先端データ解析論レポート 第 2 回

荻野 聖也（37-196323）

2019 年 4 月 21 日

課題 1

推定する真の関数　講義内でやったものと同様に

$$f(x) = \frac{\sin(\pi x)}{\pi x} + 0.1x$$

とする。

訓練サンプル　訓練サンプル $\{(x_i, y_i\}_{i=1}^n$ も，講義と同様に

$$y_i = f(x_i) + \varepsilon_i$$

ただし，ノイズ ε_i は正規分布

$$\varepsilon_i \sim N(0, \sigma^2)$$

に従うものとする。サンプル数 n は，$n = 50$ とする。

ガウスカーネルモデル

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{j=1}^{n} \theta_j \boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}_j)$$

$$\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{c}) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{c}\|^2}{2h^2}\right)$$

に対して，l_2 正則化の最小二乗回帰により学習させる。推定されるパラメータ $\hat{\boldsymbol{\theta}}$ は

$$\hat{\boldsymbol{\theta}} = \arg\min_{\hat{\boldsymbol{\theta}}} \left\{ \frac{1}{2} \sum_{i=1}^{n} \left(f_{\boldsymbol{\theta}}(x_i) - y_i\right)^2 + \frac{\lambda}{2}\|\boldsymbol{\theta}\|^2 \right\}$$

により求まる。ただし，λ は正則化パラメータである。ここで，

$$J \equiv \frac{1}{2} \sum_{i=1}^{n} \left(f_{\boldsymbol{\theta}}(x_i) - y_i\right)^2 + \frac{\lambda}{2}\|\boldsymbol{\theta}\|^2$$

とするとき，

$$\begin{aligned} J &= \frac{1}{2} \left(\sum_{i=1}^{n} \left(\sum_{j=1}^{n} \theta_j \boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) - y_i \right)^2 + \lambda\|\boldsymbol{\theta}\|^2 \right) \\ &= \frac{1}{2} \left(\|\boldsymbol{K}\boldsymbol{\theta} - \boldsymbol{y}\|^2 + \lambda\|\boldsymbol{\theta}\|^2 \right) \end{aligned}$$

なので，

$$\nabla_{\boldsymbol{\theta}} J = \boldsymbol{K}^T \boldsymbol{K} \boldsymbol{\theta} - \boldsymbol{K}^T \boldsymbol{y} + \lambda \boldsymbol{\theta}$$

なので，$\boldsymbol{\theta}$ に関する停留点は，

$$\nabla_{\boldsymbol{\theta}} J = \boldsymbol{0}$$

であたえられ，これを解くと，$\boldsymbol{K}$ が対称行列であることも考慮し，

$$\hat{\boldsymbol{\theta}} = (\boldsymbol{K}^2 + \lambda \boldsymbol{I})^{-1} \boldsymbol{K} \boldsymbol{y}$$

により，与えられる。

評価およびハイパーパラメータについて

交差確認法を用いる。ただし分割数 k は,

$$k = 10$$

とする。また，テスト誤差の平均値 e を

$$e = \frac{1}{k}\sum_{i=1}^{k}\sum_{x',y' \in group(k_i)}\left(\hat{f}(x') - y'\right)^2$$

とし，誤差の小ささを評価する。また，ハイパーパーラメータであるガウス幅 h と正則化パラメータ λ は

$$h \in \{0.1, 0.3, 1\}$$
$$\lambda \in \{0.05, 0.1, 0.5\}$$

とし，総当たり的に最適なパラメータを探索する。

結果

コードについては，付属の python ファイルか補填を参考にしてください。
各パラメータによる交差確認による誤差の結果を下に示す。

交差確認法による誤差

```
ガウス幅:0.10 正則化パラム:0.05 err:0.2802
ガウス幅:0.10 正則化パラム:0.10 err:0.2016
ガウス幅:0.10 正則化パラム:0.50 err:0.2245
ガウス幅:0.30 正則化パラム:0.05 err:0.0222
ガウス幅:0.30 正則化パラム:0.10 err:0.0239
ガウス幅:0.30 正則化パラム:0.50 err:0.0262
ガウス幅:1.00 正則化パラム:0.05 err:0.0440
ガウス幅:1.00 正則化パラム:0.10 err:0.0665
ガウス幅:1.00 正則化パラム:0.50 err:0.1221
```

これより，求めるべき正則化パラメータ λ およびガウス幅 h は,

$$(\lambda, h) = (0.05, 0.30)$$

に決定された。なおこの時の予測関数 $\hat{f}_{\boldsymbol{\theta}}(x)$ のグラフを以下に示す。

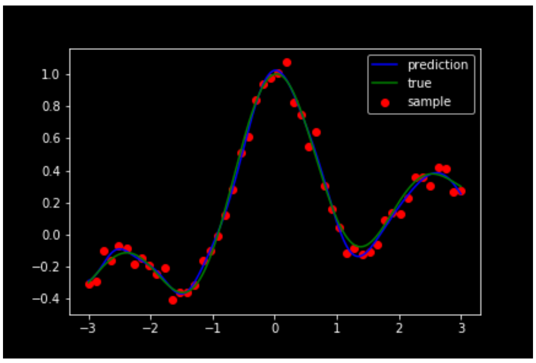

図 1　l2 正則化付き最小二乗回帰

課題 2

線形モデル

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{j=1}^{b} \theta_j \phi_j(\boldsymbol{x})$$

について，l_2 正則化回帰によるパラメータ推定を行う。この時，学習により得られるパラメータ $\hat{\boldsymbol{\theta}}$ について，LOOCV のために $(\boldsymbol{x}_i, y_i)$ を除いて学習した時のパラメータを $\hat{\boldsymbol{\theta}}_i$ としたときに，これは

$$\hat{\boldsymbol{\theta}}_i = \arg\min_{\boldsymbol{\theta}_i} \left[\frac{1}{2} \sum_{\substack{j=1 \\ j \neq i}}^{n} (f_{\boldsymbol{\theta}_i}(\boldsymbol{x}_j) - y_j)^2 + \frac{\lambda}{2} \|\boldsymbol{\theta}_i\|^2 \right]$$

とあらわされる。以下，計画行列 $\boldsymbol{\Phi}$ に対して，$(\boldsymbol{x}_i, y_i)$ を除く学習によるものを $\boldsymbol{\Phi}_i$ とし，サンプル $\{z_i\}_{i=1}^{n} = \{(x_i, y_i)\}_{i=1}^{n}$ に対し，i 番目を除いた x および y のベクトル表記を $\boldsymbol{x}^{\{i\}} \in R^{n-1}$，$\boldsymbol{y}^{\{i\}} \in R^{n-1}$ とする。(表記ややこしくてすいません。) 講義内と同様に最小二乗回帰を解く。

$$J \equiv \frac{1}{2} \sum_{\substack{j=1 \\ j \neq i}}^{n} (f_{\boldsymbol{\theta}_i}(\boldsymbol{x}_j) - y_j)^2 + \frac{\lambda}{2} \|\boldsymbol{\theta}_i\|^2$$

とするとき，これを変形して，

$$\begin{aligned} J &= \frac{1}{2} \|\boldsymbol{\Phi}_i \boldsymbol{\theta}_i - \boldsymbol{y}^{\{i\}}\|^2 + \frac{\lambda}{2} \|\boldsymbol{\theta}_i\|^2 \\ &= \frac{1}{2} \left(\boldsymbol{\Phi}_i \boldsymbol{\theta}_i - \boldsymbol{y}^{\{i\}} \right)^T \left(\boldsymbol{\Phi}_i \boldsymbol{\theta}_i - \boldsymbol{y}^{\{i\}} \right) + \frac{\lambda}{2} \|\boldsymbol{\theta}_i\|^2 \\ &= \frac{1}{2} \left(\boldsymbol{\theta}_i^T \boldsymbol{\Phi}_i^T \boldsymbol{\Phi}_i \boldsymbol{\theta}_i - \boldsymbol{\theta}_i^T \boldsymbol{\Phi}_i^T \boldsymbol{y}^{\{i\}} - \boldsymbol{y}^{\{i\}T} \boldsymbol{\Phi}_i \boldsymbol{\theta}_i + \|\boldsymbol{y}^{\{i\}}\|^2 \right) + \frac{\lambda}{2} \|\boldsymbol{\theta}_i\|^2 \end{aligned}$$

J の $\boldsymbol{\theta}_i$ に関する停留点を求めると，

$$\nabla_{\boldsymbol{\theta}_i} J = \boldsymbol{\Phi}_i^T \boldsymbol{\Phi}_i \boldsymbol{\theta}_i - \boldsymbol{\Phi}_i^T \boldsymbol{y}^{\{i\}} + \lambda \boldsymbol{\theta}_i$$

なので，

$$\boldsymbol{\theta}_i = (\boldsymbol{\Phi}_i^T \boldsymbol{\Phi}_i + \lambda \boldsymbol{I})^{-1} \boldsymbol{\Phi}_i^T \boldsymbol{y}^{\{i\}}$$

で与えられる。ここで，$\boldsymbol{\Phi}_i^T \boldsymbol{\Phi}_i$ について，$\boldsymbol{\Phi}_i$ は $\{\phi_j\}_{j \neq i}$ で表されるので，

$$\begin{aligned} \boldsymbol{\Phi}_i^T \boldsymbol{\Phi}_i &= \begin{pmatrix} \sum_{j=1}^n \phi_1^2(\boldsymbol{x}_j) - \phi_1^2(\boldsymbol{x}_i) & \sum_{j=1}^n \phi_1(\boldsymbol{x}_j)\phi_2(\boldsymbol{x}_j) - \phi_1(\boldsymbol{x}_i)\phi_2(\boldsymbol{x}_i) & \dots & \sum_{j=1}^n \phi_1(\boldsymbol{x}_j)\phi_b(\boldsymbol{x}_j) - \phi_1(\boldsymbol{x}_i)\phi_b(\boldsymbol{x}_i) \\ \sum_{j=1}^n \phi_2(\boldsymbol{x}_j)\phi_1(\boldsymbol{x}_j) - \phi_2(\boldsymbol{x}_i)\phi_1(\boldsymbol{x}_i) & \sum_{j=1}^n \phi_2^2(\boldsymbol{x}_j) - \phi_2^2(\boldsymbol{x}_i) & \dots & \dots \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{j=1}^n \phi_b(\boldsymbol{x}_j)\phi_1(\boldsymbol{x}_j) - \phi_b(\boldsymbol{x}_i)\phi_1(\boldsymbol{x}_i) & \dots & \dots & \sum_{j=1}^n \phi_b^2(\boldsymbol{x}_j) - \phi_b^2(\boldsymbol{x}_i) \end{pmatrix} \\ &= \boldsymbol{\Phi}^T \boldsymbol{\Phi} - \begin{pmatrix} \phi_1^2 & \phi_1\phi_2 & \dots & \phi_1\phi_b \\ \vdots & \phi_2^2 & & \\ \vdots & & \ddots & \vdots \\ \phi_b\phi_1 & \dots & \dots & \phi_b^2 \end{pmatrix}_{\boldsymbol{x}=\boldsymbol{x}_i} \\ &= \boldsymbol{\Phi}^T \boldsymbol{\Phi} - \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_b \end{bmatrix} \begin{bmatrix} \phi_1 & \phi_2 & \dots & \phi_b \end{bmatrix} \\ &= \boldsymbol{\Phi}^T \boldsymbol{\Phi} - \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T \end{aligned}$$

また，$\boldsymbol{\Phi}_i^T \boldsymbol{y}^{\{i\}}$ について，同様にして

$$
\begin{aligned}
\boldsymbol{\Phi}_i^T \boldsymbol{y}^{\{i\}} &= \sum_{\substack{j=1 \\ j \neq i}}^{n} y_j \boldsymbol{\phi}_j \\
&= \sum_{j=1}^{n} y_j \boldsymbol{\phi}_j - y_i \boldsymbol{\phi}_i \\
&= \boldsymbol{\Phi}^T \boldsymbol{y} - y_i \boldsymbol{\phi}_i
\end{aligned}
$$

さらに，$(\boldsymbol{\Phi}_i^T \boldsymbol{\Phi}_i + \lambda \boldsymbol{I})^{-1}$ について，

$$
\begin{aligned}
(\boldsymbol{\Phi}_i^T \boldsymbol{\Phi}_i + \lambda \boldsymbol{I})^{-1} &= (\boldsymbol{\Phi}^T \boldsymbol{\Phi} - \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T + \lambda \boldsymbol{I})^{-1} \\
&= \{(\boldsymbol{\Phi}^T \boldsymbol{\Phi} + \lambda \boldsymbol{I}) - \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T\}^{-1}
\end{aligned}
$$

$\boldsymbol{U} = \boldsymbol{\Phi}^T \boldsymbol{\Phi} + \lambda \boldsymbol{I}$ とすると，講義資料の Sherman-Morrison-Woodbury の公式を用いて，

$$
\begin{aligned}
(\boldsymbol{\Phi}_i^T \boldsymbol{\Phi}_i + \lambda \boldsymbol{I})^{-1} &= (\boldsymbol{U} - \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T)^{-1} \\
&= \boldsymbol{U}^{-1} + \frac{\boldsymbol{U}^{-1} \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T \boldsymbol{U}^{-1}}{1 - \boldsymbol{\phi}_i^T \boldsymbol{U}^{-1} \boldsymbol{\phi}_i}
\end{aligned}
$$

とあらわされるので，以上のことを用いて，$\boldsymbol{\theta}_i$ の学習されたパラメータ $\hat{\boldsymbol{\theta}}_i$ は，

$$
\boldsymbol{\theta}_i = \left(\boldsymbol{U}^{-1} + \frac{\boldsymbol{U}^{-1} \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T \boldsymbol{U}^{-1}}{1 - \boldsymbol{\phi}_i^T \boldsymbol{U}^{-1} \boldsymbol{\phi}_i} \right) (\boldsymbol{\Phi}^T \boldsymbol{y} - y_i \boldsymbol{\phi}_i)
$$

以下，$\boldsymbol{\phi}_i^T \boldsymbol{U}^{-1} \boldsymbol{\phi}_i = a_i \ (\in R)$ として．

$$
\begin{aligned}
\hat{\boldsymbol{\theta}}_i &= \left(\boldsymbol{I} + \frac{\boldsymbol{U}^{-1} \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T}{1 - \boldsymbol{\phi}_i^T \boldsymbol{U}^{-1} \boldsymbol{\phi}_i} \right) \boldsymbol{U}^{-1} (\boldsymbol{\Phi}^T \boldsymbol{y} - y_i \boldsymbol{\phi}_i) \\
&= \left(\boldsymbol{I} + \frac{\boldsymbol{U}^{-1} \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T}{1 - a_i} \right) \boldsymbol{U}^{-1} \boldsymbol{\Phi}^T \boldsymbol{y} - \frac{(1 - a_i) \boldsymbol{I} + \boldsymbol{U}^{-1} \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T}{1 - a_i} \boldsymbol{U}^{-1} \boldsymbol{\phi}_i y_i \\
&= \left(\boldsymbol{I} + \frac{\boldsymbol{U}^{-1} \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T}{1 - a_i} \right) \boldsymbol{U}^{-1} \boldsymbol{\Phi}^T \boldsymbol{y} - \frac{y_i}{1 - a_i} ((1 - a_i) \boldsymbol{U}^{-1} \boldsymbol{\phi}_i + a_i \boldsymbol{U}^{-1} \boldsymbol{\phi}_i) \\
&= \left(\boldsymbol{I} + \frac{\boldsymbol{U}^{-1} \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T}{1 - a_i} \right) \boldsymbol{U}^{-1} \boldsymbol{\Phi}^T \boldsymbol{y} - \frac{y_i}{1 - a_i} \boldsymbol{U}^{-1} \boldsymbol{\phi}_i
\end{aligned}
$$

よって，(x_i, y_i) を抜いて学習したパラメータからの y_i の推定値 $\boldsymbol{\phi}^T \hat{\boldsymbol{\theta}}_i$ は，

$$
\begin{aligned}
\boldsymbol{\phi}_{\boldsymbol{i}}^T \hat{\boldsymbol{\theta}}_i &= \left(\boldsymbol{\phi}_i^T + \frac{\boldsymbol{\phi}_i^T \boldsymbol{U}^{-1} \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T}{1 - a_i} \right) \boldsymbol{U}^{-1} \boldsymbol{\Phi}^T \boldsymbol{y} - \frac{y_i}{1 - a_i} \boldsymbol{\phi}_i^T \boldsymbol{U}^{-1} \boldsymbol{\phi}_i \\
&= \left(1 + \frac{a_i}{1 - a_i} \right) \boldsymbol{\phi}_i^T \boldsymbol{U}^{-1} \boldsymbol{\Phi}^T \boldsymbol{y} - \frac{a_i}{1 - a_i} y_i \\
&= \frac{1}{1 - a_i} \left(\boldsymbol{\phi}_i^T \boldsymbol{U}^{-1} \boldsymbol{\Phi}^T \boldsymbol{y} - a_i y_i \right)
\end{aligned}
$$

よって，

$$
\boldsymbol{\phi}_{\boldsymbol{i}}^T \hat{\boldsymbol{\theta}}_i - y_i = \frac{1}{1 - a_i} \left(\boldsymbol{\phi}_i^T \boldsymbol{U}^{-1} \boldsymbol{\Phi}^T \boldsymbol{y} - y_i \right)
$$

これより，LOOCV による 2 乗誤差 ε は，

$$
\begin{aligned}
\varepsilon &= \frac{1}{n}\sum_{i=1}^{n}\left(\boldsymbol{\phi}_{\boldsymbol{i}}^T\hat{\boldsymbol{\theta}}_i - y_i\right)^2 \\
&= \frac{1}{n}\sum_{i=1}^{n}\left\{\frac{1}{1-a_i}(\boldsymbol{\phi}_i^T\boldsymbol{U}^{-1}\boldsymbol{\Phi}^T\boldsymbol{y} - y_i)\right\}^2 \\
&= \frac{1}{n}\left\|\begin{pmatrix} \frac{1}{1-a_1}(\boldsymbol{\phi}_1^T\boldsymbol{U}^{-1}\boldsymbol{\Phi}^T\boldsymbol{y} - y_1) \\ \frac{1}{1-a_2}(\boldsymbol{\phi}_2^T\boldsymbol{U}^{-1}\boldsymbol{\Phi}^T\boldsymbol{y} - y_2) \\ \vdots \\ \frac{1}{1-a_n}(\boldsymbol{\phi}_n^T\boldsymbol{U}^{-1}\boldsymbol{\Phi}^T\boldsymbol{y} - y_n) \end{pmatrix}\right\|^2
\end{aligned}
$$

ここでカッコ内について

$$
\boldsymbol{Y} \equiv \begin{pmatrix} \frac{1}{1-a_1}(\boldsymbol{\phi}_1^T\boldsymbol{U}^{-1}\boldsymbol{\Phi}^T\boldsymbol{y} - y_1) \\ \frac{1}{1-a_2}(\boldsymbol{\phi}_2^T\boldsymbol{U}^{-1}\boldsymbol{\Phi}^T\boldsymbol{y} - y_2) \\ \vdots \\ \frac{1}{1-a_n}(\boldsymbol{\phi}_n^T\boldsymbol{U}^{-1}\boldsymbol{\Phi}^T\boldsymbol{y} - y_n) \end{pmatrix}
$$

$$
\begin{aligned}
\boldsymbol{Y} &= \begin{pmatrix} \frac{1}{1-a_1} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{1-a_n} \end{pmatrix}\left(\begin{pmatrix} \boldsymbol{\phi}_1^T \\ \vdots \\ \boldsymbol{\phi}_n^T \end{pmatrix}\boldsymbol{U}^{-1}\boldsymbol{\Phi}^T\boldsymbol{y} - \boldsymbol{y}\right) \\
&= \begin{pmatrix} \frac{1}{1-a_1} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{1-a_n} \end{pmatrix}(\boldsymbol{\Phi}\boldsymbol{U}^{-1}\boldsymbol{\Phi}^T - \boldsymbol{I})\boldsymbol{y} \\
&= -\begin{pmatrix} \frac{1}{1-a_1} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{1-a_n} \end{pmatrix}(\boldsymbol{I} - \boldsymbol{\Phi}\boldsymbol{U}^{-1}\boldsymbol{\Phi}^T)\boldsymbol{y}
\end{aligned}
$$

ここで，

$$
\boldsymbol{H} = \boldsymbol{I} - \boldsymbol{\Phi}\boldsymbol{U}^{-1}\boldsymbol{\Phi}^T
$$

について，この対角成分 $D_j(j = 1, 2, ..., n)$ は

$$
D_j = 1 - a_j
$$

なので，

$$
\tilde{\boldsymbol{H}}^{-1} = \begin{pmatrix} \frac{1}{1-a_1} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{1-a_n} \end{pmatrix}
$$

となり，

$$
\varepsilon = \frac{1}{n}\|\tilde{\boldsymbol{H}}^{-1}\boldsymbol{H}\boldsymbol{y}\|^2
$$

よって題意は示された。

補遺

Listing 1　課題 1 のソースコード

```
    ##Requirement
import numpy as np
import matplotlib.pyplot as plt
import random
```

```
def func(x, with_noise=True):
    """基本方程式
␣␣␣␣Arg:
␣␣␣␣␣␣␣␣x(float)␣座標 x
␣␣␣␣␣␣␣␣with_noise(boolian)␣ノイズ入れるかどうか
␣␣␣␣Return:
␣␣␣␣␣␣␣␣fx(float)␣座標に対応する xy(with_noise?)
␣␣␣␣"""
    noise = 0.05 * np.random.normal(loc=0., scale=1.) if with_noise else 0
    return np.sin(np.pi*x)/np.pi/x+0.1*x + noise

def get_sample(x_min=-3., x_max=3., num_sample=50):
    """サンプル生成
␣␣␣␣Arg:
␣␣␣␣␣␣␣␣x_min(float)␣座標の最小値 x␣default=-3.
␣␣␣␣␣␣␣␣x_max(float)␣座標の最大値 x␣default=3.
␣␣␣␣␣␣␣␣num_sample(int)␣サンプルの生成数 (=dim(theta))␣default=50

␣␣␣␣Return:
␣␣␣␣␣␣␣␣x_lst(list(float))␣座標の集合 x
␣␣␣␣␣␣␣␣fx_lst(list(float))␣座標の集合 y
␣␣␣␣"""
    x_lst = np.linspace(x_min, x_max, num_sample)
    fx_lst = np.array([func(x, with_noise=True) for x in x_lst])
    return x_lst, fx_lst

def calc_least_squared_parameter(x_lst, fx_lst, h, lam, num=1000):
    """正則化付きで最小二乗回帰を解く l2
␣␣␣␣Arg:
␣␣␣␣␣␣␣␣x_lst(ndarray(float))␣サンプルの座標 x
␣␣␣␣␣␣␣␣fx_lst(ndarray(float))␣サンプルの出力系 (に対応 x)
␣␣␣␣␣␣␣␣h(float)␣ハイパーパラメータ

␣␣␣␣Return:
␣␣␣␣␣␣␣␣x_lst_detail(ndarray(float))␣予測系の座標 x自分で設定する ()
␣␣␣␣␣␣␣␣fx_pred(ndarray(float))␣最小二乗回帰で予測された f(x)(に対応させて。x)
␣␣␣␣"""ガウスカーネル
    #
    def get_kernel(x, c, h):
        return np.exp(-(x-c)**2/2/h**2)次元
    #
    n = len(x_lst)
    k_matrix = np.empty((n,n))
    for i in range(len(k_matrix)):
        for j in range(len(k_matrix[0])):
            k_matrix[i][j] = get_kernel(x_lst[i], x_lst[j], h)
    #theta(dim=n)
    theta_pred = np.dot(np.linalg.inv((np.dot(k_matrix, k_matrix)+lam*np.eye(n))), np.dot(k_matrix.T, fx_lst))以下生
        成フェーズ
    #
    x_lst_detail = np.linspace(-3, 3, num=num)
    #値の予測 fx引数 (x)
    def get_pred_fx(x):
        fx_pred = 0
        for i in range(n):
            k=get_kernel(x, x_lst[i], h)
            fx_pred += theta_pred[i]*k
        return fx_pred

    fx_pred_lst = np.empty_like(x_lst_detail)
    for i in range(len(x_lst_detail)):
        fx_pred_lst[i] = get_pred_fx(x_lst_detail[i])
```

```

    return x_lst_detail, fx_pred_lst

def calc_least_squared_parameter_with_cross_valid(x_lst, fx_lst, h, lam, k_num):
    """クロスバリデーション付きで最小二乗回帰を解く l2
␣␣␣␣Arg:
␣␣␣␣␣␣␣␣x_lst(ndarray(float))サンプルの座標 x
␣␣␣␣␣␣␣␣fx_lst(ndarray(float))サンプルの出力座標 (座標に対応 x)
␣␣␣␣␣␣␣␣h(float)
␣␣␣␣␣␣␣␣lam(float)
␣␣␣␣␣␣␣␣k_num(int)␣分割数

␣␣␣␣Return:
␣␣␣␣␣␣␣␣err_mean(float)␣誤差値の平均
␣␣␣␣"""サンプル数
    #
    sample_num = len(x_lst)
    #→シャッフルさせる index
    idx = list(range(sample_num))
    random.shuffle(idx)分割された時のリスト当たりの個数
    #1
    split_lst_num = int(sample_num/k_num)
    #を分割 index
    idx_split = [idx[i:i+split_lst_num] for i in range(0, sample_num, split_lst_num)]
    err = 0
    for ki in range(k_num):
        test_idx = idx_split[ki]
        train_idx = []
        for idx in idx_split:
            if idx == test_idx:
                pass
            else:
                for i in idx:
                    train_idx.append(i)
        train_x = [x_lst[i] for i in train_idx]
        train_y = [fx_lst[i] for i in train_idx]
        test_x = [x_lst[i] for i in test_idx]
        test_y = [fx_lst[i] for i in test_idx]最小二乗回帰により解く
        #
        _, y_pred = calc_least_squared_parameter(train_x, train_y, h, lam, num=sample_num)
        y_pred_for_err = [y_pred[i] for i in test_idx]
        for i in range(len(test_x)):
            err += (y_pred_for_err[i]−test_y[i])**2
    err /= k_num
    return err

def plot_graph(x_sample, y_sample, x_pred, y_pred, y_acc):
    fig = plt.figure()
    plt.clf()
    plt.scatter(x_sample, y_sample, label="sample" ,c='red', marker='o')
    plt.plot(x_pred, y_pred, label="prediction", c='blue')
    plt.plot(x_pred, y_acc, label="true", c="green")
    plt.legend()
    plt.savefig(r"C:\Users\msy-o\Documents\lecture\summer\data_analytics\report\02\output\prediction.png")
    plt.show()
    return None

#サンプル生成とりあえず個 (50)
x_sample, y_sample = get_sample()
#ガウス幅 h
gauss_widths = [0.1, 0.3, 1]
#正則化パラメータ
regularize_params = [0.05, 0.1, 0.5]
```

```
#分割数 k
k = 10

errs = []
#とラムダを変化させて実験 h
for h in gauss_widths:
    for l in regularize_params:
        err = calc_least_squared_parameter_with_cross_valid(x_sample, y_sample, h, l, k)
        print("ガウス幅:%1.2f␣正則化パラム:%1.2f␣err:%1.4f" %(h, l, err))
        errs.append([h,l,err])
#最も評価の良かったと正則化パラムとその時のをとる herr
err_min = min(errs, key=lambda x: x[2])
print(err_min)
x_pred, y_pred = calc_least_squared_parameter(x_sample, y_sample, h=err_min[0], lam=err_min[1])
y_acc = np.empty_like(x_pred)
for i in range(len(y_acc)):
    y_acc[i] = func(x_pred[i], with_noise=False)

plot_graph(x_sample, y_sample, x_pred, y_pred, y_acc)
```