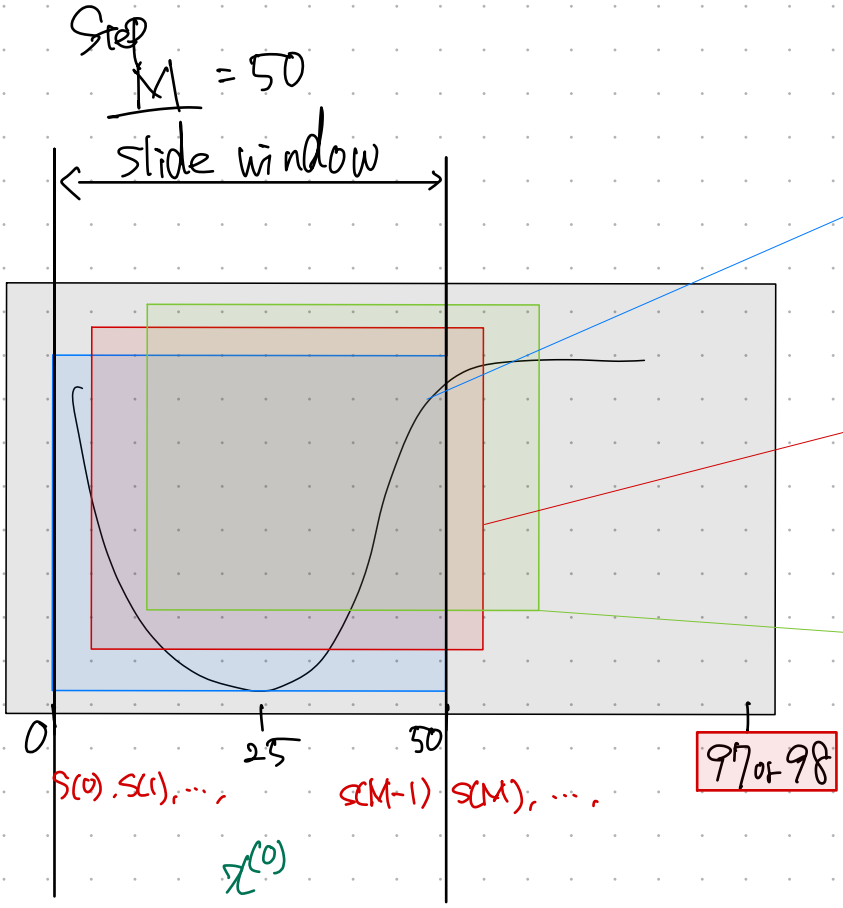
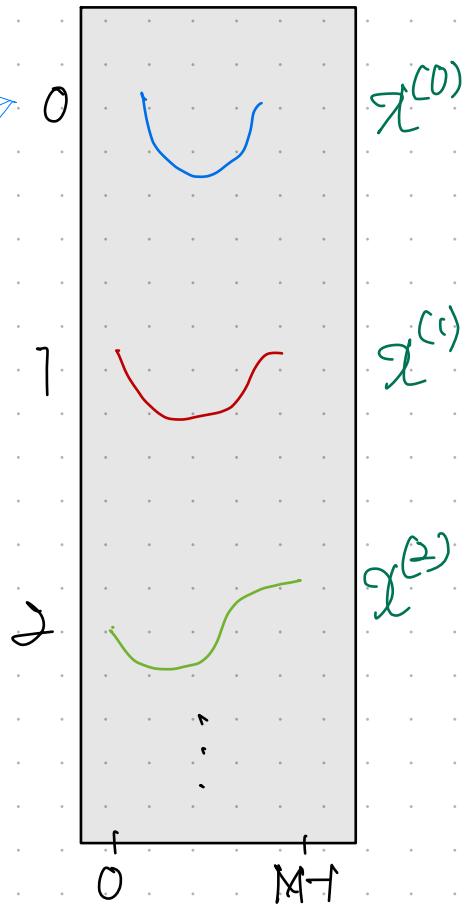
特異値分解 実装 XE

S(t)：適定ごとの時系列情報（電力） → split_experimental_data.csvで保存してある。





1次元時系列データ

M次元部分時系列データ

$S = [S(0), S(1), \cdots, S(L)]$, $L = \mathrm{len}(S)$

$M = 50$

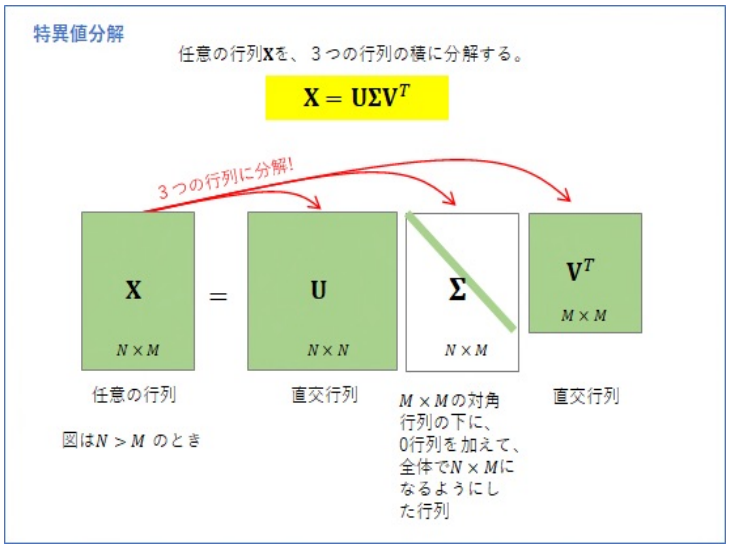
$X^{(M)} = [x^{(0)}, x^{(1)}, \cdots, x^{(M-1)}]$

$$= \begin{bmatrix} [S(0), S(1), \cdots, S(M-1)], \\ [S(1), S(2), \cdots, S(M)], \\ [S(2), S(3), \cdots, S(M+1)], \\ \vdots \\ [S(L-M), \cdots, S(L-2), S(L-1)] \end{bmatrix}$$

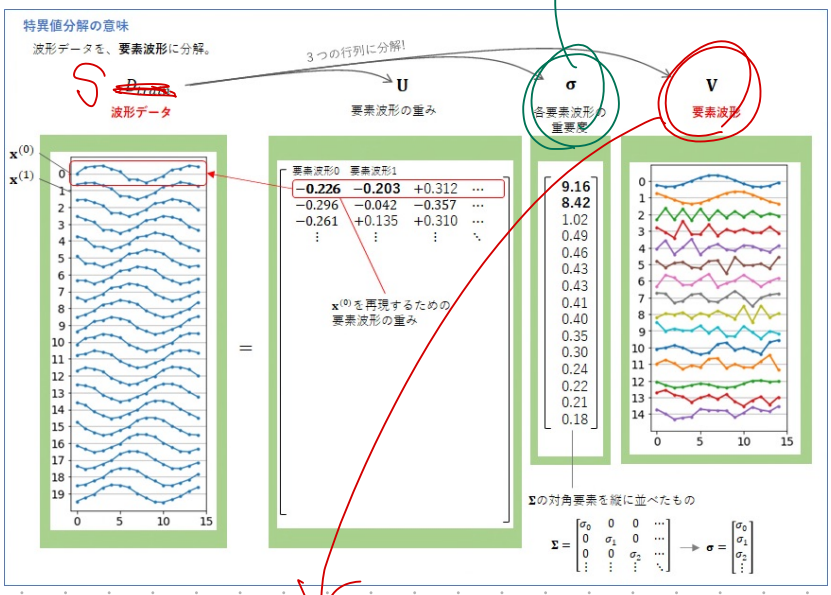
```
X = []
for i in range( len(S) - M ):
    x = []
    for j in range(M-1):
        x.append( S[i+j] )
    X.append(x)
```

Oになるはず？

What is SVG?



特異値



★特異値(重要度)が大きい要素波形は波形データの再現に必須
特異値が小さい要素波形は、なくてもあまり問題にならない

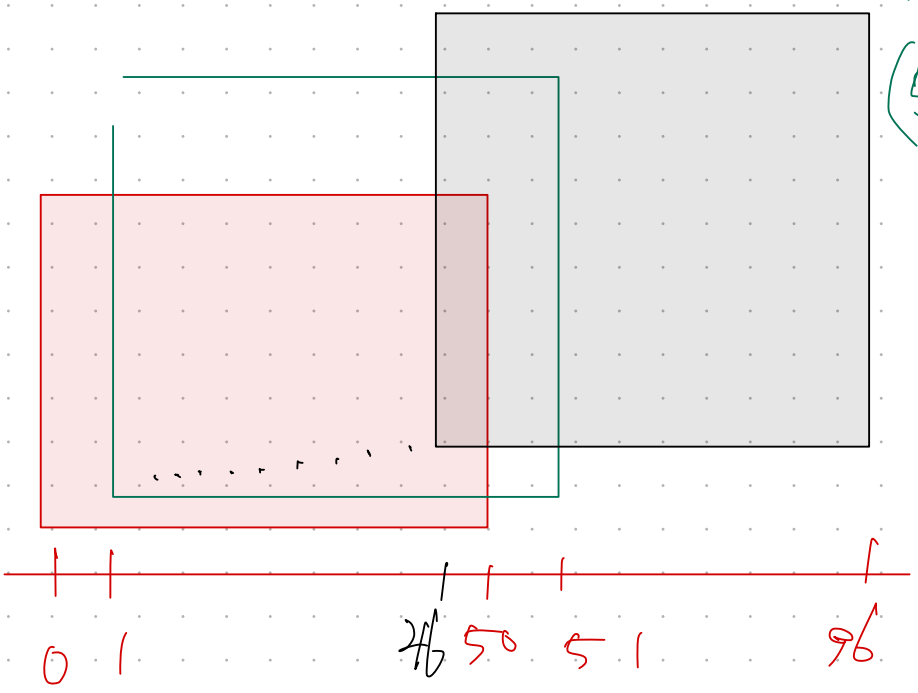
M × (M

※ len(S) < 2M

M × (len(S) − M)

※ len(S) < M のときは
そもそも部分時系列データが作れない

shape(V)
(50 × 47)



main関数概説

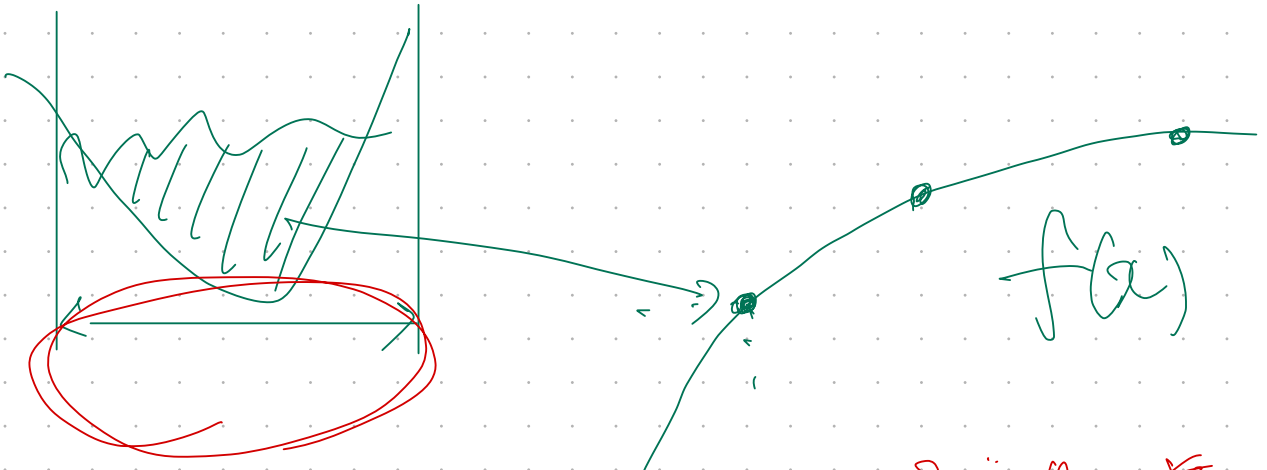
main.py | singular_value_decomposition.py | func.py | ITC_file_data_visualization.py | nitpic_file_data_visualization.py

```python
Python 3.7.6

% 実行方法
$ python3 main.py
'''
import os

# ITC_file_data_visualization.pyのitc_file_data_visualization関数をitsとしてimport
from ITC_file_data_visualization import itc_file_data_visualization as itc
# nitpic_file_data_visualization.pyのnitpic_file_data_visualization関数をnitpicとしてimport
from nitpic_file_data_visualization import nitpic_file_data_visualization as nitpic
# singular_value_decomposition.pyのsingular_value_decomposition関数をsvdとしてimport
from singular_value_decomposition import singular_value_decomposition as svd

"""
main関数
"""
def main():
    input_file_path_itc = "data/210107C.ITC"
    # input_file_path_nitpic = "data/200203C.nitpic"  # 210107C.nitpicもあるのか？

    os.makedirs("output", exist_ok=True)

    '''
    itc_file_data_visualization(分析対象のITCファイルのPATH（入力ファイル）, output_folder_path)
    第1引数: input_file_path_itc → 分析対象のITCファイルのパス
    第2引数: output_folder_path → 出力フォルダのパス ("/output")
    '''
    print('\033[34m' + 'RUN：itc_file_data_visualization' + '\033[0m')
    input_file_path_svd, all_data_path = itc(input_file_path_itc, "output/")
    print('\033[34m' + 'Task of itc_file_data_visualization completed!!!')

    '''
    singular_value_decomposition(input_file_path_itc, input_file_path_svd, all_data_path, output_folder_path, M, V_index)
    第1引数: input_file_path_itc → 分析対象のITCファイルのパス
    第2引数: input_folder_path → itc_file_data_visualization関数からの返り値でsplit_experimental_data/フォルダのパス
    第3引数: all_data_path → itc_file_data_visualization関数で作成したすべての実験データが保存されているCSVファイルのパス
    第4引数: output_folder_path → 出力フォルダのパス ("/output")
    第5引数: M → スライド窓のステップ数M
    第6引数: V_index → 要素波形のどこまでをピーク成分とするか
    '''
    print('\033[34m' + 'RUN：singular_value_decomposition' + '\033[0m')
    svd(input_file_path_itc, input_file_path_svd, all_data_path, "output/", 50, 4)
    print('\033[34m' + 'Task of singular_value_decomposition completed!!!')

    '''
    nitpic_file_data_visualization(input_file_path_nitpic, output_folder_path)
    第1引数: input_file_path_nitpic, → 分析対象のnitpicファイルのパス
    第2引数: output_folder_path → 出力フォルダのパス ("/output")
    '''
    # print('\033[34m' + 'RUN：nitpic_file_data_visualization' + '\033[0m')
    # nitpic(input_file_path_nitpic, "output/")
    # print('\033[34m' + 'Task of nitpic_file_data_visualization completed!!!' + '\033[0m')

if __name__ == '__main__':
    print('\033[33m' + 'RUN：main' + '\033[0m')
    main()
    print('\033[33m' + 'All tasks completed!!!' + '\033[0m')
```

Live Share　行 72、列 83　スペース: 4　UTF-8　LF　Python

ここにITCファイルのパスを書くだけ
必要な時にフォルダを作成するように修正しました
存在しないフォルダは、自動で作成
上から4つ分がPeak成分 (V(0)～V(3))
スライド窓のステップ数M
ITCファイルから抽出した全ての実験データが入ってるCSVファイルのパス
CSVファイルが分割して入ってるフォルダのパス
210107C/split_experiment_data/

1波形ごとのPeakとNoiseの差(積分値)をプロット



時間じゃなくて
ステップ数…

時間データに差し替え!!