7.2.1 回帰のためのRVM

平成 28 年 9 月 11 日

概 要

PRML の「7.2.1 回帰のための RVM」についての実装と考察

目 次

1　問題設定

回帰問題に RVM を適用する. メリットとしては, 疎な解が得られるということである.

2　アルゴリズム

与えられた入力ベクトル $\boldsymbol{x}$ に対する実数うちの目標変数 t の条件付分布を

$$p(t|\mathbf{x}, \mathbf{w}, \beta) = N(t|y(\mathbf{x}), \beta^{-1}) \quad (7.76)$$

とする, ただし平均は,

$$y(\mathbf{x}) = \sum_{m=1}^{M} w_m \phi_m(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) \quad (7.77)$$

で与える. これを N このデータ点でまとめると,

$$p(\mathbf{t}|X, \mathbf{w}, \beta) = \prod_{n=1}^{N} N(t_n|y(\mathbf{x}_n), \beta^{-1}) \quad (7.79)$$

となる.
次に事前分布を導入するが, ここで

$$p(\mathbf{w}|\boldsymbol{\alpha}) = \prod_{m=1}^{M} N(w_m|\mathbf{0}, \alpha_m^{-1}) \quad (7.80)$$

と, それぞれの要素に対して別々の精度パラメータを用意する.
線形ベイズモデルの畳込みは解析的に解けるため, 重みベクトル W に対する事後分布は

$$p(\mathbf{w}|\mathbf{t}, X, \alpha, \beta) = N(t|\mathbf{m}, \Sigma) \quad (7.81)$$

ここで, 平均, 分散は

$$\mathbf{m} = \beta \Sigma \Phi^T \mathbf{t}, \ \Sigma = (A + \beta \Phi^T \Phi)^{-1} \quad (7.82), (7.83)$$

となる. ただし, Φ は $\phi_{nm} = \phi_m(\mathbf{x}_n), \ A = diag(\alpha_n)$ である.
パラメータ $\boldsymbol{\alpha}, \beta$ はエビデンス近似によって求めることができ,

$$\alpha_i^{new} = \frac{\gamma_i}{m_i^2}, \ \beta^{new} = \frac{N - \sum_i \gamma_i}{\|\mathbf{t} - \Phi \mathbf{m}\|^2} \quad (7.87), (7.88)$$

ただし,

$$\gamma_i = 1 - \alpha_i \Sigma_{ii}$$

であり, この再定義式を解き, 収束した値を用いることで予測分布が導出でき

$$p(t|\mathbf{x}, X, \mathbf{t}, \alpha^*, \beta^*) = N(t|\mathbf{m}^T \boldsymbol{\phi}(\mathbf{x}), \sigma^2(\mathbf{x})) \quad (7.90)$$

ただし,

$$\mathbf{m} = \beta \Sigma \Phi^T \mathbf{t}, \ \sigma^2 = (\beta^*)^{-1} + \boldsymbol{\phi}(\mathbf{x})^T \Sigma \boldsymbol{\phi}(\mathbf{x}) \quad (7.82), (7.91)$$

3　コード

RVM 回帰のコード (RVRpy).

```
def gaussian_basis(x,z):
        theta=0.5*M
        return np.exp(-theta*(x-z)**2)

#基底関数
P=np.zeros((N,M))
mu=[2*pi*(m+1/2)/(M-1) for m in range(M-1)]
for n in range(N):
        for m in range(M-1):
                P[n,m]=gaussian_basis(x[n],mu[m])
        P[n,M-1]=1

sig=np.zeros((N,N))
mean=np.zeros(N)
A=np.zeros((M,M))
alpha=np.ones(M)
gamma=np.zeros(M)
beta=1
frag=0
roop=0

"""パラメータ決定"""
while frag==0:
        for m in range(M):
                A[m,m]=alpha[m]
        sig=inv(A+beta*dot(P.T,P))
        mean=beta*dot(sig,dot(P.T,t))

        temp1,temp2=alpha,beta

        for m in range(M):
                gamma[m]=1-alpha[m]*sig[m,m]
                alpha[m]=min(gamma[m]/(mean[m]**2+1.7e-300),1.7e+300)
        beta=(N-np.sum(gamma))/(norm(t-dot(P,mean))**2)

        roop+=1
        if norm(temp1-alpha)<M*10**-6 and abs(temp2-beta)<10**-6 or roop>10000:
                frag=1
print("(N,M)=",N,M)
print("aplha=",alpha)

"""モデル"""
def model_f(z):
        psi=np.zeros(M)
        for m in range(M-1):
                psi[m]=gaussian_basis(z,mu[m])
        psi[M-1]=1
        return dot(mean,psi)
```

4　結果

4.1　ガウスカーネル

カーネル関数に

$$k(\mathbf{x}_n, \mathbf{x}_m) = exp(-\theta(x_n - \mu_m)^2)$$

を用いた. ただし, データ数は $N = 20, 50, 100, 200,\ M = 5, 10, 30, 50$ とし $\theta = 0.5M$ を用いた.

4.1.1 回帰結果

図 1: $M = 5, N = 20, 50, 100, 200$

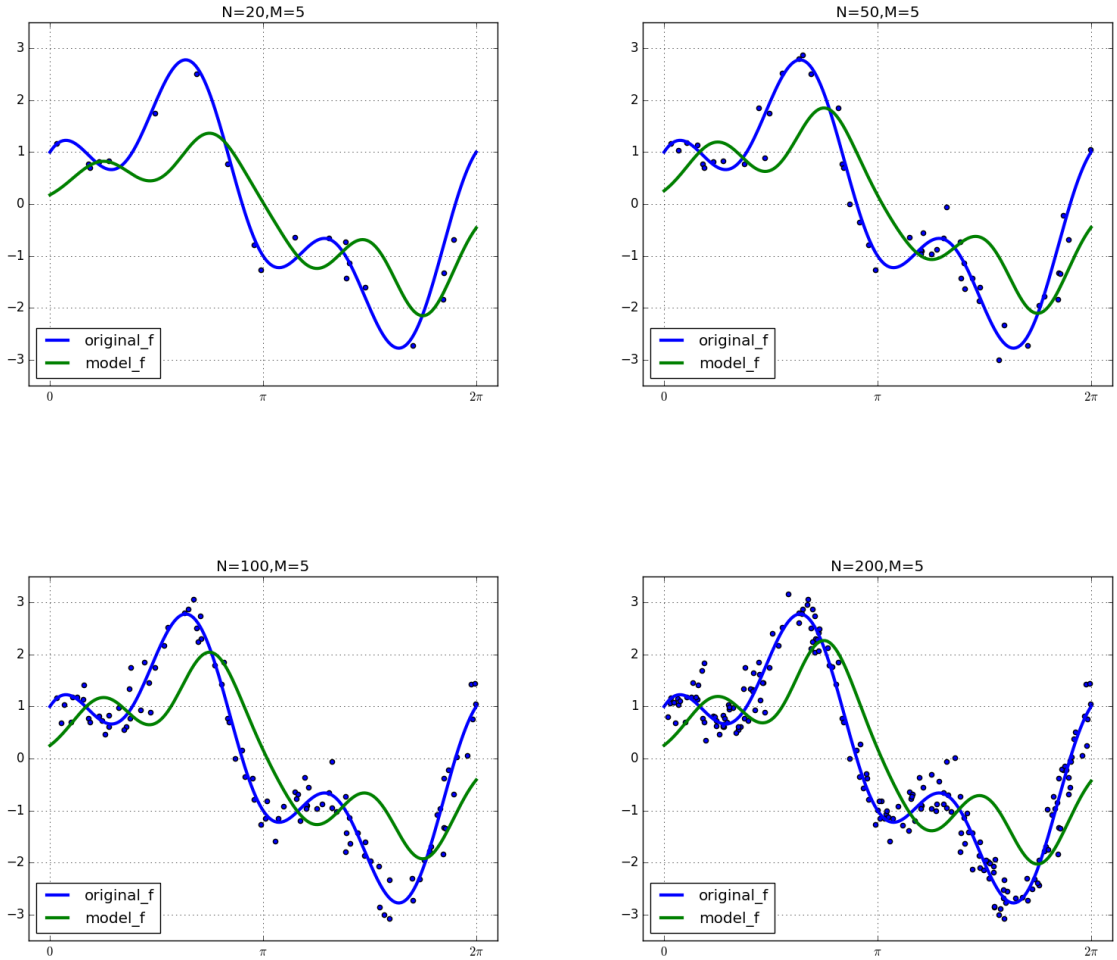
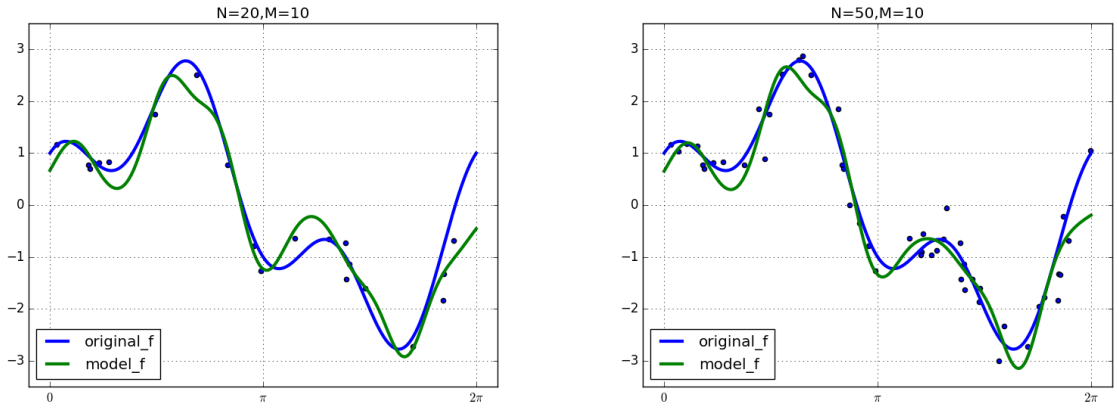
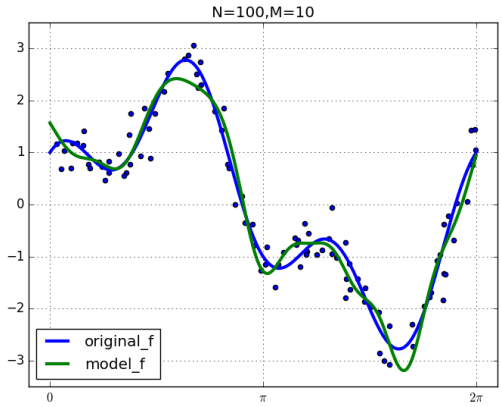


図 2: $M = 10, N = 20, 50, 100, 200$





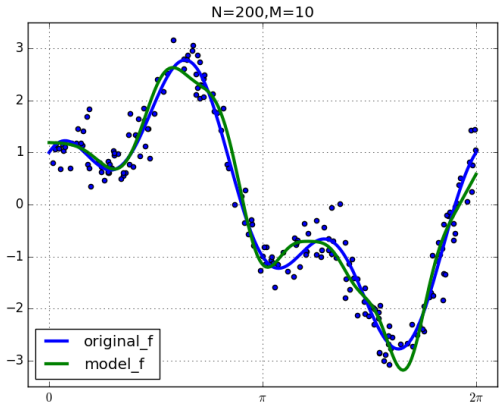
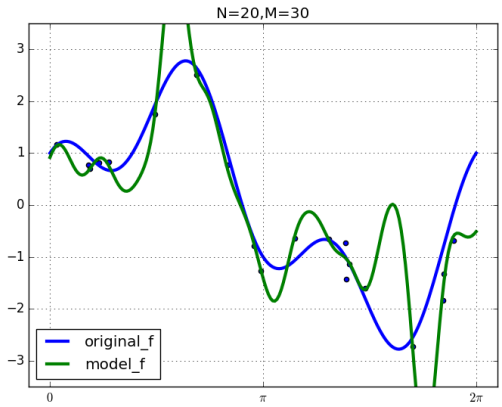
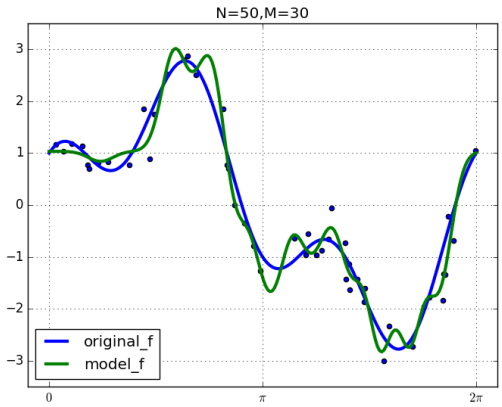
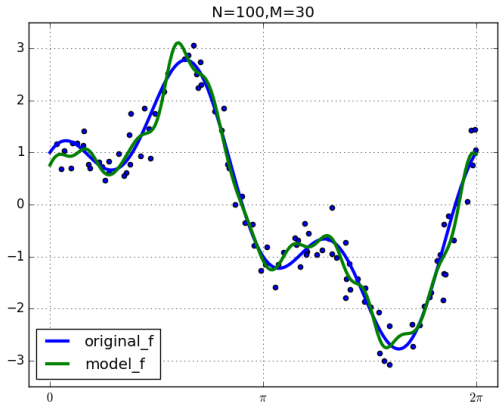


図 3: $M = 30, N = 20, 50, 100, 200$







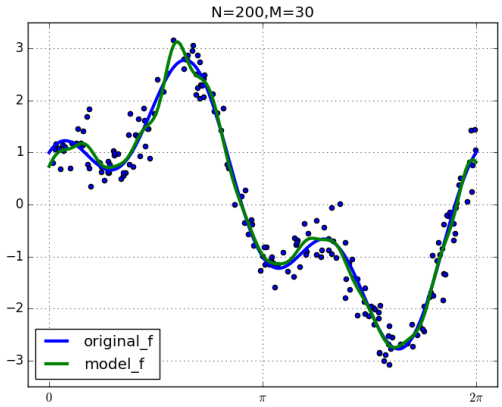
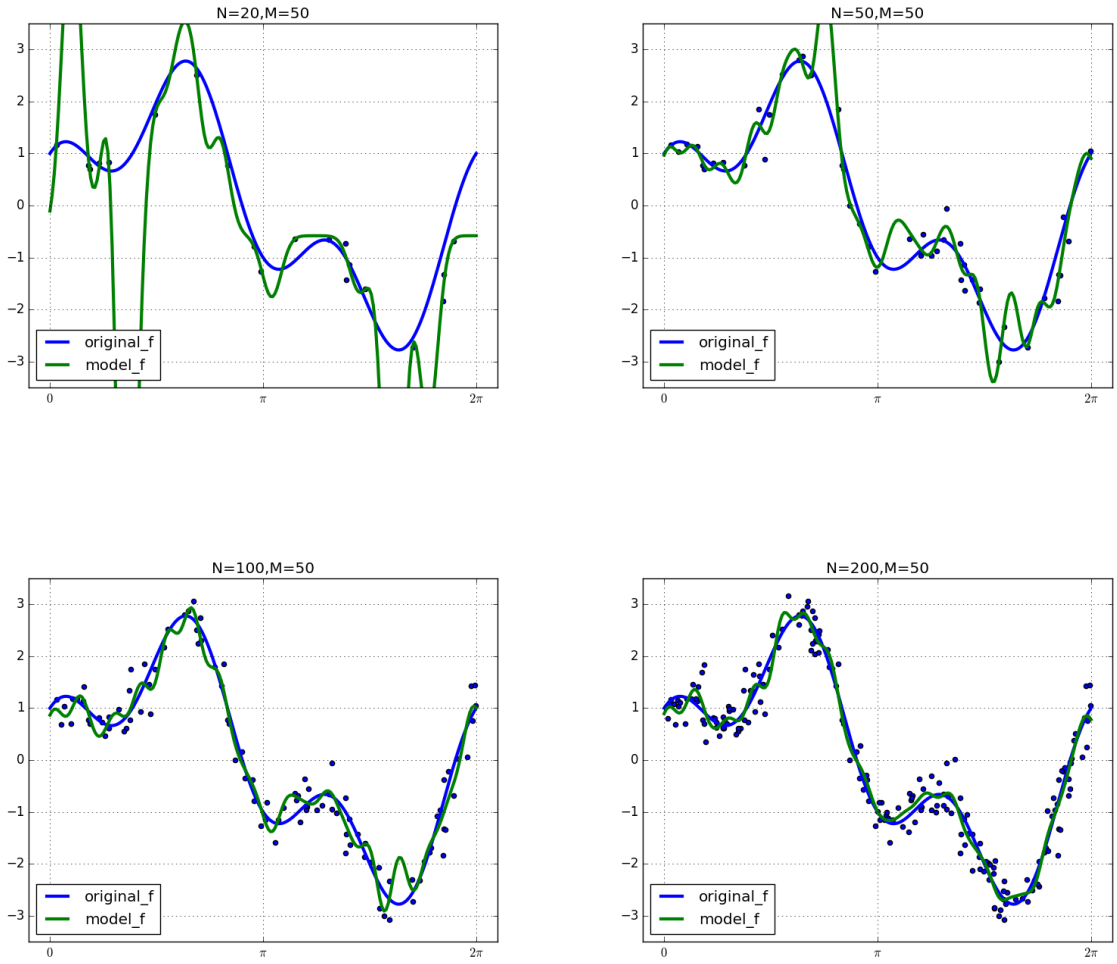


図 4: $M = 50, N = 20, 50, 100, 200$



M が小さいときなんかずれてる, M が大きいときうまくいっている. N が小さすぎるとき過学習気味になるが, そんなにひどくない

4.1.2 誤差

M \ N	20	50	100	200
5	0.90	0.90	0.92	0.92
10	0.47	0.39	0.36	0.36
30	0.96	0.36	0.29	0.28
50	2.58	0.60	0.34	0.30

表 1: E_{RMS} の N,M との関係 (ガウスカーネル)

4.1.3 パラメータ α, β

$N = 200, M = 50$ のとき

```
aplha=
[  1.03210215e+000   1.49000585e+002   1.47141069e+006   6.02614368e-001
   6.30596754e+004   1.25255259e+001   3.47664951e+012   1.85085088e+000
   4.30126648e+013   1.43348897e+008   5.26437717e-001   1.21340818e+014
   1.11325811e+006   1.65818835e-001   4.20639228e+001   6.87016589e-001
   3.54036047e-001   8.39562665e+002   5.79081271e-001   1.43347370e+000
   8.19627571e+011   1.34144803e+016   2.34625329e+011   0.00000000e+000
   1.48959166e+000   9.58470740e+001   1.29329993e+000   7.24692239e+010
   2.28664362e+000   2.09542540e+001   1.42045065e+008   2.26085804e+000
   1.04702641e+015   8.01543522e+016   9.58098158e-001   1.42506329e+001
   6.63228094e-001   5.05697092e+005   2.33233107e-001   1.38992358e+020
   2.91174331e-001   0.00000000e+000   2.64324376e-001   4.59027935e+007
   2.35069550e+000   8.74889763e+000   1.18464149e+008   4.06789084e+007
   1.31046021e+000   1.07236401e+173]
beta= 9.93096269807
```

となった.
α は絶対値が非常に大きくなる要素がいくつかある. RVM が効力を発揮している.
β は元データの精度 (分散の逆数)$1/0.3^2 \neq 11$ 程度になっていて正しいように思う.

4.2 多項式カーネル

カーネル関数に

$$k(\mathbf{x}_n, \mathbf{x}_m) = (x_n \mu_m + 1)^\theta$$

を用いた. ただし, データ数は $N = 50, 100$ $M = 10, 30$ とし $\theta = 2, 3, 7$ を用いた.

4.2.1 回帰結果

あまり面白い結果にならなかったので, 一部のみ記す.
$\theta = 2$ のとき

図 5: $M = 10, N = 50, 100$

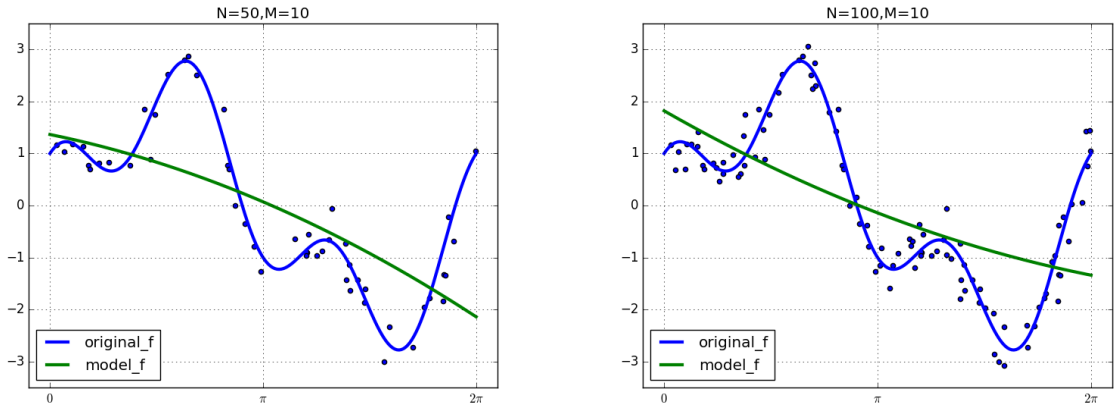
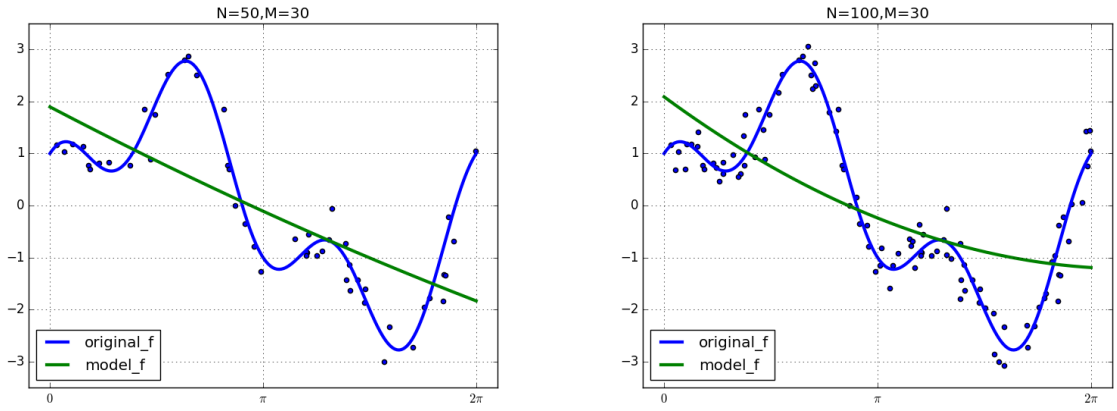


図 6: $M = 30, N = 50, 100$



カーネルに依存し 2 次関数となり, 回帰としては不十分.

$\theta = 3$ のとき

図 7: $M = 10, N = 50, 100$

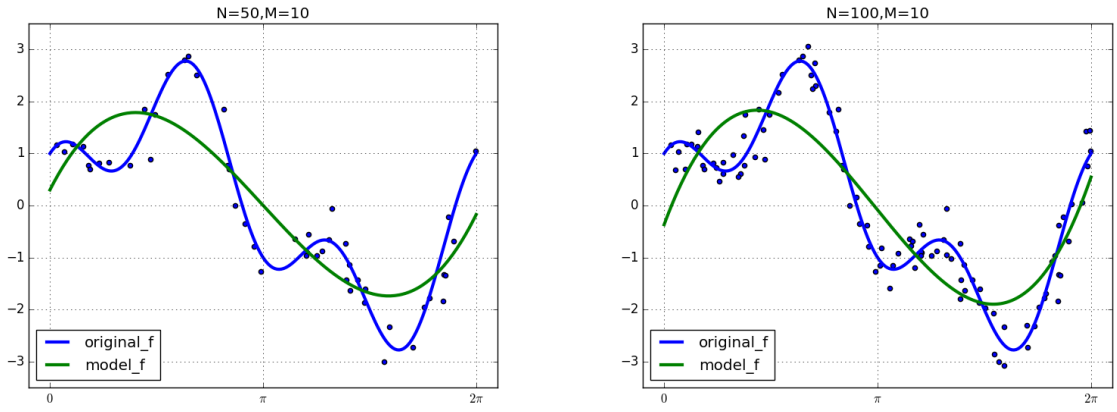
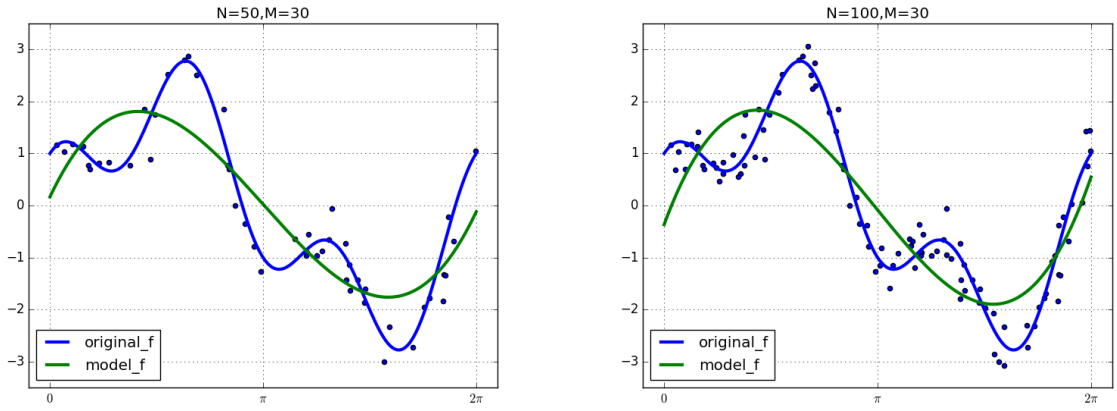


図 8: $M = 30, N = 50, 100$



カーネルに依存し 3 次関数となり, 回帰としては不十分.

$\theta = 7$ のとき

図 9: $M = 10, N = 50, 100$

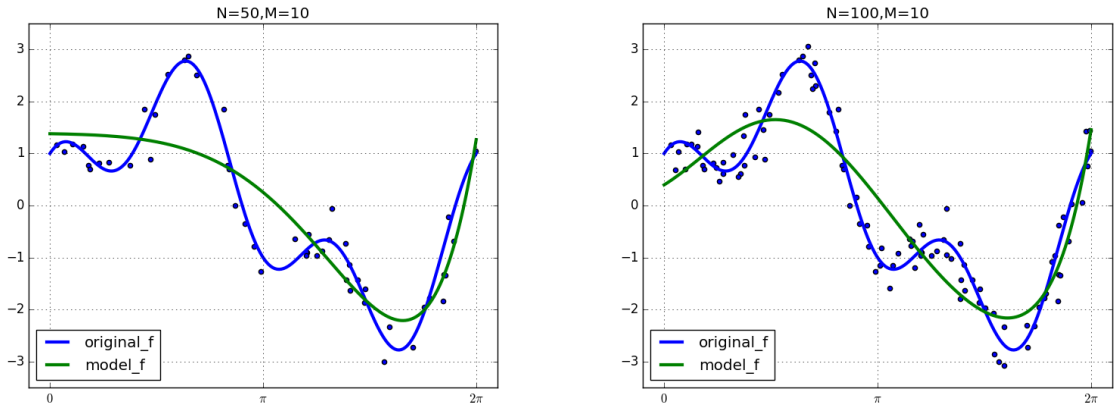
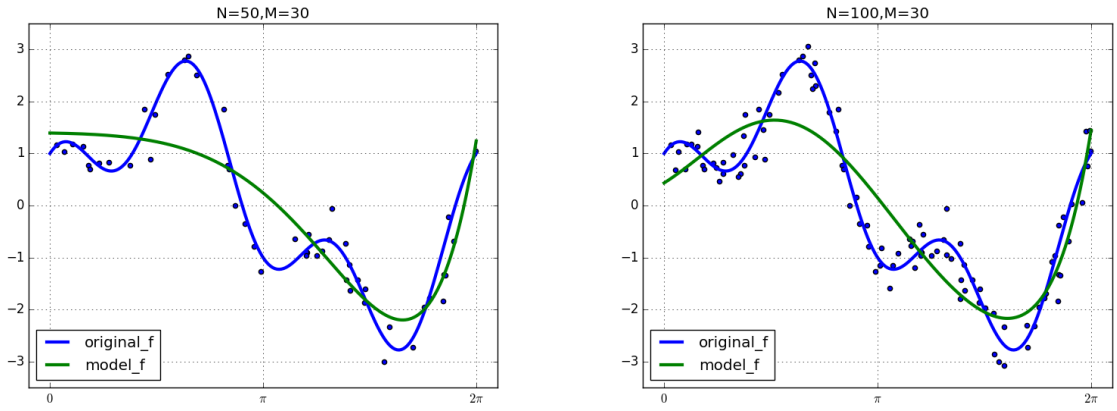


図 10: $M = 30, N = 50, 100$



山, 谷が 6 つなので, 7 次関数で精度よく回帰できると考えたが, 多項式カーネルは絶対値が大きい領域での回帰を優先するため, 絶対値の小さい領域では回帰に失敗している.

4.2.2 誤差

M \ N	20	50	100	200
5	1.17	1.18	1.21	1.19
10	1.17	1.68	1.21	1.19
30	1.15	1.15	1.21	1.19
50	960	4.27	1.53	1.19

表 2: E_{RMS} の N,M との関係 (多項式カーネル,$\theta = 2$)

M \ N	20	50	100	200
5	0.98	0.80	0.79	0.78
10	1.02	0.80	0.79	0.78
30	0.95	0.79	0.79	0.78
50	0.86	0.78	0.79	0.78

表 3: E_{RMS} の N,M との関係 (多項式カーネル,$\theta = 3$)

M \ N	20	50	100	200
5	0.81	0.77	0.76	0.73
10	0.83	0.76	0.69	0.69
30	0.84	0.76	0.69	0.70
50	0.84	0.74	0.69	0.89

表 4: E_{RMS} の N,M との関係 (多項式カーネル,$\theta = 7$)

カーネル関数の選び方で凡化性能に限界がある.

4.2.3 パラメータ α, β

$N = 200, M = 50, \theta = 7$ のとき

```
aplha=
[  1.18574425e-003   1.04139538e+001   3.31047546e-002   1.77882807e-001
  -3.47046296e+002  -4.61559521e+002   1.34322299e+000   2.79414730e+001
  -1.86062844e+010  -1.54567781e+003  -5.14047608e+002   3.10796253e+001
   1.04442733e+002   4.19904323e+009   8.05665954e+010  -2.31815061e+015
  -1.35174881e+011  -8.32538438e+014  -5.46911535e+015   9.69192473e+008
   1.69798816e+285   2.41211252e+010   6.40257997e+007  -3.78405631e+005
  -1.37882658e+013   4.17044230e+020  -2.33728519e+013  -9.73764337e+009
  -8.57214679e+011   1.03035660e+013  -3.35679197e+286   6.23052007e+008
   1.40643411e+021   1.12679081e+023   8.93861844e+007  -1.81746855e+019
  -1.13981069e+013  -1.74536866e+018  -5.66009253e+018   2.52728947e+015
  -8.56284273e+017  -1.85211323e+287   1.78936718e+017   6.42057977e+015
   7.43593241e+013   9.49429315e+017  -1.21732689e+287   1.78638340e+010
  -1.00451932e+014   1.60723899e-003]
beta= 1.08546893411
```

となった.
ガウスカーネルのとき同様, α は絶対値が非常に大きくなる要素がいくつかある. RVM が効力を発揮している.
β は元データの精度 (分散の逆数)$1/0.3^2 \neq 11$ より小さくなっていて, つまり分散が大きくなっていて, 回帰は失敗していると考える.

5　まとめ

α の一部の要素が非常に大きくなることが確かめられた, RVM のこの性質は疎な解を得るのにつながっている.