2021年度　情報理工学実験I

実験テーマ：データサイエンス

実験日：

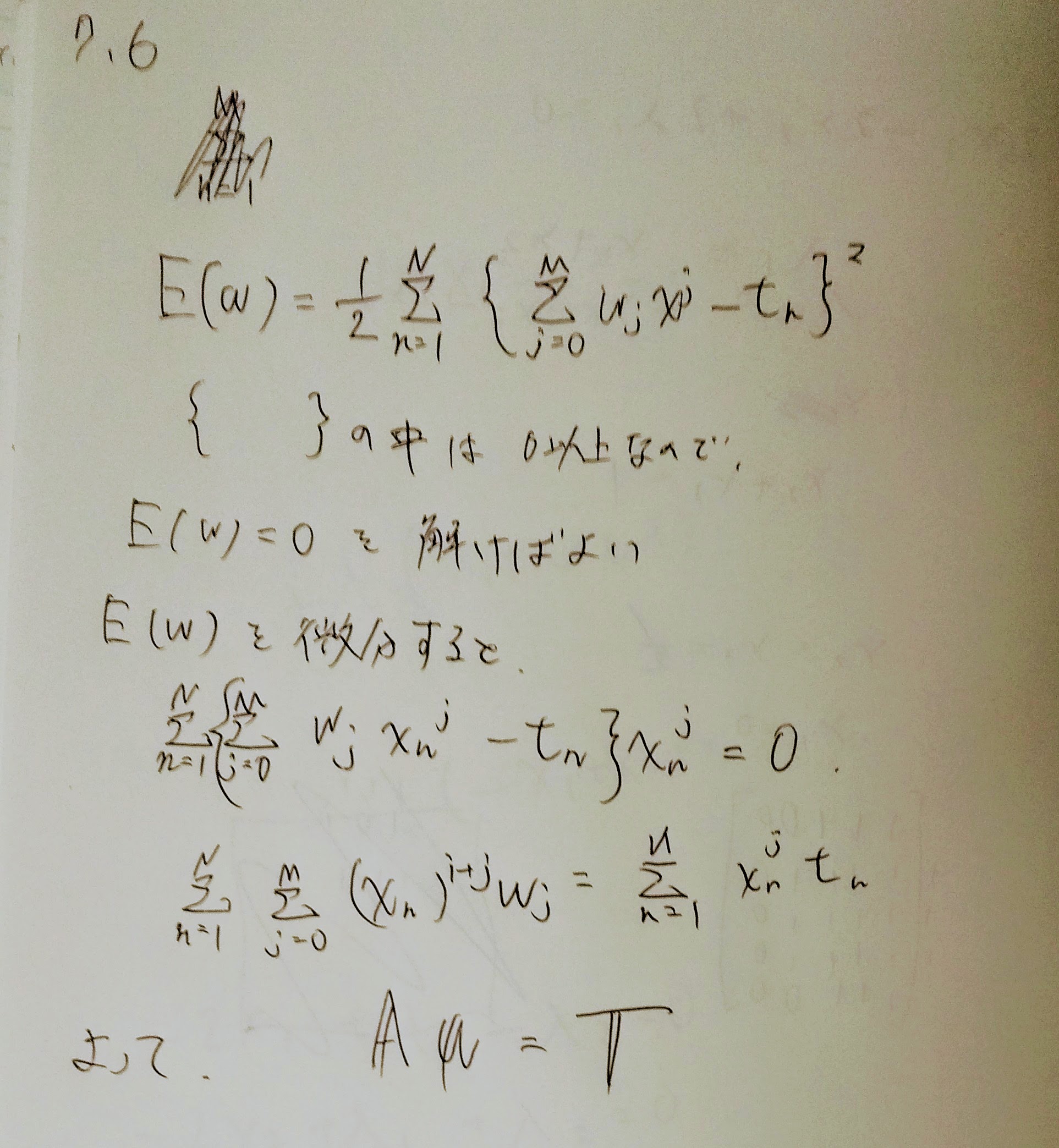
氏名：　高柳奏和

学生番号：　02190367

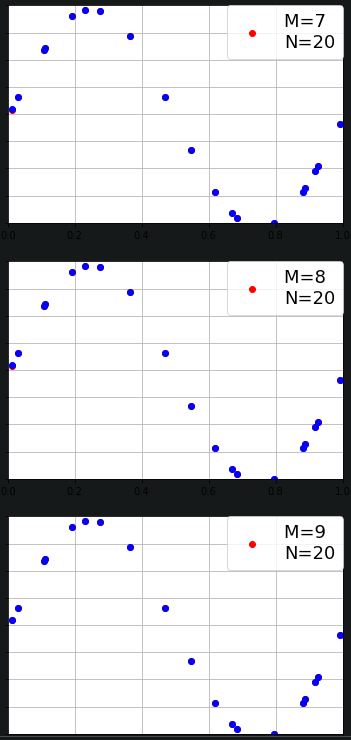
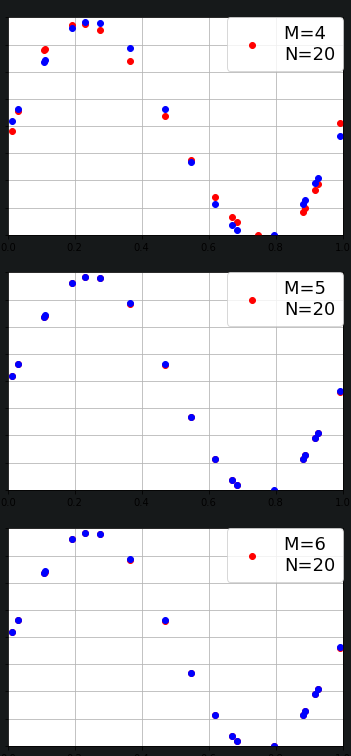
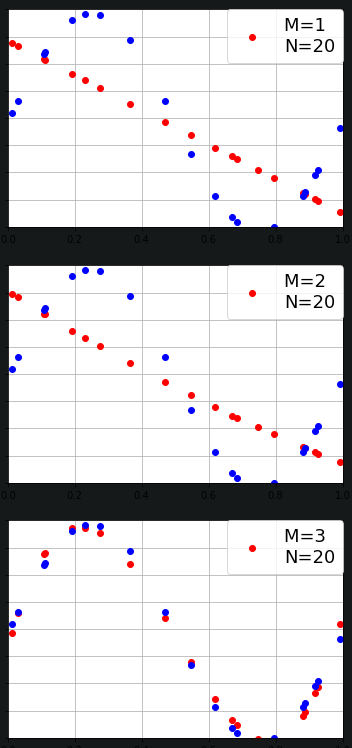
班名：

提出日：8/4

7.6



7.9

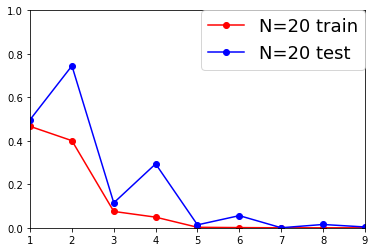


多項式の次数が上がり、精度が良くなったため、重なる部分が多くなったと考えられる。

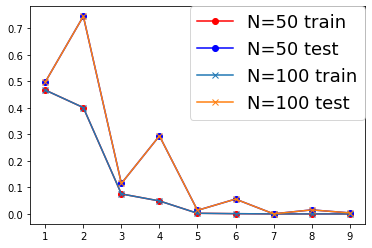
7.10

次数を増やしていくと、データファイルから描画したグラフと多項式のグラフが重なる部分が多くなり、100で数個の点を除いてほとんど一致した。

7.15



7.16

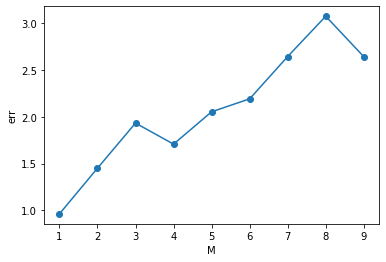


次数を増やすと誤差が小さくなった。これは、多項式からもわかるように、次数が増えて精度が上がり、誤差関数にかけたときにほとんどデータとの差がなくなったからだと考えられる。

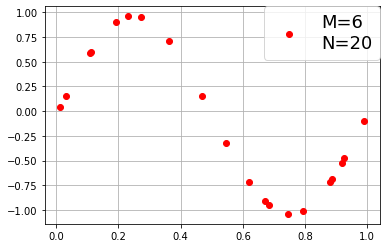
また、データ数を増やすと、次数が小さいときの誤差が、データ数が多いときのほうが大きくなった。これはデータ数が増え、誤差が大きくなるようなデータを使う確率が大きくなるからだとかんがられる（？）

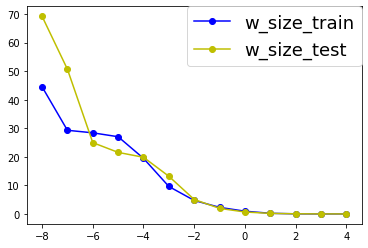
7.20

式もあっているはずだが、次数が上がるほど誤差が大きくなる理由が思いつかなかった



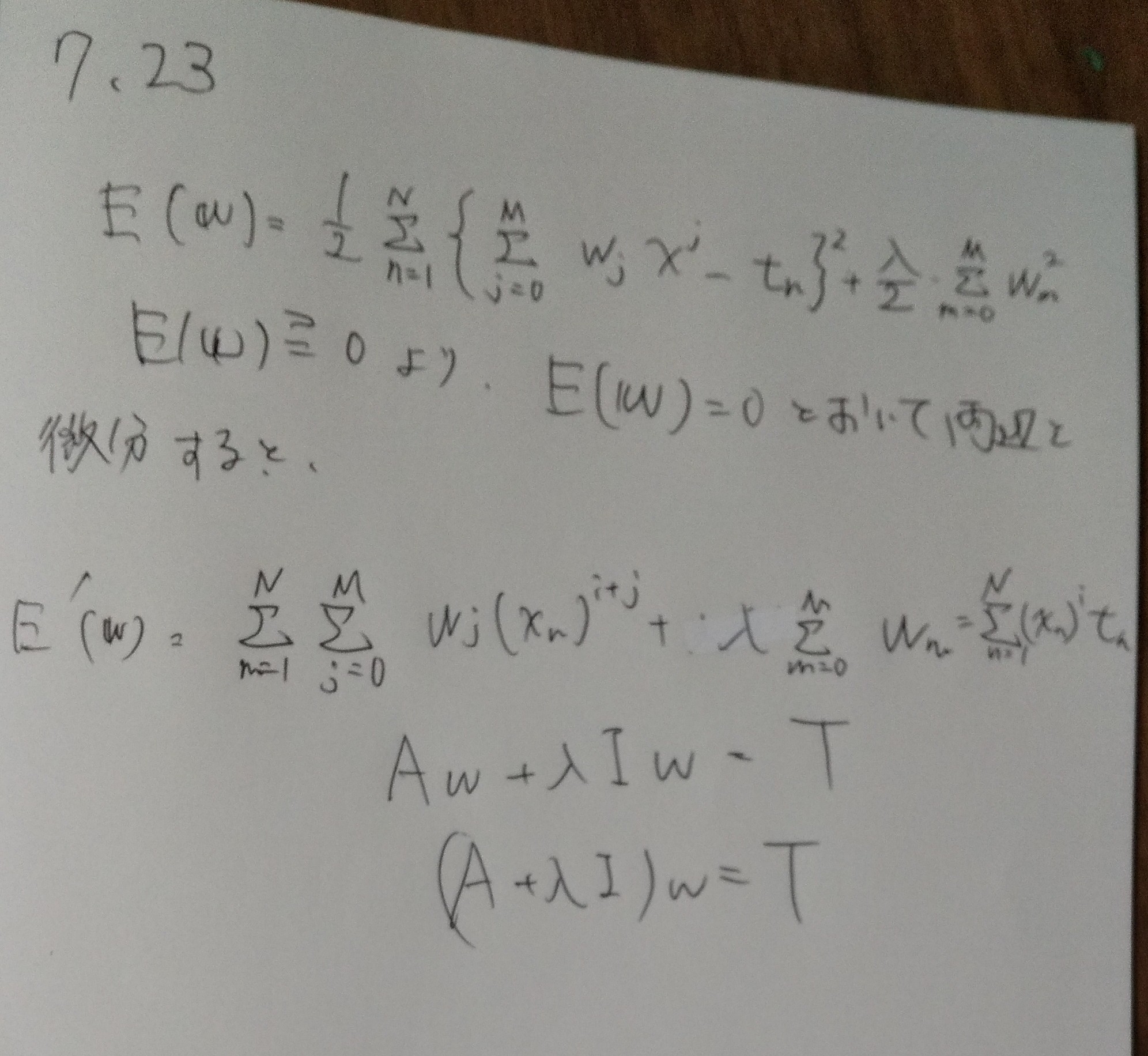
7.22





7.23

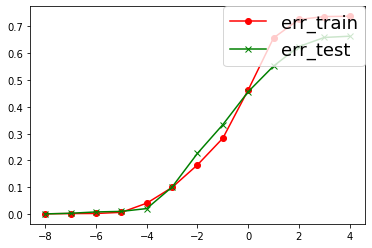
証明

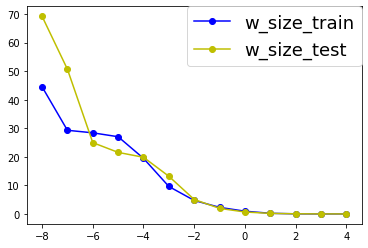


7.23

7.25

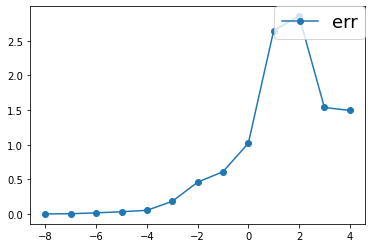
λが小さいほど元のデータとの乖離が小さいママ最適化できるのだと考えられる





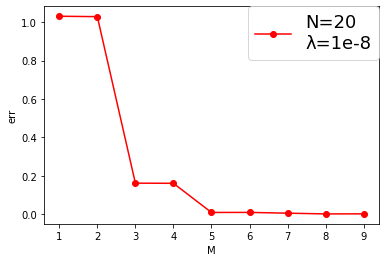
7.26

λが大きいほど、多項式を作る際、λがないときとの差が大きくなり、誤差が大きくなるのだと考えられる



7.28

次数が上がり、精度が上がったのでこの形になったと考えられる



7.21コード

|  |
| --- |
| def opt\_m():  min\_err=10000  for m in range(1,10):  err=cross\_validation(m)  if err<min\_err:  min\_err=err  return min\_err |

|  |
| --- |
| def cross\_validation(M):  mean\_erms=[]  for \_ in range(5):  err=split\_and\_test(M)  mean\_erms.append(err)    return (np.sum(mean\_erms)/5) |

|  |
| --- |
| def split\_and\_test(M):  \_w=minimize\_erms(M, 20, 10)  train,test=separate\_data()  x\_train, y\_train=train[:,0], train[:, 1]  x\_test, y\_test=test[:, 0], test[:, 1]  test\_err=calc\_erms(M, x\_train, y\_train, \_w)  return test\_err |

|  |
| --- |
| def separate\_data():  a=generate\_rand\_int()  data=np.loadtxt('N-20.txt')  data\_train=data[a!=5]  data\_test=data[a==5]  return [data\_train,data\_test] |

|  |
| --- |
| def generate\_rand\_int():  a=np.empty((1,20))  while True:  f=False  a=np.random.randint(1,6,20)  for i in range(1,6):  if i not in a:  f=True  break  if f: continue  else: return a |

7.27

コード

|  |
| --- |
| def calc\_opt\_lambda():  min\_err=100000000  opt\_lam=0  for i in range(-8,5):  lam=10\*\*i  err=cross\_validation\_reg\_split(lam) #5分割  if err<min\_err:  min\_err=err  opt\_lam=lam  return opt\_lam |

|  |
| --- |
| def cross\_validation\_reg\_split(lam):  mean\_erms=[]  for \_ in range(5):  err=split\_and\_test\_reg(lam)  mean\_erms.append(err)    return (np.mean(mean\_erms)) |

|  |
| --- |
| def split\_and\_test\_reg(lam):  train,test=separate\_data()  x\_train, t\_train=train[:,0], train[:, 1]  x\_test, t\_test=test[:, 0], test[:, 1]  test\_err=calc\_erms\_reg(9, x\_train, t\_train, train, lam)  return test\_err |

|  |
| --- |
| def calc\_erms\_reg(M,x,t,data,lam):  N=len(t)  A=generate\_matrix(M,x)  T=generate\_vector(M,data)  w=solve\_rpr(A,lam,T)  y=calc\_polynomial(M,x,w)  sum=0  for n in range(N):  sum+=np.power(y[n]-t[n], 2)  return np.sqrt(sum)/2 + np.power(np.abs(np.sum(w)), 2)\*lam/2 |

|  |
| --- |
| def solve\_rpr(A,lam,T):  N=len(T)  return np.linalg.solve(A+lam\*np.eye(N), T) |

講義を通して

numpyやmatplotlibの基本的な操作が身についた。また、この長さのコードを書いたのは初めてだったので、変更を考慮したコードを書くスキルが少し伸びた。