Introduction of python & combustion instability

2019.12.17

김보연

Contents

1. Introduction of python
2. Basic grammar of python
3. Data visualization with python
4. Data analysis with python
5. Introduction of combustion instability
6. Next plan

Introduction

- Python은 1991년 프로그래머인 귀도 반 로섬(Guido van Rossum)이 발표한 고급 프로그래밍 언어이다.
- 특징
  - 인터프리터 언어, 한번에 한줄씩 실행
  - 무료로 사용가능
  - C, C++, JAVA 등 다른 프로그래밍 언어와 쉽게 통합할 수 있다.
  - 교차 플랫폼 언어, 윈도우, 리눅스 등 다양한 플랫폼에서 쓸 수 있다
  - 빠른개발속도
  - 간결한 코드

```c
#include <stdio.h>
int main(){
    printf("Hello world");
    return 0;
}
```

C

```python
print("Hello  world")
```

Python

Introduction

Feb 2019	Feb 2018	Change	Programming Language	Ratings	Change
1	1		Java	15.876%	+0.89%
2	2		C	12.424%	+0.57%
3	4	^	Python	7.574%	+2.41%
4	3	v	C++	7.444%	+1.72%
5	6	^	Visual Basic .NET	7.095%	+3.02%
6	8	^	JavaScript	2.848%	-0.32%
7	5	v	C#	2.846%	-1.61%
8	7	v	PHP	2.271%	-1.15%
9	11	^	SQL	1.900%	-0.46%
10	20	^^	Objective-C	1.447%	+0.32%
11	15	^^	Assembly language	1.377%	-0.46%
12	19	^^	MATLAB	1.196%	-0.03%
13	17	^^	Perl	1.102%	-0.66%
14	9	vv	Delphi/Object Pascal	1.066%	-1.52%
15	13	v	R	1.043%	-1.04%
16	10	vv	Ruby	1.037%	-1.50%
17	12	vv	Visual Basic	0.991%	-1.19%
18	18		Go	0.960%	-0.46%
19	49	^^	Groovy	0.936%	+0.75%
20	16	vv	Swift	0.918%	-0.88%

- 파이썬은 많은 제품이나 기업 및 연구기관에서 쓰이고 있다. 대표적인 예는 다음과 같다.
- **파이썬으로 작성된 자유-오픈 소스 소프트웨어**
  - 아나콘다(Anaconda)
  - 비트토렌트(BitTorrent)
  - 장고(웹 프레임워크)
  - 드롭박스(Dropbox)
- **파이썬을 내부적으로 사용하는 소프트웨어**
  - softimage|xsi (3D 애니메이션 소프트웨어)
  - 문명 IV
  - 오토데스크 마야 (3D 애니메이션 소프트웨어)
  - TORRENT(공유 프로그램)
  - 카카오톡(모바일/PC 메신저)
- **파이썬을 이용하고 있는 기업, 정부 기관**
  - 야후
  - 구글
  - 카카오
  - 미국항공우주국(NASA)

Basic grammar

Number

```
# 기본적인 사칙연산
print(5 + 6)   # 11
print(5 - 2)   # 3
print(3 * 8)   # 24
print(3 ** 3)  # 27 제곱
print(8 / 2)   # 4.0 float형
print(8 // 2)  # 4 int형
print(8 % 3)   # 2 나머지
```

String

```
test = "Hello World!"
print(test)  # Hello World!

test = 'Hello!'
print(test)  # Hello!

test = 'I don\'t need Coke!'
print(test)  # I don't need Coke!

test = "I don't need Coke!"
print(test)  # I don't need Coke!

first = 'Myungseo'
last = 'Kang'

print(first + last)  # Myungseo Kang
print(last * 5)      # KangKangKangKangKang
```

Slicing string

```
test_str = 'Leopold'

print(test_str[0])   # L
print(test_str[1])   # e
print(test_str[-1])  # d
print(test_str[-2])  # l

print(test_str[2:5])  # opo
print(test_str[3:6])  # pol
print(test_str[:5])   # Leopo
print(test_str[3:])   # pold
```

Basic grammar

If, elif, else

```
name = 'Leopold'

if name is 'Myungseo':
    print('Hello Myungseo')
elif name is 'Kang':
    print('Hello Kang!')
else:
    print('Hello Everyone!')
```

List

```
a = [] # a = list()와 동일
b = [1, 3, 5]
c = ['Leopold', 'Myungseo', 'Kang', 'L3opold7']
d = [7, 9, ['Myungseo', 'L3opold7']]

print(b[-1])     # 5
print(c[-2])     # Kang
print(d[-1][0])  # Myungseo

# List 값 수정
test = [1, 2, 3, 4, 5]
test[3] = 6

print(test)  # [1, 2, 3, 6, 5]
```

Tuple

```
tp1 = (1, 2, 3)
tp2 = (4, 5, 6)

print(tp1[2])     # 3
print(tp1[1:])    # (2, 3)
print(tp1 + tp2)  # (1, 2, 3, 4, 5, 6)
print(tp2 * 2)    # (4, 5, 6, 4, 5, 6)
```

Basic grammar

Dictionary

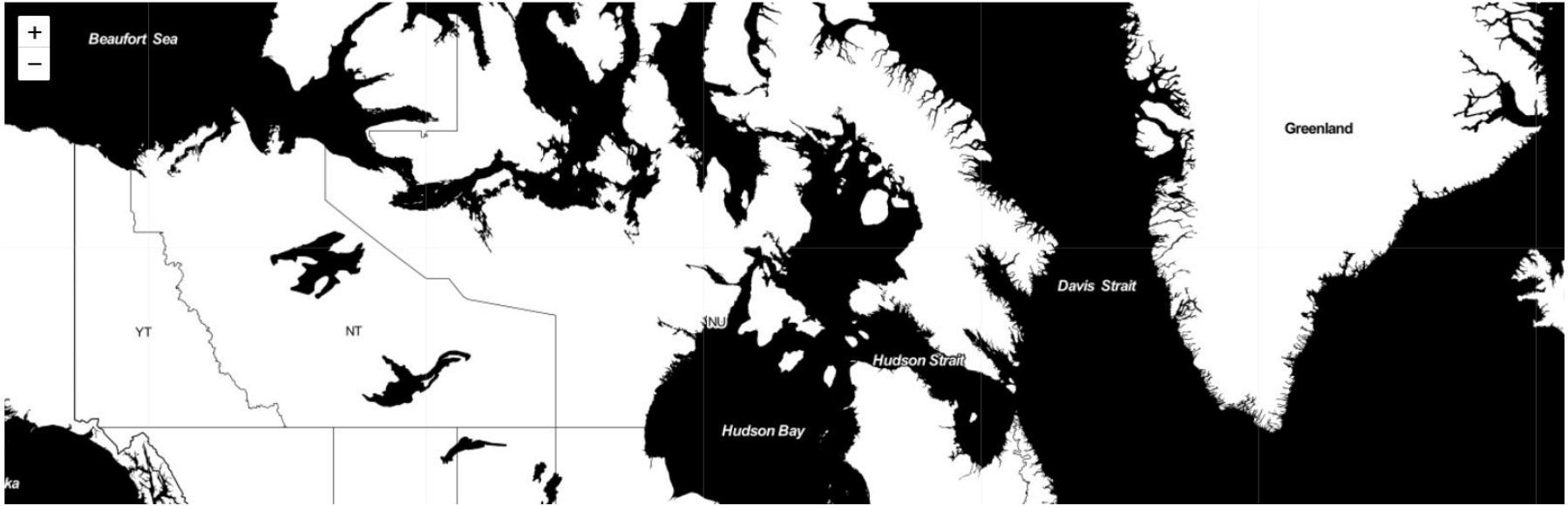
```
dic1 = dict()
dic2 = {'k1': 'v1', 'k2': 'v2', 'k3': 'v3'}
dic3 = dict([('name', 'L3opold7'), ('phone', '010-1234-5678')])
dic4 = dict(firstname='Myungseo', lastname='Kang')
dic5 = {'ls': ['a', 'b', 'c']}

print(dic2)               # {'k1': 'v1', 'k3': 'v3', 'k2': 'v2'}
print(dic2['k2'])         # v2
print(dic3)               # {'phone': '010-1234-5678', 'name': 'L3opold7'}
print(dic3['name'])       # L3opold7
print(dic4)               # {'firstname': 'Myungseo', 'lastname': 'Kang'}
print(dic4['firstname'])  # Myungseo
print(dic5['ls'])         # ['a', 'b', 'c']
```

Set

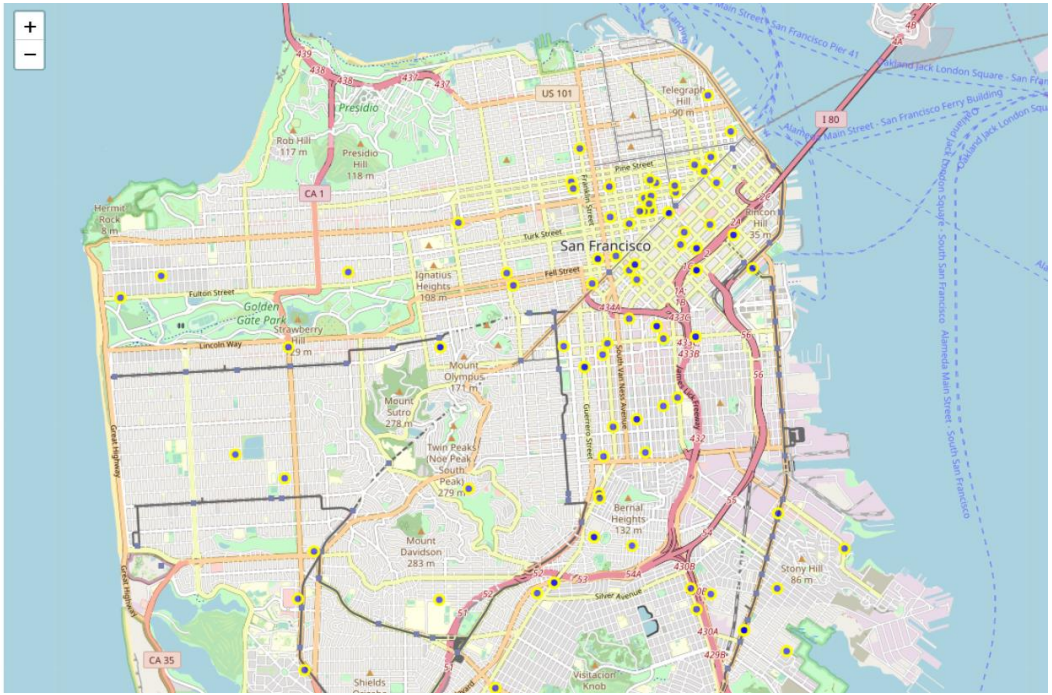
```
s = set([1, 2, 3, 4, 5])
print(s)  # {1, 2, 3, 4, 5}

hello = set('Hello World!')
print(hello)  # {' ', 'H', '!', 'e', 'l', 'o', 'd', 'W', 'r'}

set1 = set([1, 2, 3, 4, 5, 6])
set2 = set([5, 6, 7, 8, 9, 0])

print(set1 & set2)  # {5, 6}
print(set1 | set2)  # {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
print(set1 - set2)  # {1, 2, 3, 4}
print(set2 - set1)  # {0, 8, 9, 7}
```

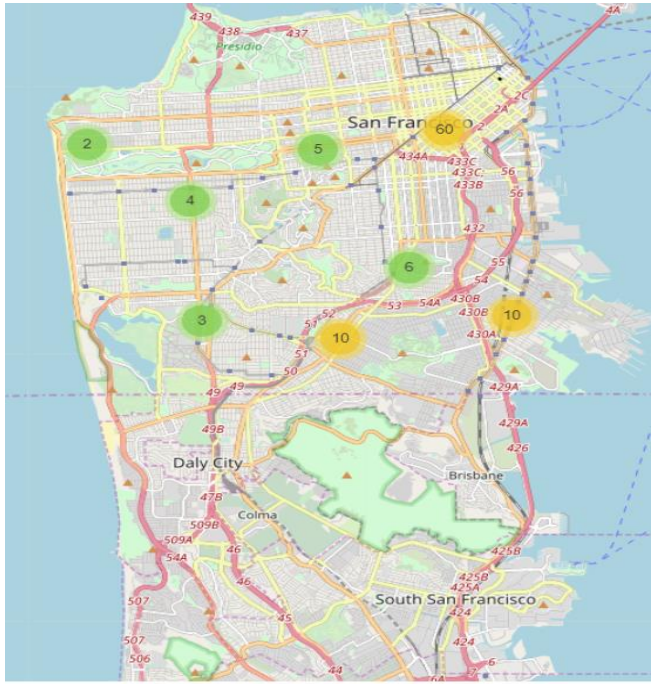
Data visualization

```
[7]  # create a Stamen Toner map of the world centered around Canada
     world_map = folium.Map(location=[56.130, -106.35], zoom_start=4, tiles='Stamen Toner')

     # display map
     world_map
```



Data visualization

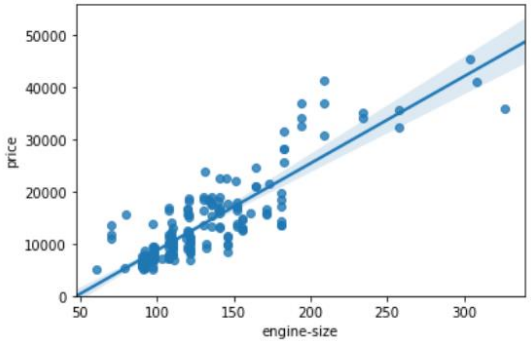




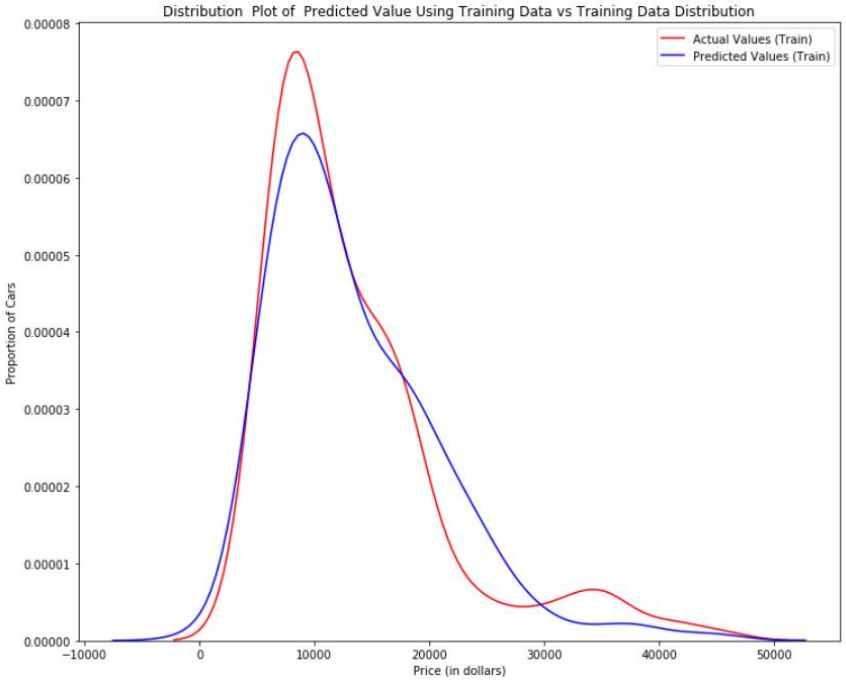
Data analysis

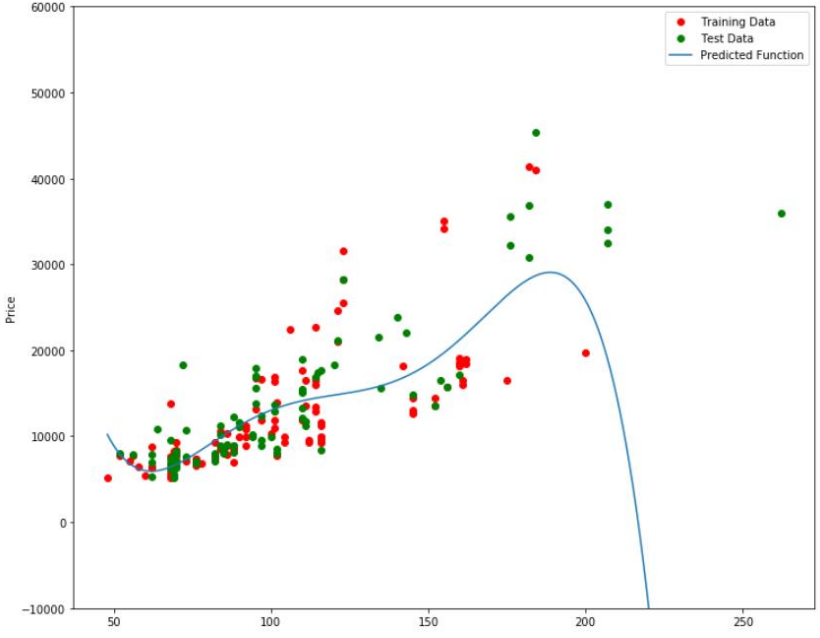
	symboling	normalized-losses	make	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	length	...	compression-ratio	horsepower	peak-rpm	city-mpg	highway-mpg	price
0	3	122	alfa-romero	std	two	convertible	rwd	front	88.6	0.811148	...	9.0	111.0	5000.0	21	27	13495.0
1	3	122	alfa-romero	std	two	convertible	rwd	front	88.6	0.811148	...	9.0	111.0	5000.0	21	27	16500.0
2	1	122	alfa-romero	std	two	hatchback	rwd	front	94.5	0.822681	...	9.0	154.0	5000.0	19	26	16500.0
3	2	164	audi	std	four	sedan	fwd	front	99.8	0.848630	...	10.0	102.0	5500.0	24	30	13950.0
4	2	164	audi	std	four	sedan	4wd	front	99.4	0.848630	...	8.0	115.0	5500.0	18	22	17450.0

```
# Engine size as potential predictor variable of price
sns.regplot(x="engine-size", y="price", data=df)
plt.ylim(0,)
```

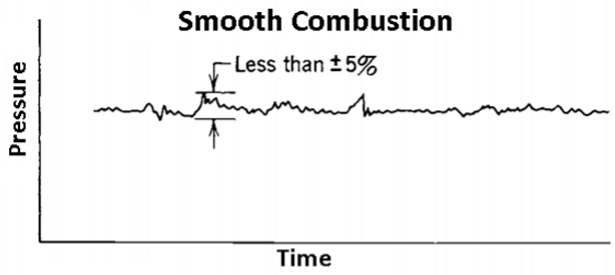


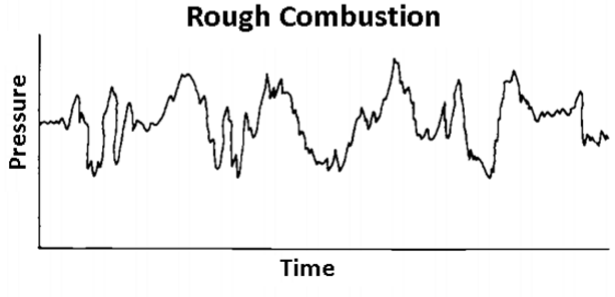
Data analysis

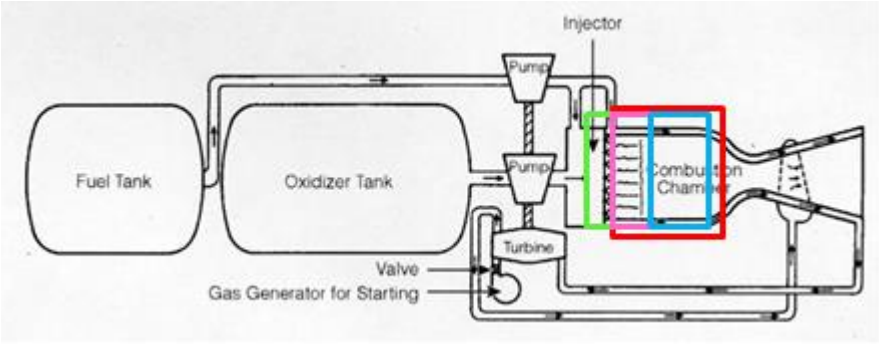


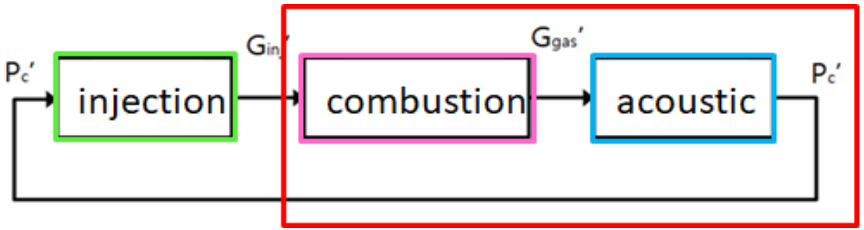


Combustion instability

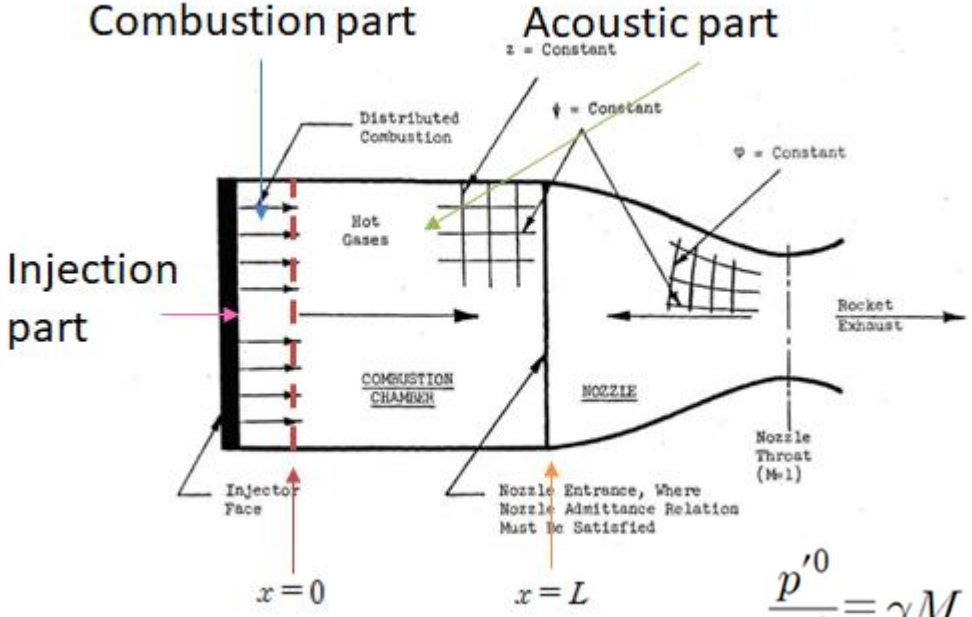








Combustion instability



$$Nozzle\ admittance = A = \frac{G_{gas}'}{p_c'}$$

$$\frac{G_{gas}'}{p'^{o}} = n(1 - e^{-iw\tau})$$

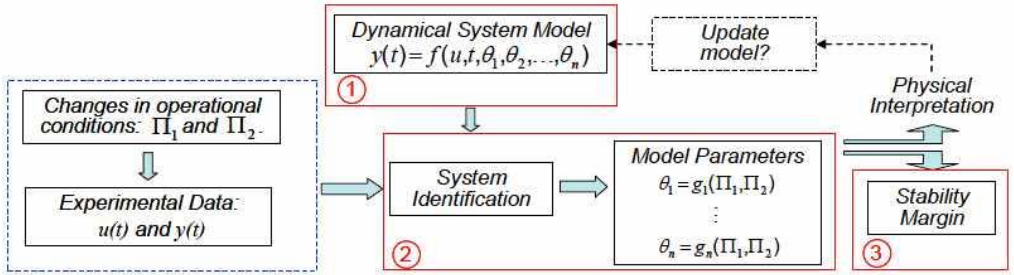
Acoustic Equation Combustion Equation

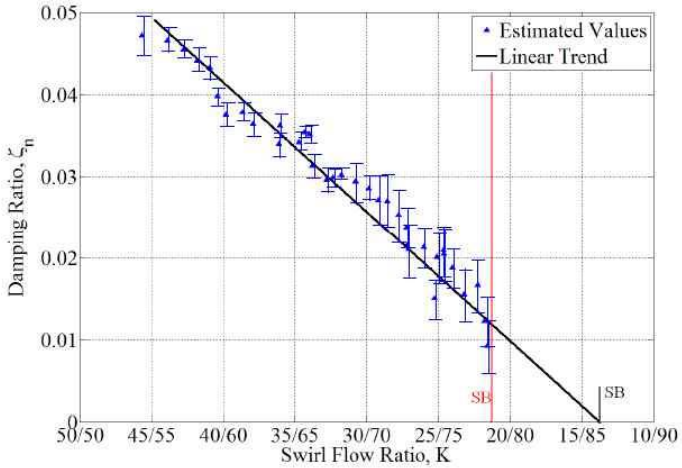
$$p_c' = p'^{o} \ , \quad u_c' = u'^{o} \quad (x = 0)$$

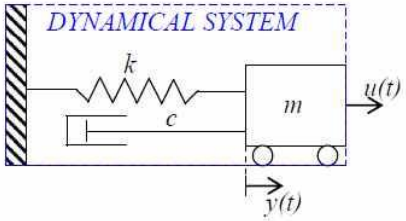
$$u_c' = \frac{\gamma - 1}{2\gamma} p_c' \qquad (x = L)$$

$$\frac{p'^{0}}{u'^{0}} = \gamma M \ \frac{1 - M(1+M)k_{mn}^{0} + B[1 + M(1-M)k_{mn}^{0}]}{1 - M(1+M)k_{mn}^{0} - B[1 + M(1-M)k_{mn}^{0}]}$$

Combustion instability







$$\frac{d^2x}{dt^2}+2\zeta\omega_n\frac{dx}{dt}+\omega_n^2x=0$$

Next plan

- 과거에는 액체로켓엔진 연소불안정 확인을 위해서 수치해석 또는 연소실험을 통한 평가가 많았다.
- 최근 데이터분석을 적용한 연소불안정 해석 연구결과가 도출되고 있다.
- 단기목표 : 데이터분석방법을 연구에 적용하여 수치해석, 연소실험결과와 비교