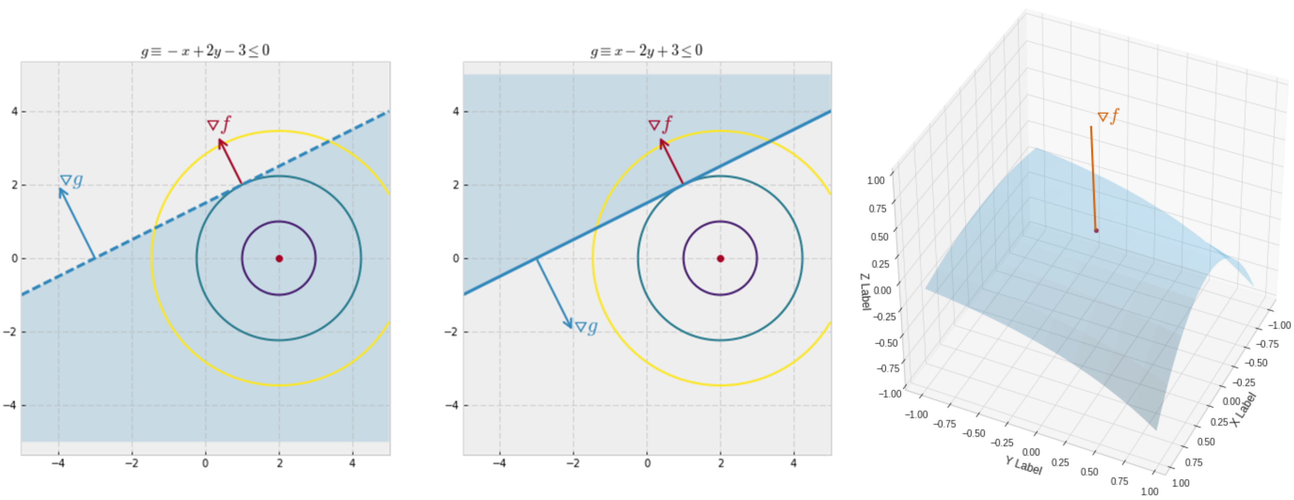
- 선형회귀에서 회귀계수를 찾아내기 위한 알고리즘의 핵심
- n개의 변수 $x_1, x_2, \ldots, x_n$에 대한 함수 $f(\mathbf{x})$의 $\mathbf{x}^*$에서의 편미분

$$c_i = \frac{\partial f(\mathbf{x}^*)}{\partial x_i}, \quad i = 1, \ldots, n$$

- 이를 열벡터로 모으면 경사도 벡터

$$\nabla f(\mathbf{x}^*) = \begin{bmatrix} \dfrac{\partial f(\mathbf{x}^*)}{\partial x_1} \\ \dfrac{\partial f(\mathbf{x}^*)}{\partial x_2} \\ \vdots \\ \dfrac{\partial f(\mathbf{x}^*)}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f(\mathbf{x}^*)}{\partial x_1} & \dfrac{\partial f(\mathbf{x}^*)}{\partial x_2} & \cdots & \dfrac{\partial f(\mathbf{x}^*)}{\partial x_n} \end{bmatrix}^{\mathrm{T}}$$

- ∇ 각 성분으로의 미분 연산을 의미하며 'nabla'라고 읽음
- 다음은 2변수, 3변수 함수의 경사도 벡터 ∇f를 나타냄



헤시안Hessian

- 경사도 벡터를 한번 더 미분, 즉 두번 미분

$$\nabla^2 f(\mathbf{x}^*) = \frac{\partial^2 f(\mathbf{x}^*)}{\partial \mathbf{x} \partial \mathbf{x}} = \begin{bmatrix} \frac{\partial^2 f(\mathbf{x}^*)}{\partial x_1^2} & \frac{\partial^2 f(\mathbf{x}^*)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(\mathbf{x}^*)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(\mathbf{x}^*)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(\mathbf{x}^*)}{\partial x_2^2} & \cdots & \frac{\partial^2 f(\mathbf{x}^*)}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f(\mathbf{x}^*)}{\partial x_n \partial x_1} & \frac{\partial^2 f(\mathbf{x}^*)}{\partial x_n^{\partial} x_2} & \cdots & \frac{\partial^2 f(\mathbf{x}^*)}{\partial x_n^2} \end{bmatrix}$$

- 헤시안은 대칭행렬
- 행렬형식 표현

$$\mathbf{H} = \left[\frac{\partial^2 f}{\partial x_i \partial x_j} \right]; \quad i = 1, n, \; j = 1, n$$

- 예제 : 다음 함수에서 점(1,2)에서의 경사도벡터와 헤시안을 계산

$$f(\mathbf{x}) = x_1^3 + x_2^3 + 2x_1^2 + 3x_2^2 - x_1 x_2 + 2x_1 + 4x_2$$

- 1계 편도함수

$$\frac{\partial f}{\partial x_1} = 3x_1^2 + 4x_1 - x_2 + 2$$
$$\frac{\partial f}{\partial x_2} = 3x_2^2 + 6x_2 - x_1 + 4$$

- $x_1 = 1, x_2 = 2$에서의 경사도 벡터

$$\nabla f(1,2) = \begin{bmatrix} 7 \\ 27 \end{bmatrix}$$

- 헤시안을 위해 한번 더 미분

$$\frac{\partial^2 f}{\partial x_1^2} = 6x_1 + 4 \qquad \frac{\partial^2 f}{\partial x_1 \partial x_2} = -1$$
$$\frac{\partial^2 f}{\partial x_2 \partial x_1} = -1 \qquad \frac{\partial^2 f}{\partial x_2^2} = 6x_2 + 6$$

- $x_1 = 1, x_2 = 2$에서의 헤시안

$$\mathbf{H}(1,2) = \begin{bmatrix} 10 & -1 \\ -1 & 18 \end{bmatrix}$$

테일러 급수Taylor series

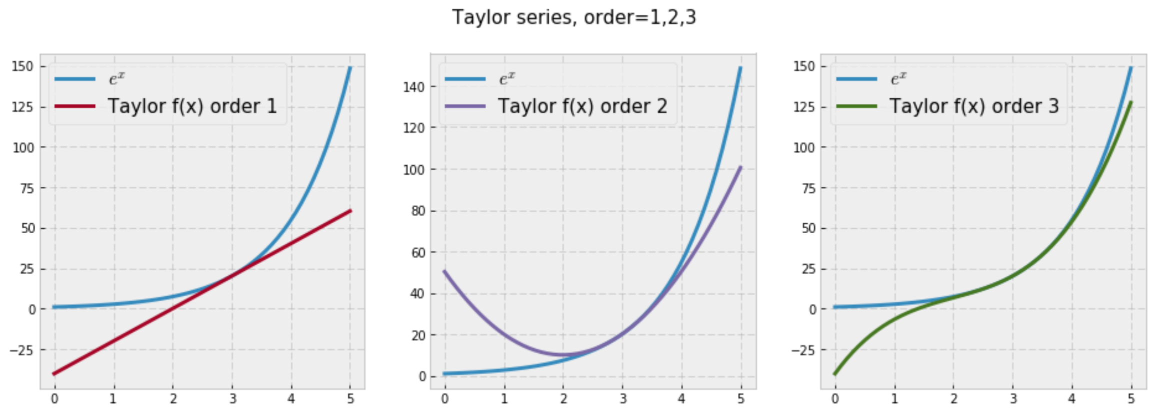
- 미분가능한 함수를 어느 점 근처에서 다항식으로 근사하는 방법
- 스코틀랜드의 수학자 그레고리James Gregory가 시초
- 1715년 이후, 영국의 수학자 테일러Brook Taylor에 의해 널리 알려짐[2]

$$f(x) = f(x^*) + \frac{df(x^*)}{dx}(x - x^*) + \frac{1}{2}\frac{d^2 f(x^*)}{dx^2}(x - x^*)^2 + R$$

- 위 식의 의미는 어떤 점 x^*이 근처에서 함수 $f(x)$의 값은 위 식처럼 계산될 수 있다는 것
- 약간 다르게 표현 하기 위해 $x - x^* = d$로 두고

$$f(x^* + d) = f(x^*) + \frac{df(x^*)}{dx}d + \frac{1}{2}\frac{d^2 f(x^*)}{dx^2}d^2 + R$$

- 어떤 점 x^*에서 d만큼 떨어진 곳의 함수값은 $f(x^*)$를 중심으로 위 식처럼 계산될 수 있다.
- 실험으로 확인



2변수 이상 테일러 급수

- 2차 항을 각 변수별로 모두 만들어 줌

$$
\begin{aligned}
& f(x_1, x_2) \\
&= f(x_1^*, x_2^*) + \frac{\partial f}{\partial x_1}(x_1 - x_1^*) + \frac{\partial f}{\partial x_2}(x_2 - x_2^*) \\
&\quad + \frac{1}{2}\left[\frac{\partial^2 f}{\partial x_1^2}(x_1 - x_1^*)^2 + 2\frac{\partial^2 f}{\partial x_1 \partial x_2}(x_1 - x_1^*)(x_2 - x_2^*) + \frac{\partial^2 f}{\partial x_2^2}(x_2 - x_2^*)^2\right] \\
&\quad + R
\end{aligned}
$$

- 시그마 기호로 쓰면

$$
\begin{aligned}
f(x_1, x_2) = f(x_1^*, x_2^*) + \sum_{i=1}^{2} \frac{\partial f}{\partial x_i}(x_i - x_i^*) \\
+ \frac{1}{2}\sum_{i=1}^{2}\sum_{j=1}^{2} \frac{\partial^2 f}{\partial x_i \partial x_j}(x_i - x_i^*)(x_j - x_j^*) + R
\end{aligned}
$$

- 색깔로 표시된 부분은 경사도벡터와 헤시안 행렬의 성분임을 이용하여 더 간략히 쓰면

$$
f(\mathbf{x}) = f(\mathbf{x}^*) + \triangledown f^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^*) + \frac{1}{2}(\mathbf{x} - \mathbf{x}^*)^{\mathrm{T}}\mathbf{H}(\mathbf{x} - \mathbf{x}^*) + R
$$

- 1 변수와 마찬가지로 $\mathbf{x} - \mathbf{x}^* = \mathbf{d}$ 로 두면

$$
f(\mathbf{x}^* + \mathbf{d}) = f(\mathbf{x}^*) + \triangledown f^{\mathrm{T}}\mathbf{d} + \frac{1}{2}\mathbf{d}^{\mathrm{T}}\mathbf{H}\mathbf{d} + R
$$

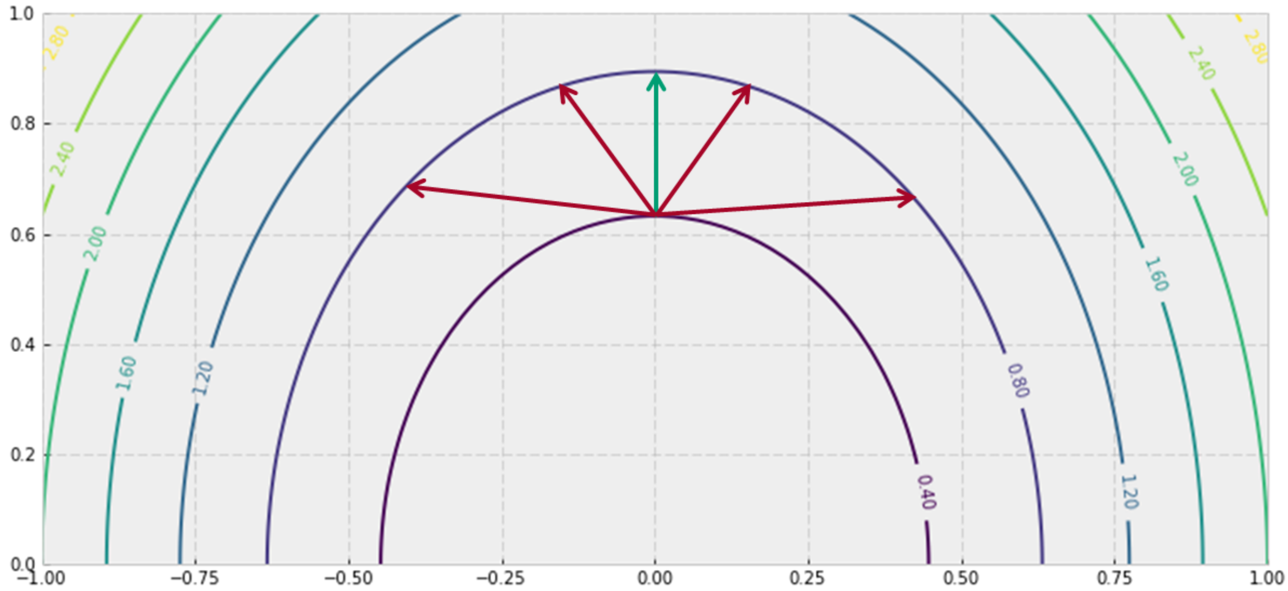
- $f(\mathbf{x}^*)$를 이항하면 좌변은 $\Delta f = f(\mathbf{x}^* + \mathbf{d}) - f(\mathbf{x}^*)$ 가 되고 이는 $\mathbf{x}^*$에서 $\mathbf{x}$로 이동했을 때 함수값의 변화량이 된다.

$$
\Delta f = \triangledown f^{\mathrm{T}}\mathbf{d} + \frac{1}{2}\mathbf{d}^{\mathrm{T}}\mathbf{H}\mathbf{d} + R
$$

경사도벡터의 중요 성질

- 다음 두 경사도벡터의 성질은 경사하강법[gradient descent]의 근간

- 1. 경사도벡터는 함수표면의 접면에 수직
- 2. 경사도벡터의 방향은 함수의 최대증가방향



- 위 그림에서 함수값 0.4에서 0.8로 가려고 할때 가장 빠른 경로를 생각
- 자세한 증명은 부록으로 제공
- 방향도함수[directional derivative]의 개념을 사용하지만 지금까지 배운 편미분과 테일러시리즈의 지식으로 이해 가능

경사도, 헤시안을 이용한 최적조건

- 최적해를 위한 필요조건과 충분조건
- 필요조건 : 최적점에서 만족되어야 할 조건
- 필요조건을 만족하지 않으면 최적점이 아님
- 하지만 필요조건이 만족되었다고 최적점을 보장하는 것은 아님
- 따라서 필요조건을 만족하면 후보최적점candidate optimum point
- 후보최적점에 대해 충분조건을 검토하여 최적점을 판단

최적화 문제에는

- 다음 두 종류의 문제가 존재

비제약 최적설계 문제

- 주어진 목적함수의 값을 무조건 작게 하는 변수 $\mathbf{x}$의 값을 찾는 문제

제약 최적설계 문제

- 주어진 목적함수의 값을 작게하나 지정된 $\mathbf{x}$의 범위 내에서만 값을 찾는 문제

비제약 최적설계 문제

- 최적화 시킬 변수 $\mathbf{x}$에 관한 제약이 없는 상태에서 $f(\mathbf{x})$를 최소화
- 제약 문제를 비제약 문제로 환원하는 것이 가능하므로 비제약 문제를 이해하는 것이 중요
- 준비
  - $\mathbf{x}^*$를 지역 최소점으로 가정
  - 최소점으로 부터 약간의 이동 $\mathbf{d} = \mathbf{x} - \mathbf{x}^*$
  - 그에 따른 함수의 변화

$$\Delta f = f(\mathbf{x}) - f(\mathbf{x}^*) \geq 0 \tag{1}$$

  - 식 (1)은 임의의 증분 $\mathbf{d}$에 대해 성립 필요
  - $\mathbf{d}$가 매우 작으므로 Δf 를 $\mathbf{x}^*$에서 테일러 시리즈로 근사

1계 필요조건first order necesssary condition

일변수 함수의 경우

- 최소점이 되기 위해 반드시 만족시켜야 하는 조건
- 최소점으로 가정한 점 x^*에서 테일러 시리즈

$$f(x) = f(x^*) + f'(x^*)\,d + \frac{1}{2}f''(x^*)\,d^2 + R$$

- $f(x^*)$를 이항하면 변화량 Δf는

$$\Delta f = f'(x^*)d + \frac{1}{2}f''(x^*)\,d^2 + R$$

- x^*가 최소라는 가정이 맞다면 Δf는 음수가 아님
- d는 매우 작은 증분이므로 Δf에서 $f'(x^*)d$가 지배항
- 그 뒤로는 제곱, 세제곱으로 급격히 작아진다.
- 따라서 Δf를 음이 아니게 하기 위해서 지배항 $f'(x^*)d$가 음이 아니어야 한다!
- d의 부호에 상관없이 Δf를 음이 아니게 하기 위해서
- **지역 최소를 위한 1계 필요조건**

$$f'(x^*) = 0 \tag{2}$$

- 필요조건을 만족시킨다면 그점은
  - 지역 최소
  - 지역 최대
  - 변곡점
- 이런 점을 통틀어 상점staionary point

다변수 함수의 경우

- 똑같은 논리 적용
- 테일러 시리즈

$$f(\mathbf{x}) = f(\mathbf{x}^*) + \nabla f(\mathbf{x}^*)^{\mathrm{T}}\mathbf{d} + \frac{1}{2}\mathbf{d}^{\mathrm{T}}\mathbf{H}\left(\mathbf{x}^*\right)\mathbf{d} + R$$

- 음이 되면 안되는 함수의 변화량

$$\Delta f = \nabla f(\mathbf{x}^*)^{\mathrm{T}}\mathbf{d} + \frac{1}{2}\mathbf{d}^{\mathrm{T}}\mathbf{H}\left(\mathbf{x}^*\right)\mathbf{d} + R$$

- 경사도벡터에 대한 필요조건

$$\nabla f\left(\mathbf{x}^*\right) = \mathbf{0} \tag{2*}$$

최적설계의 수치해법

- 만약 목적함수가 미분 가능이기는 하지만 미분 가능할 정도로 정식화되지 않았다면?
- 변수의 값에 따른 목적함수의 값만을 계산할 수 있는 상황이라면?

수치 알고리즘 일반개념

- 최적화 수치 알고리즘은 일반적으로 다음 단계를 따름
  - 단계 1. 타당성있는 출발점 $\mathbf{x}^{(0)}$ 추정, $k=0$
  - 단계 2. 탐색방향 $\mathbf{d}^{(k)}$를 계산
  - 단계 3. 수렴 검토
  - 단계 4. 양의 이동거리 α_k 계산
  - 단계 5. 새로운 설계 변수 계산 $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k\mathbf{d}^{(k)}$, $k=k+1$ 단계 2로
- 따라서 α_k와 $\mathbf{d}^{(k)}$ 계산이 중요

경사도 수치계산

- 목적함수를 미분할 수 있는 함수의 형태를 모를 때 함수값만으로 미분계수를 구함
- 이론적으로 극한을 정의를 동원하지만 컴퓨테이션에서는 그냥 종속변수의 작은 변화를 독립변수의 작은 변화로 나눈 것

전방차분법

$$\frac{\partial f}{\partial x_i} \approx \frac{f(x_1, \cdots, x_i + \Delta x_i, \cdots, x_n) - f(x_1, \cdots, x_i, \cdots, x_n)}{\Delta x_i}$$

후방차분법

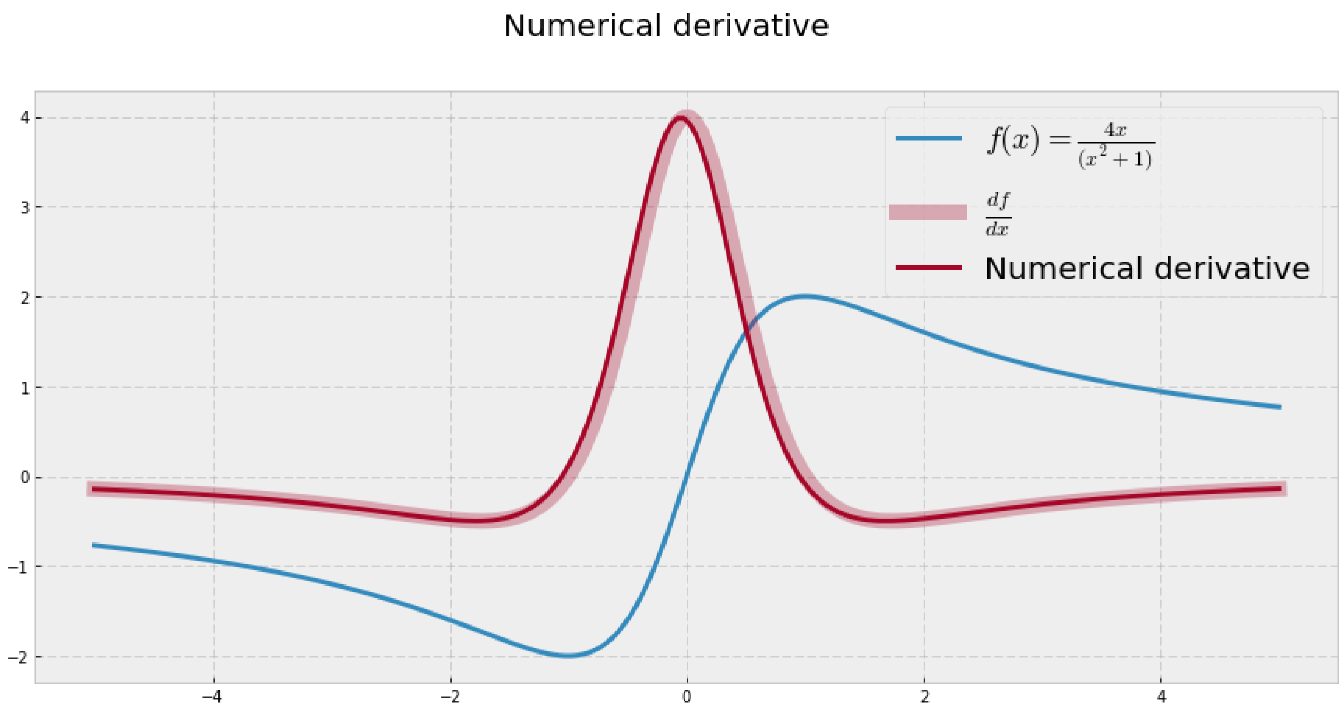
$$\frac{\partial f}{\partial x_i} \approx \frac{f(x_1, \cdots, x_i, \cdots, x_n) - f(x_1, \cdots, x_i + \Delta x_i, \cdots, x_n)}{\Delta x_i}$$

중앙차분법

$$\frac{\partial f}{\partial x_i} \approx \frac{f(x_1, \cdots, x_i + \frac{1}{2}\Delta x_i, \cdots, x_n) - f(x_1, \cdots, x_i - \frac{1}{2}\Delta x_i, \cdots, x_n)}{\Delta x_i}$$

경사도 수치계산 실습

$$f(x) = \frac{4x}{x^2 + 1}$$



In [5]:

```
x = sympy.Symbol('x')

sympy.simplify(sympy.diff((4*x)/(x**2 + 1), x))
```

Out[5]:

$$\frac{-4x^2 + 4}{x^4 + 2x^2 + 1}$$

In [6]:

```
plt.rcParams["figure.figsize"] = (15,7)

def f(x):
    """
    실습용 함수 정의
    """
    return 4*x / (x**2  + 1)

def df_anal(x):
    """
    sympy의 결과로 얻은 도함수
    """
    return (-4*x**2 + 4)  / (x**4 + 2*x**2 + 1)

def df_numer(x):
    """
    수치적으로 도함수의 값을 계산하는 함수
    중앙차분법으로 바꿔보세요.
    """
    h = 0.1
    return (f(x+h) - f(x)) / h

x = np.linspace(-5, 5, 200)

plt.plot(x, f(x), lw=3, color=style_colors[0], label=r"$f(x) = \frac{4x}{(x^2  + 1)}$")
plt.plot(x, df_anal(x), lw=10, color=style_colors[1], alpha=0.3 , label=r"$\frac{df}{dx}$")
plt.plot(x, df_numer(x), color=style_colors[1], lw=3, label=r"Numerical derivative")
plt.legend(fontsize=20)

plt.suptitle("Numerical derivative", fontsize=20)
plt.show()
```

강하단계

- 어떤 방향이 바람직한 $\mathbf{d}^{(k)}$ 방향인가?
- 반복의 목적은 목적함수 $f(\mathbf{x})$의 최소점에 도달하는 것
- 알고리즘의 k번째 반복에서 $\mathbf{x}^{(k)}$가 최소점이 아니라면 더 작은 목적함수값을 가지는 다른점 $\mathbf{x}^{(k+1)}$가 있다.

$$f\left(\mathbf{x}^{(k+1)}\right) < f\left(\mathbf{x}^{(k)}\right)$$

- 위 식 좌변에 단계 5의 식을 대입($\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)}$)

$$f\left(\mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)}\right) < f\left(\mathbf{x}^{(k)}\right) \qquad (*)$$

- $f\left(\mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)}\right)$는 $f\left(\mathbf{x}^{(k)}\right)$에서 $\alpha_k \mathbf{d}^{(k)}$만큼 변화한 것이므로 테일러 시리즈 전개

$$f(\mathbf{x}^* + \mathbf{d}) = f(\mathbf{x}^*) + \nabla f^{\mathrm{T}} \mathbf{d} + \frac{1}{2} \mathbf{d}^{\mathrm{T}} \mathbf{H} \mathbf{d} + R$$

- 위 식대로 맞춰서 전개하면 ($\mathbf{c}^{(k)}$는 경사도벡터)

$$f\left(\mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)}\right) = f\left(\mathbf{x}^{(k)}\right) + \mathbf{c}^{(k)} \cdot \left(\alpha_k \mathbf{d}^{(k)}\right) + R$$

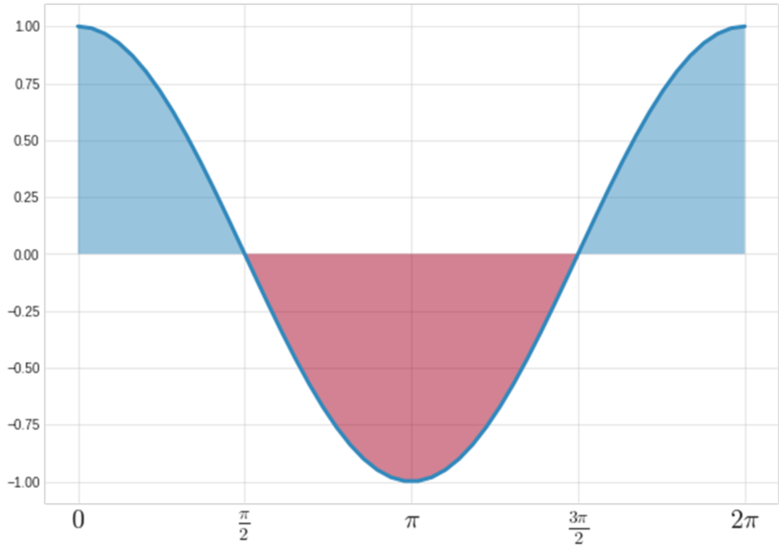
- 전개한 결과를 $(*)$에 대입

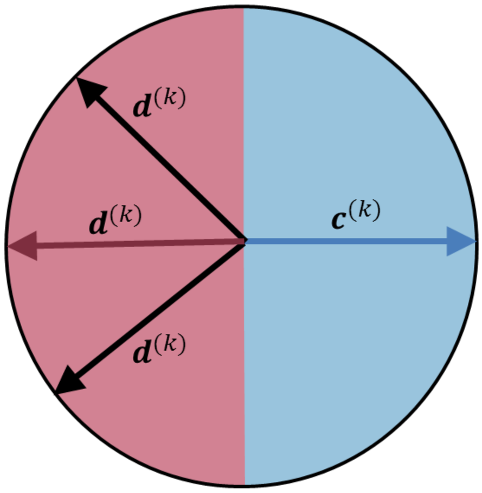
$$\begin{aligned} f\left(\mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)}\right) &= f\left(\mathbf{x}^{(k)}\right) + \mathbf{c}^{(k)} \cdot \left(\alpha_k \mathbf{d}^{(k)}\right) + R \\ &\approx f\left(\mathbf{x}^{(k)}\right) + \mathbf{c}^{(k)} \cdot \left(\alpha_k \mathbf{d}^{(k)}\right) < f\left(\mathbf{x}^{(k)}\right) \end{aligned}$$

- 따라서 최종적으로 다음과 같은 부등식을 얻는다.

$$\begin{aligned} \alpha_k \left(\mathbf{c}^{(k)} \cdot \mathbf{d}^{(k)}\right) &< 0 \\ \mathbf{c}^{(k)} \cdot \mathbf{d}^{(k)} &< 0 \quad \because \alpha_k > 0 \end{aligned}$$

- 위 부등식의 의미 : $\mathbf{x}^{(k)}$를 $\mathbf{x}^{(k+1)}$로 **개선시키기 위한 탐색방향 $\mathbf{d}^{(k)}$는 경사도벡터 $\mathbf{c}^{(k)}$와 90°에서 270° 사이에 있어야 한다.**
- 위 부등식으로 표시되는 강하조건descent condition을 만족하는 방향을 강하방향descent direction
- 강하방향에 기반한 방법을 **강하법** 이라 한다.





일차원 최소화

일변수 함수로 환원

- 설계 변화의 바람직한 방향 $\mathbf{d}^{(k)}$를 알 때 설계변수의 업데이트에 대해서 목적함수는

$$f\left(\mathbf{x}^{(k+1)}\right) = f\left(\mathbf{x}^{(k)} + \alpha\mathbf{d}^{(k)}\right)$$

- α만의 함수로 목적함수가 업데이트

$$f\left(\mathbf{x}^{(k+1)}\right) = f\left(\mathbf{x}^{(k)} + \alpha\mathbf{d}^{(k)}\right) \equiv \bar{f}\left(\alpha\right)$$

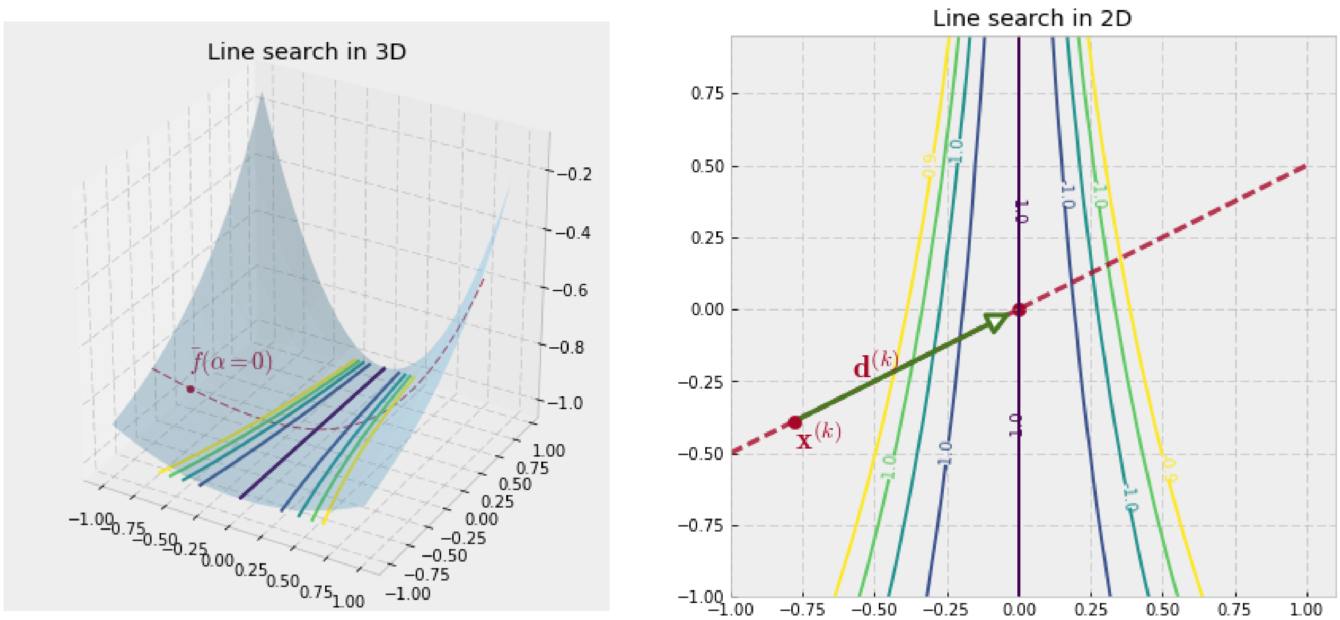
- 즉, 현재 위치 $\mathbf{x}^{(k)}$ 에서 $\mathbf{d}^{(k)}$방향으로 양수 α 만큼 나아가면서 목적함수 f를 줄여나간다.
- 1변수 함수에 대한 필요조건 $\dfrac{df}{d\alpha} = 0$을 도입하면 정확하게(해석적으로) 얼마나 나아갈지 결정할 수 있다.
- f는 $\mathbf{x}^{(k+1)}$ 의 함수인데 $\mathbf{x}^{(k+1)}$ 은 다시 α의 함수이므로 연쇄법칙을 적용

$$\frac{df\left(\mathbf{x}^{(k+1)}\right)}{d\alpha} = \frac{\partial f\left(\mathbf{x}^{(k+1)}\right)}{\partial\,\mathbf{x}}\frac{d\,\mathbf{x}^{(k+1)}}{d\alpha} = \nabla f\left(\mathbf{x}^{(k+1)}\right)\cdot\mathbf{d}^{(k)} = 0$$

- 따라서 강하방향으로의 이동거리는 아래 식처럼 경사도벡터와 강하방향의 내적이 0 다시말해 두 벡터가 수직이 될때까지 진행

$$\mathbf{c}^{(k+1)}\cdot\mathbf{d}^{(k)} = 0$$

- 다음 그림은 일차원 최소화 과정을 나타낸 것이다.

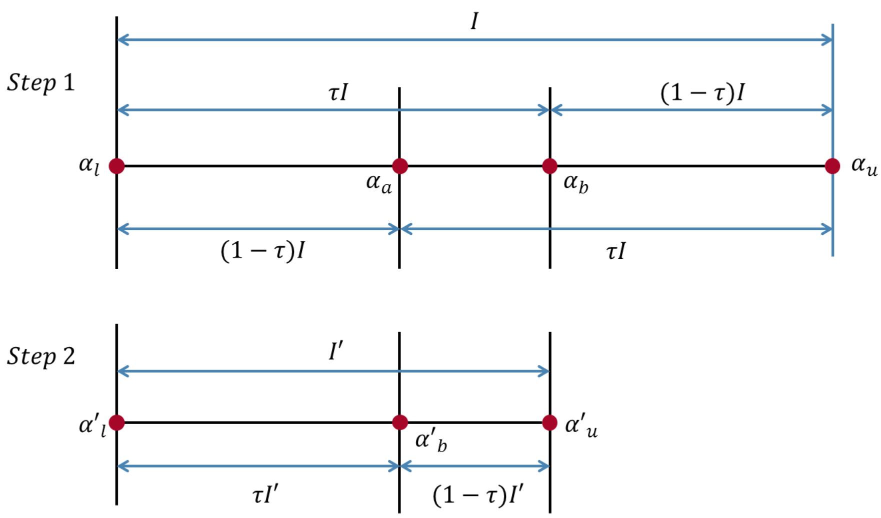


등간격 탐색

- 가장 쉽게 생각할 수 있는 방법 : $f(\alpha=0)$에서 α를 등간격 δ만큼 증가시키면서 함수값 계산
- 현재 함수값이 증가하면 전, 전 스탭 함수값을 좌측값, 현재 함수값을 우측값으로 설정하고 δ를 적당히 줄이고 다시 반복

황금분할탐색

- 최소값이 있을 것 같은 구간을 황금비율로 잘라 구간을 계속 좁혀나가는 방법
- 좁혀진 구간을 다시 좁힐 때 이전에 계산해둔 값을 재사용할 수 있어 계산량을 줄임



- 황금비 $\tau = \left(\sqrt{5}-1\right)/2 = 0.618$로 두고 탐색 구간 I 내부의 두 점을 $\tau I = 0.618I$와 $(1-\tau)I = 0.382I$ 로 잡고 구간 줄이기

In [8]:

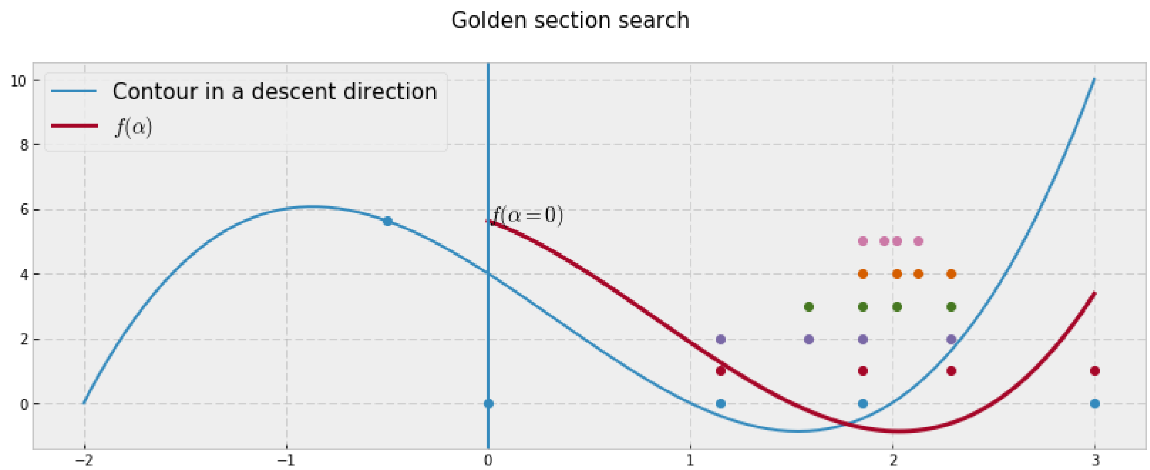
```
def gss(a, b, x, d,  tol=1e-12):
    '''
    https://en.wikipedia.org/wiki/Golden-section_search
    golden section search
    to find the minimum of f on [a,b]
    f: a strictly unimodal function on [a,b]
    a, b : search interval
    x :  current design variables
    d : descent direction
    '''
    gr = (np.sqrt(5) + 1) / 2

    l = b - (b - a) / gr
    r = a + (b - a) / gr

    while abs(l - r) > tol:
        if f_alpha(l, x, d) < f_alpha(r, x, d):
            b = r
        else:
            a = l

        # we recompute both c and d here to avoid loss of precision
        # which may lead to incorrect results or infinite loop
        l = b - (b - a) / gr
        r = a + (b - a) / gr

    return (b + a) / 2
```

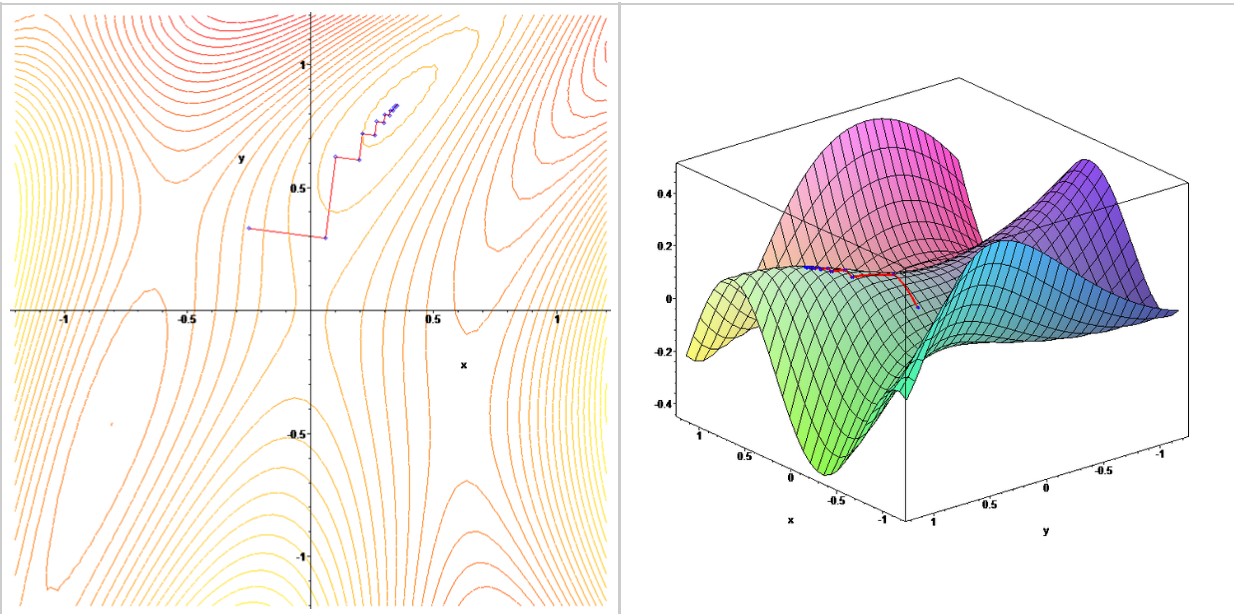



```
minimum at alpha= 2.035184E+00 , start at x=-0.5
```

최속강하법steepest descent method

- 최속강하법 알고리즘[1]

> Step 1. Estimate a starting design as $\mathbf{x}^{(0)}$. Set the iteration counter $k=0$. Select the convergence parameter ϵ.
>
> Step 2. Calculate the gradient of $f(\mathbf{x})$ at the current point $\mathbf{x}^{(k)}$ as $\mathbf{c}^{(k)} = \nabla f\left(\mathbf{x}^{(k)}\right)$.
>
> Step 3. Calculate the length of $\mathbf{c}^{(k)}$ as $\|\mathbf{c}^{(k)}\|$. If $\|\mathbf{c}^{(k)}\| < \epsilon$, then stop the iterative process because $\mathbf{x}^* = \mathbf{x}^{(k)}$ is a local minimum point. Otherwise, continue.
>
> Step 4. Let the search direction at the current point $x^{(k)}$ be $\mathbf{d}^{(k)} = -\mathbf{c}^{(k)}$.
>
> Step 5. Calculate a step size α_k that minimizes $f(\alpha) = f\left(\mathbf{x}^{(k)} + \alpha\mathbf{d}^{(k)}\right)$ in the direction $\mathbf{d}^{(k)}$. Any one-dimensional search algorithm may be used to determine α_k.
>
> Step 6. Change the design as follows: set $k=k+1$ and go to Step 2.
> $$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k\mathbf{d}^{(k)}$$



https://en.wikipedia.org/wiki/Gradient_descent : public domain

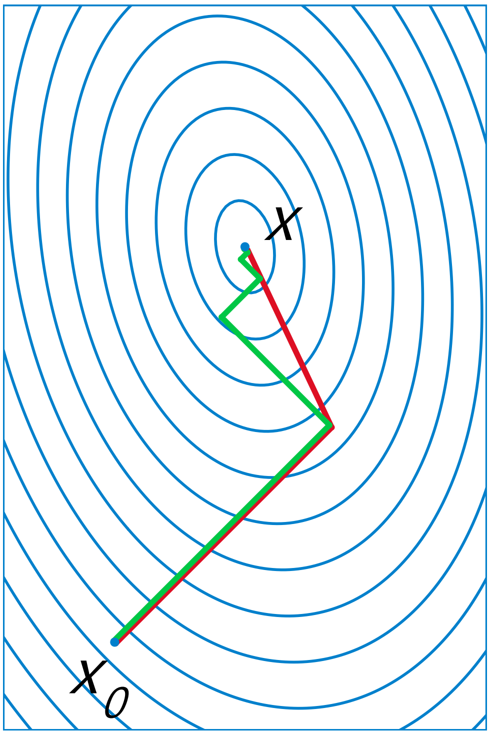
공액경사도법conjugate gradient method

- 공액경사도법 알고리즘[1]

> Step 1. Estimate a starting design as $\mathbf{x}^{(0)}$. Set the iteration counter $k=0$. Select the convergence parameter ϵ. Calculate
> $$\mathbf{d}^{(0)}=-\mathbf{c}^{(0)}=-\nabla f\left(\mathbf{x}^{(0)}\right)$$
> Step 2. Compute the gradient of the cost function as
> $$\mathbf{c}^{(k)}=\nabla f\left(\mathbf{x}^{(k)}\right)$$
> Step 3. Calculate $\|\mathbf{c}^{(k)}\|$. If $\|\mathbf{c}^{(k)}\|<\epsilon$, then stop; otherwise continue.
>
> Step 4. Calculate the new conjugate direction as
> $$\mathbf{d}^{(k)}=-\mathbf{c}^{(k)}+\beta_k\mathbf{d}^{(k-1)}$$
> $$\beta_k=\left(\frac{\|\mathbf{c}^{(k)}\|}{\|\mathbf{c}^{(k-1)}\|}\right)^2$$
>
> Step 5. Compute a step size $\alpha_k=\alpha$ to minimize $f(\alpha)=f\left(\mathbf{x}^{(k)}+\alpha\mathbf{d}^{(k)}\right)$.
>
> Step 6. Change the design as follows: set $k=k+1$ and go to Step 2.
> $$\mathbf{x}^{(k+1)}=\mathbf{x}^{(k)}+\alpha_k\mathbf{d}^{(k)}$$

- 공액방향은 강하조건을 만족
- 강하조건에 $\mathbf{d}^{(k)}$를 대입

$$\begin{aligned}\mathbf{c}^{(k)}\cdot\mathbf{d}^{(k)}&=\mathbf{c}^{(k)}\cdot\left(-\mathbf{c}^{(k)}+\beta_k\mathbf{d}^{(k-1)}\right)\\&=-\underbrace{\mathbf{c}^{(k)}\cdot\mathbf{c}^{(k)}}_{|\mathbf{c}^{(k)}|^2}+\beta_k\underbrace{\mathbf{c}^{(k)}\cdot\mathbf{d}^{(k-1)}}_{\text{step size cond. }=0}<0\end{aligned}$$



https://en.wikipedia.org/wiki/Conjugate_gradient_method : public domain

수치해법 적용 실습

- 다음 함수들에 대해서 최속강하법, 공액경사도법을 적용해 해를 구해본다.

$$f_1(x_1, x_2) = x_1^2 + x_2^2$$

$$f_2(x_1, x_2) = 50x_1^2 + x_2^2$$

$$f_3(x_1, x_2) = 50(x_2 - x_1^2)^2 + (2 - x_1)^2$$

In [10]:

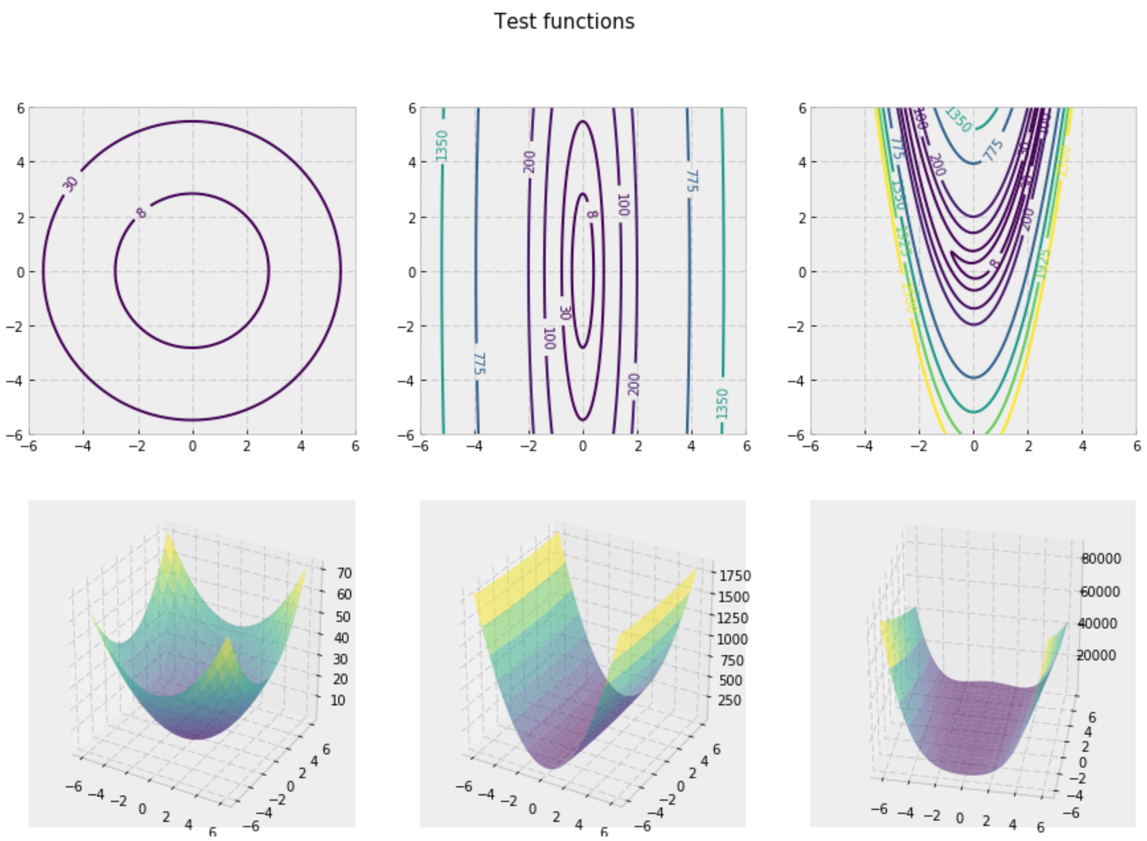
```python
def f_1(x) :
    """
    convex function
    """
    return  x[0]**2 + x[1]**2

def f_2(x):
    return  50*x[0]**2 + x[1]**2

def f_3(x) :
    """
    Rosenbrock function, example 11.8 from Jasbir Aroa
    """
    return 50*(x[1]-x[0]**2)**2 + (2-x[0])**2
    #return (1-x[0])**2 + 100*(x[1]-x[0]**2)**2

F = [f_1, f_2, f_3]
OPTIMS = [(0,0), (0,0),(2,4)]
```

- 각 함수들의 그림



코드 실습

- 주피터 노트북에 제공된 코드에 빈 부분을 채워서 완성해보세요.
- `def f_alpha(alpha, x, d)`를 완성하기 위한 수식

$$\bar{f}\,(\alpha) \equiv f\left(\mathbf{x}^{(k)} + \alpha\mathbf{d}^{(k)}\right)$$

- `def grad(x)`를 완성하기 위한 수식

$$\frac{\partial\, f}{\partial\, x_i} \approx \frac{f(x_1, \cdots, x_i + \frac{1}{2}\Delta x_i, \cdots, x_n) - f(x_1, \cdots, x_i - \frac{1}{2}\Delta x_i, \cdots, x_n)}{\Delta x_i}$$

- 강하방향 계산 부분을 완성하기 위한 수식
  - 최속강하법

$$\mathbf{d}^{(k)} = -\mathbf{c}^{(k)}$$

  - 공액경사법

$$\mathbf{d}^{(k)} = -\mathbf{c}^{(k)} + \beta_k\mathbf{d}^{(k-1)}$$

$$\beta_k = \left(\frac{\|\mathbf{c}^{(k)}\|}{\|\mathbf{c}^{(k-1)}\|}\right)^2$$

- 업데이트 부분을 완성하기 위한 수식

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k\mathbf{d}^{(k)}$$

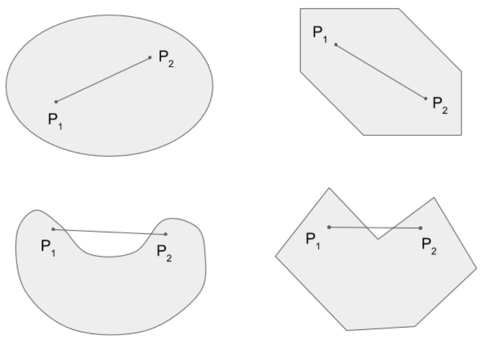
전역적 최적성

볼록집합convex set

- 어떤 집합 S내의 임의의 점 P_1, P_2를 잇는 선분 전체가 S내에 있다면 집합 S는 볼록집합
- 일반적으로 두점으로 정의되는 선분

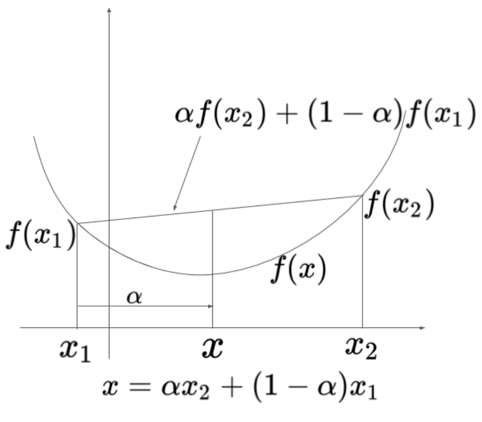
$$\mathbf{x} = \alpha \mathbf{x}^{(2)} + (1-\alpha)\mathbf{x}^{(1)}$$

- 위 선분 전체가 집합 S에 속하면 볼록집합



볼록함수convex function

- 볼록집합 S상에 정의된 n변수의 함수 $f(\mathbf{x})$는 집합 S내의 모든 점에서 함수의 헤시안이 양정 또는 양반정이면 볼록함수
- 1차원 문제에서는 2계 도함수(2번 미분)가 음이 아님



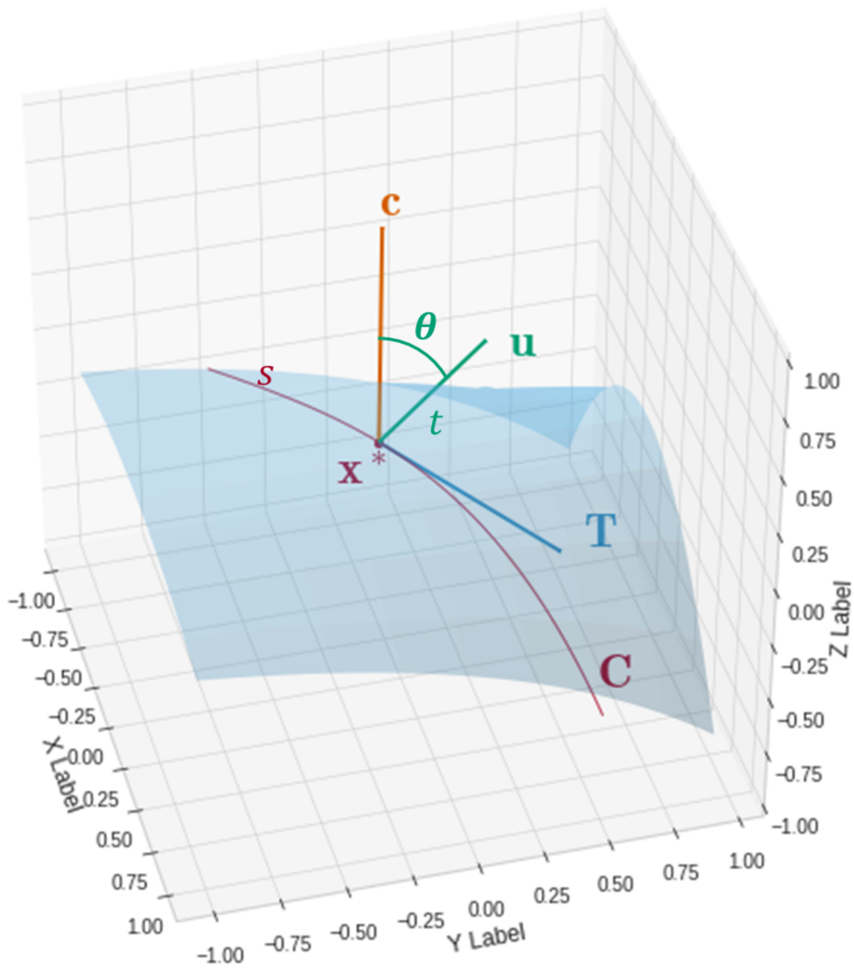
볼록계획문제convex programming problem

- 볼록집합에서 정의된 목적함수가 볼록함수 $\Longrightarrow$ 볼록계획문제
- 볼록계획문제에서는 지역최소는 곧 전역최소

부록

1. 경사도 벡터는 함수표면의 접면에 수직

- 여기서 접면은 개념적 표현
- 2변수 실함수라면 접선이되고, 3변수 실함수라면 접면, 4변수 이상 실함수라면 초 접평면hyper tangent plane 이 된다.
- 아래 그림은 3변수 음함수를 나타낸것



[증명]

- 위 그림처럼 $f(\mathbf{x}) = \text{constant}$인 곡면 위를 지나고 $\mathbf{x}^*$를 통과하는 임의의 곡선 $\mathbf{C}$에 대해서
- 곡선 $\mathbf{C}$가 s에 대해서 매개변수화되었다 하자.
- 즉 곡선 $\mathbf{C} = [\, f_1(s) \quad f_2(s) \quad f_3(s)\,]^{\mathrm{T}}$ 인 "일변수 벡터함수"이다.
- 이렇게 정의된 곡선 $\mathbf{C}$의 $\mathbf{x}^*$에서 접선벡터는 각 성분을 s에 대해 미분한 것이므로

$$\mathbf{T} = \begin{bmatrix} \dfrac{\partial\, x_1}{\partial\, s} & \dfrac{\partial\, x_2}{\partial\, s} & \cdots & \dfrac{\partial\, x_n}{\partial\, s} \end{bmatrix}^{\mathrm{T}}$$

- 이제 매개변수 s가 미세하게 변할 때 곡면 함수 $f(\mathbf{x})$의 변화를 생각하자.
- 즉, 곡면 함수 $f(\mathbf{x})$를 s로 미분하는 것이다.
- 그런데 s가 변해감에 따라 곡면 함수 $f(\mathbf{x})$의 값은 변하지 않고 constant로 일정하므로

$$\frac{\partial f}{\partial s} = 0$$

- 미분의 연쇄법칙을 적용하면

$$\frac{\partial f}{\partial s} = \frac{\partial f}{\partial x_1}\frac{\partial x_1}{\partial s} + \frac{\partial f}{\partial x_2}\frac{\partial x_2}{\partial s} + \cdots + \frac{\partial f}{\partial x_n}\frac{\partial x_n}{\partial s} = 0$$

- 위 식을 파란색 부분과 주황색 부분을 따로 모아 벡터형식으로 쓰면

$$\frac{\partial f}{\partial s} = \begin{bmatrix} \dfrac{\partial f}{\partial x_1} & \dfrac{\partial f}{\partial x_2} & \cdots & \dfrac{\partial f}{\partial x_n} \end{bmatrix}^{\mathrm{T}} \cdot \begin{bmatrix} \dfrac{\partial x_1}{\partial s} & \dfrac{\partial x_2}{\partial s} & \cdots & \dfrac{\partial x_n}{\partial s} \end{bmatrix} = 0$$

- 결국 파란부분은 경사도벡터, 주황색 부분은 접선벡터 $\mathbf{T}$가 되고

$$\mathbf{c} \cdot \mathbf{T} = 0$$

- 경사도벡터 $\mathbf{c}$는 접선과의 내적이 0이므로 접선에 수직이 된다.

2. 경사도벡터의 방향은 함수의 최대증가방향

[증명]

- 이번에는 $\mathbf{x}^*$ 에서 접선 방향이 아닌 방향으로의 단위벡터를 $\mathbf{u}$라 하자.
- $\mathbf{x}^*$ 에서 $\mathbf{u}$방향으로의 변화는 $\mathbf{x}^* + t\,\mathbf{u}$로 표시 가능
- 이제 점 $\mathbf{x}^*$ 에서 $\mathbf{u}$방향으로 t에 대한 미분을 생각해 볼 수 있는데
- 이러한 미분을 방향미분이라 한다. 미분의 정의대로 쓰면

$$\nabla_{\mathbf{u}} f = \lim_{t \to 0} \frac{f(\mathbf{x}^* + t\,\mathbf{u}) - f(\mathbf{x}^*)}{t} \tag{*}$$

- 분자의 $f(\mathbf{x}^* + t\,\mathbf{u})$을 테일러 시리즈 전개하면

$$f(\mathbf{x}^* + t\mathbf{u}) = f(\mathbf{x}^*) + t\left[u_1 \frac{\partial f}{\partial x_1} + u_2 \frac{\partial f}{\partial x_2} + \cdots + u_n \frac{\partial f}{\partial x_n}\right] + O(t^2)$$

- 시그마 기호를 도입하고 양변에 $f(\mathbf{x}^*)$를 빼면

$$f(\mathbf{x}^* + t\mathbf{u}) - f(\mathbf{x}^*) = t \sum_{i=1}^{n} u_i \frac{\partial f}{\partial x_i} + O(t^2)$$

- 위 결과를 $(*)$ 대입하면

$$\begin{aligned}\nabla_{\mathbf{u}} f &= \lim_{t \to 0} \frac{t \sum_{i=1}^{n} u_i \frac{\partial f}{\partial x_i} + O(t^2)}{t} \\ &= \sum_{i=1}^{n} u_i \frac{\partial f}{\partial x_i} = \mathbf{c} \cdot \mathbf{u}\end{aligned}$$

- 즉, 함수 $f(\mathbf{x})$의 $\mathbf{u}$방향으로의 변화율은 경사도벡터 $\mathbf{c}$와 단위벡터 $\mathbf{u}$의 내적
- 내적의 정의에 의해

$$\mathbf{c} \cdot \mathbf{u} = |\mathbf{c}||\mathbf{u}| \cos\theta$$

- 위 변화율이 최대가 되려면 θ가 0° 또는 180°
- 따라서 경사도벡터 방향으로 변화할때 변화율이 가장 크다. 그 변화율은 경사도벡터의 크기이다. ($\because |\mathbf{u}| = 1, \quad \cos\theta = 1$)

참고문헌

1. 최적설계입문Introduction to Optimum Design, Jasbir S. Arora
2. http://jjycjnmath.tistory.com/32?category=738760 (http://jjycjnmath.tistory.com/32?category=738760)