수치해석 중간대비 정리.

중간고사 → 코딩문제 있다 (빈칸채우기), 컨닝페이퍼 1장 제작, 공학용계산기 지참, 중간 이유는 프로펠e.
(2, 3장 + 뒤에조금) ↘ 이코드를 실행시 a, b 변수가 어떻게되는가? (A4 한면 자필)
↳ 예제를 그대로 안내질지?!

*코딩 관련 복습 (MATLAB)

- 사칙연산자 * : 행렬곱
  .* : 원소곱
  ./ : 원소나눗셈
  .\ : 원소 역나눗셈
  .^ : 원소제곱
- 관계연산자 ~= : 같지않으면
  == : 같으면
  <= : 작거나같으면
  >= : 크거나같으면
- 논리 연산자 &&, || : and, or (읽기)
  &, | : and, or (쓰기)
- 치환 A=1;
- 주석 %
- Print disp('')
  fprintf("%d는 \n", val)
- 자리수표현 format short (e)
  format long (e)

```
*if 조건식
  실행문1;
elseif
  실행문2;
else
  실행문3;
end
```

```
*for 변수= 시작점:간격:끝점
  반복실행할 내용;
end
```

```
for i=1:10
  i =i+2
end
```
→ 어케되나?

```
*while 조건문
  반복실행할 내용;
    if 조건
      break;
    end
end
```

함수
① [출력변수들] = function name (입력변수들)
end
② 저장! ③ z = fun (3,7) 로 호출

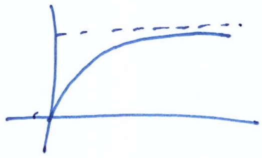
- nargin : 함수를 구성하는 인수의 개수
- a:b:c ⇒ a부터 b간격으로 c까지. b를 생략시 b=1
- linspace (a,b,c) ⇒ a이상 b이하인 범위에서 c개의 항 생성
- 문자열의 길이 : size(A)
- A(□, △) ⇒ □행 △열의 원소 의미.
  ex) A(:,2:5) ⇒ 모든행에 대하여 2번째 열부터 5번째 열까지의 원소 의미.
  < 파이썬에서는 이상, 미만 / 매트랩 에서는 이상, 이하
- (sin(x))^2 ⇒ o
  sin^2(x), sin^2x, sin(x^2), sin(x)^2 ⇒ x

1장.

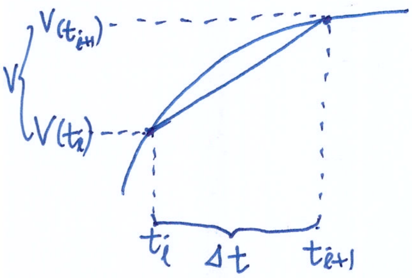
- 번지점프하는 사람에 대한 운동방정식

$$\frac{dv}{dt} = g - \frac{c_d}{m} v^2$$

if. $t=0$에서 $v=0$이면 $\Rightarrow v(t) = \sqrt{\frac{gm}{c_d}} \tanh\left(\sqrt{\frac{gc_d}{m}}\, t\right)$



$\Rightarrow$ 정확한 해석해 대신에 근사적으로 수치해를 구하는건이 필요하다!



$$\frac{\Delta v}{\Delta t} = \frac{v(t_{i+1}) - v(t_i)}{t_{i+1} - t_i} \approx \frac{dv}{dt}$$

$$\Rightarrow \frac{v(t_{i+1}) - v(t_i)}{t_{i+1} - t_i} = g - \frac{c_d}{m} v^2$$

$$\Rightarrow v(t_{i+1}) = v(t_i) + \left(g - \frac{c_d}{m} v^2\right)(t_{i+1} - t_i)$$

new = old + slope × step

2장 MATLAB 기초

3장 MATLAB 프로그래밍

- function v = freefallvel (t,m,cd)

```
function v = freefallvel(t,m,cd)
g=9.81;
v=sqrt(g*m/cd)*tanh(sqrt(g*cd/m)*t);
end
```

- if문, for문, while문

4장. 반올림오차와 절단오차

- 수치해석에서는 여러 오차 중 수치오차를 다룸 < 반올림오차, 절단오차
- 오차의 정의

	절대오차	상대오차(%)
<참값을 알때> 참오차 (True ...)	E_t = 참값 − 근사값	$\varepsilon_t = \frac{\text{참값} - \text{근사값}}{\text{참값}}$
<참값을 모를때> 근사오차 (Approxi...)	E_a = 현재근사값 − 이전근사값	$\varepsilon_a = \frac{\text{현재근사값} - \text{이전근사값}}{\text{현재근사값}}$

※ ε_s (백분율 허용치) 컨닝페이퍼에적기

$= (0.5 \times 10^{2-n})\%$

적어도 n개의 유효숫자 내 정확

4장 PPt 36page 코드도 적기!

① 반올림 오차 : 수를 정확하게 처리할 수 없기 때문에 발생

*컴퓨터상에서의 수의 표현

realmax = 1.797 ···· e+308

realmin = 2.225 ···· e-308

eps(machine epsilon) = 2.2204··· e-016 (이거보다 작은 수 표현x)

⇒ 반올림오차를 발생시킬 우려가 있다.

⇒ 뺄셈의 무효화, 큰수와 작은 수의 덧셈등.

② 절단오차 : 수학적 연산을 근사적으로 표현하여 발생

⇒ Taylor 급수!

0차 근사 $f(x_{i+1}) \cong f(x_i)$

1차 근사 $f(x_{i+1}) \cong f(x_i) + \underline{f'(x_i)h}$

2차근사 $f(x_{i+1}) \cong f(x_i) + \underline{f'(x_i)h} + \underline{\frac{f''(x_i)}{2!}h^2}$ ✓

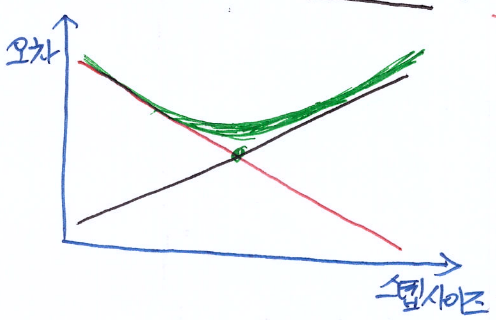
ex) 번지 점프 예제에서 Taylor급수 전개

$$v(t_{i+1}) = v(t_i) + v'(t_i)(t_{i+1} - t_i) + \underbrace{\frac{v''(t_i)}{2!}(t_{i+1} - t_i)^2 + \cdots + R_n}_{\text{절단오차로 날라감.(급수를 절단)}}$$

※ 전향차분, 후향차분, 중심차분.

$\frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} + \underline{O(h)}$ $\frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}} + \underline{O(h)}$ $\frac{f(x_{i+1}) - f(x_{i-1})}{x_{i+1} - x_{i-1}} + \underline{O(h^2)}$

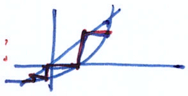
③ 전체 수치오차 = 절단오차 + 반올림오차



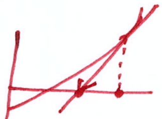
5장. 근 : 구간법

- 근을 구하는 방법 : 그래프적 방법 $\xrightarrow{\text{정밀X}}$ 시행착오법 $\xrightarrow{\text{비효율적}}$ 수치기법
  (f(x)와 x축만나는점) (x의값을가정하여 Try and error) (시행을체계적으로!)

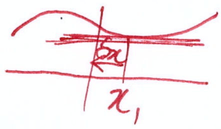
- 초기 가정법
  - 1) 구간법 : 양끝값초기가정, 수렴느림(-), 항상근찾음(+)
    - ① 증분탐색법 : 전체구간 N등분후 각구간의 끝값의부호로 판단
    - ☆② 이분법 : 구간폭을 항상 반으로 나누는 방법, $|\varepsilon_a| = \left|\frac{x_r^{new} - x_r^{old}}{x_r^{new}}\right| 100\% < \varepsilon_s$ 에서 종료☆
    - ③ 가위치법(선형보간법) : 두점의 x절편으로 구간을 나눔. $x_r = x_u - \frac{f(x_u)(x_l - x_u)}{f(x_l) - f(x_u)}$
  - 2) 개방법 : x값초기가정, 수렴빠름(+), 근을못찾을수도(-)
    - ① 단순고정점 반복법 :



    - ② Newton-Raphson 법



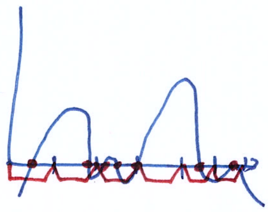
    - ③ 할선법 (+수정된 할선법)



1)-① 증분탐색법

1. 근이 존재하는 구간이 주어짐. 그걸 N등분.
2. $f(x_l) \cdot f(x_u) < 0$ 이면 x_l과 x_u 사이에 적어도 하나이상의 실근 존재.

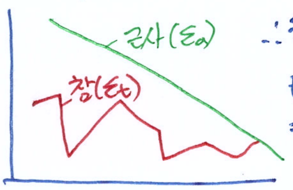
⇒ 단점 : N이↓ … 근을 놓치게됨
N이↑ … 계산시간↑
중근을 놓칠위험.



1)-②☆ 이분법

반복	x_l	x_u	x_r	$\lvert\varepsilon_a\rvert$	$\lvert\varepsilon_t\rvert$ (참값:142.7376)
0	50	200	125	x	12.43%
1	125	200	162.5	23.08	13.85%
2	125	162.5	143.75	13.04	0.71
3	125	143.75	134.375	6.98	5.86
4	134.375	143.75	139.0625	3.37	2.58
5	139.0625	143.75	141.4063	1.66	0.93
6	141.4063	143.75	142.5781	0.82	0.11
7	142.5781	143.75	143.1641	0.41	0.30

이분법의 오차



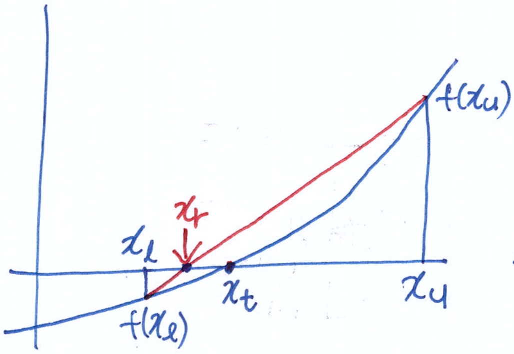
∴ 근사오차를쓸.
ε_a가 높이씩 줄어든다.
⇒ 예측할수있음!

※ 이분법 코드예시.

```
iter=0;
xr=xl;
while (1)
  xrold=xr;
  xr=(xl+xu)/2;
  iter=iter+1;
  disp(iter); disp(xr)
  if xr~=0
    ea=abs((xr-xrold)/xr)*100;
  end
test=feval(func,xl)*feval(func,xr);
  if test<0
    xu=xr;
  elseif test>0
    xl=xr;
  else
    ea=0;
  end
=> if ea<=es | iter>=maxit
     break, end
end ... while문끝
```

1) - ③ 가위치법

: $f(x_l)$과 $f(x_u)$를 연결하는 직선과 x축의 교점으로 새로운 근 도출.

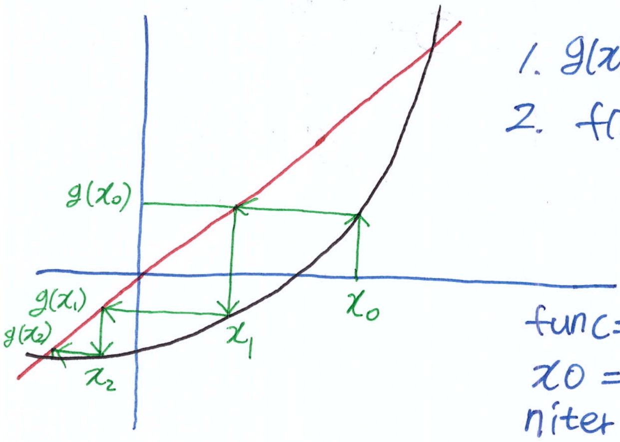


☆ $$\boxed{x_r = x_u - \frac{f(x_u)(x_l - x_u)}{f(x_l) - f(x_u)}}$$

⇒ 특정 상황에서는 이분법보다 느리다.

6장. 근: 개방법

2) - ① 단순고정점 반복법



1. $g(x) = f(x) + x$ 라는 임의의 $g(x)$를 만듦.
2. $f(x)=0 \Rightarrow g(x) - x = 0 \Rightarrow \underline{g(x) = x}$

$\therefore x_{i+1} = g(x_i)$

<코드>

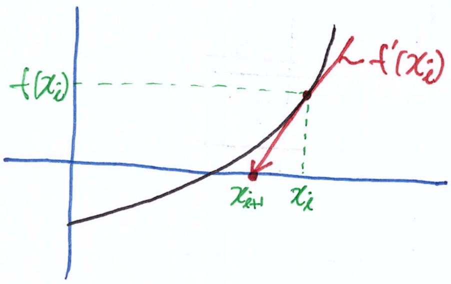
```
func=@(x) exp(-x)-x;
x0 = 0;
niter=10;
xi = x0;
xt =0.56714329;
err_t_old = 100;

for i = 1:niter
  xii = func(xi)+xi;
  err_a = (xii-xi)/xii;
  err_t = (xr-xii)/xr;
  xi = xii
  fprintf ...
end
```

장) 오차예상가능

단) $|g'(x)| < 1$ 일때만 수렴..

2)☆ - ② Newton-Raphson 법



$$f'(x_i) = \frac{f(x_i) - 0}{x_i - x_{i+1}} \Rightarrow ☆ \boxed{x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}}$$

<코드 中>

```
while (1)
  xrold = xr;
  xr = xr - func(xr)/dfunc(xr);
  iter = iter+1;
     ⋮
end
```

2)-③ 할선법

: N-R법에서 미분을 못구하면?

$\Rightarrow f'(x_i) \cong \dfrac{f(x_{i-1})-f(x_i)}{x_{i-1}-x_i}$ $\Rightarrow \boxed{x_{i+1} = x_i - \dfrac{f(x_i)}{f'(x_i)}}$ (N-R법)

$\boxed{\therefore x_{i+1} = x_i - \dfrac{f(x_i)(x_{i-1}-x_i)}{f(x_{i-1})-f(x_i)}}$

※ 수정된 할선법

$\Rightarrow \boxed{f'(x_i) \cong \dfrac{f(x_i+\delta x_i)-f(x_i)}{\delta x_i}}$ $\boxed{\therefore x_{i+1} = x_i - \dfrac{f(x_i)\,\delta x_i}{f(x_i+\delta x_i)-f(x_i)}}$

8장. 선형대수 방정식과 행렬.

- 선형대수 방정식의 행렬형태 표현 : $[A]\{x\} = \{b\}$

$[A]$ ↓ 계수행렬, $\{x\}$ ↓ 미지수열벡터, $\{b\}$ ↓ 상수열벡터

- 계산법 $\Rightarrow \{x\} = [A]^{-1}\{b\}$ → 가장 효율적인 것은 아님!

∴ ① Gauss 소거법(9장), ② LU분해(인수)법, ③ 반복법 을 배울것.

9장 ① Gauss 소거법

- 소규모 연립 방정식 ($n \le 3$)을 푸는법

① 도식적 방법 ② Cramer 공식 ③ 미지수소거법

① 도식적 방법

(장) 직관적

(단) 비효율적 (해가없거나 (평행), 무수히 많을수도 (일치))

② Cramer 공식

$[A]\{x\} = \{b\}$

$$D = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = a_{11}\begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} - a_{12}\begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} + a_{13}\begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix}$$

(디터미넌트)

$$= a_{11}(a_{22}a_{33} - a_{23}a_{32}) - a_{12}(a_{21}a_{33} - a_{23}a_{31}) + a_{13}(a_{21}a_{32} - a_{22}a_{31})$$

- Cramer공식 해

$$x_1 = \frac{\begin{vmatrix} b_1 & a_{12} & a_{13} \\ b_2 & a_{22} & a_{23} \\ b_3 & a_{32} & a_{33} \end{vmatrix}}{D}, \quad x_2 = \frac{\begin{vmatrix} a_{11} & b_1 & a_{13} \\ a_{12} & b_2 & a_{23} \\ a_{13} & b_3 & a_{33} \end{vmatrix}}{D}, \quad x_3 = \frac{\begin{vmatrix} a_{11} & a_{12} & b_1 \\ a_{12} & a_{22} & b_2 \\ a_{13} & a_{32} & b_3 \end{vmatrix}}{D}$$

③ 미지수소거법

1) 상수를 곱함
2) 뺄셈으로 x_1소거
3) x_2 결과 대입
4) $x_1 = \dfrac{a_{22}b_1 - a_{12}b_2}{a_{11}a_{22} - a_{21}a_{12}}$

* Gauss소거법

1) 전진소거 2) 후진대입.

1) 전진소거

(1) 첫 식①에 $\times a_{21}/a_{11}$ ⇒ $①'_2 = ① \times \dfrac{a_{21}}{a_{11}}$

(2) ②에서 $①'_2$을 뺌 ⇒ $②' = ② - ①'_2$

(3) ① 에 $\times a_{31}/a_{11}$ ⇒ $①'_3 = ① \times \dfrac{a_{31}}{a_{11}}$

(4) ③에서 $①'_3$을 뺌 ⇒ $③' = ③ - ①'_3$ ⇒ ①식 ②'식 ③'식

(5) ②'에 $\times a'_{32}/a'_{22}$ ⇒ $②''_3 = ②' \times \dfrac{a'_{32}}{a'_{22}}$

(6) ③'에서 $②''_3$을 뺌 ⇒ $③'' = ③' - ②''_3$ ⇒ ①식 ②'식 ③''식

2) 후진대입 ⇒ 간단.

- 가우스소거법 단점 : 0으로 나누는 나눗셈발생

⇒ 부분 피봇팅으로 행의 순서를 바꿔 해결!