Use MQHOA TO Fit Parameters

Lu ZhiJun

2015 年 12 月 13 日

1 误差评判方法研究

1.1 $y = x^2$ 傅里叶级数展开

如果 $f(x)$ 在 [-l,l] 上分段连续，其傅里叶级数

$$f(x)\ \sum_{k=1}^{\infty}(a_k cos\frac{k\pi}{l}x + b_k sin\frac{k\pi}{l}x) + \frac{a_0}{2}$$

那么 $y = x^2$ 在 $[-l, l]$ 傅里叶级数展开：

$$f(x) = 4\sum_{k=1}^{\infty}(-1)^k(\frac{l}{k\pi})^2 cos(\frac{k\pi}{l}x) + \frac{l^2}{3}$$

也可写成：

$$f(x) = \sum_{k=0}^{\infty} a_k cos(\frac{k\pi}{l}x)$$

其中

$$a_k = \begin{cases} \frac{l^2}{3} & \text{k=0} \\ (-1)^k(\frac{2l}{k\pi})^2 & \text{k=1,2...} \end{cases}$$

又 $f(x)$ 在 $[-l, l]$ 上连续，又 $f'(x)$ 绝对可积，有：

$$f'(x) = \frac{A_0}{2} + \sum_{k=1}^{\infty}(A_k cos(\frac{k\pi}{l}x) + B_k sin(\frac{k\pi}{l}x))$$

利用求系数公式和分部积分可证明：

$$\begin{cases} A_k = \frac{k\pi}{l}b_k + (-1)^k c & \text{k=0,1,2...} \\ B_k = -\frac{k\pi}{l}a_k & \text{k=1,2...} \\ c = \frac{1}{l}[f(l) - f(-l)] \end{cases}$$

由于 $f(l) = f(-l)$, 则 $f'(x)$ 的傅里叶级数可以通过对 $f(x)$ 的逐项求导而得，即

$$f'(x) = 4\sum_{k=1}^{\infty}(-1)^{k+1}(\frac{l}{k\pi})sin(\frac{k\pi}{l}x) = \sum_{k=1}^{\infty} -\frac{k\pi}{l}a_k sin(\frac{k\pi}{l}x)$$

1.2 构造评价函数

对参数 a_k 进行拟合，取 $k=[0,1,...D-1]$, 即评价函数维度为 D 维实验数据集
* $S1=\{(x,y)|y=x^2+noraml(0,2),x=linspace(-l,l,N)\}$
* $S2=\{(x,y')|gaussian\_filter(diff(y)),x=linspace(-l,l,N-1)\}$

其中 normal 为正态分布函数 (为模拟工程测量误差),linspace 为线性分布函数;diff 为差分函数,gaussian_filter 对差分结果作高斯平滑处理. 为保证算法对 $f(x)$ 与 $f'(x)$ 都有较好拟合，构造评价函数.

$$G(a_0,a_1,...a_{D-1})=MSE(f(a_0,a_1,...a_{D-1},x))+MSE(f'(a_1,...a_{D-1},x))$$

$$=\frac{\sum_{i=0}^{N-1}(\frac{y_i-f(a_0,a_1,...a_{D-1},x_i)}{y_i})^2}{N}+\frac{\sum_{i=0}^{N-1}(\frac{y'_i-f'(a_1,...a_{D-1},x_i)}{y'_i})^2}{N}$$

转化为求 $minG(a_0,a_1,...a_{D-1})$ 的解.

1.3 实验

实验假设 $l=5,D=20$, 数据集采样点数为 50 可根据上面公式计算 a_k

```
In [1]: import pandas as pd
        import math
        from pandas import Series, DataFrame
        %pylab inline
        plt.rc('figure', figsize=(12, 8))
        l=5
        D=20
        Ak=[l**2/3.0]
        for i in range(D-1):
            Ak.append((-1)**(i+1)*(2*l/(i+1.0)/math.pi)**2)
        print Ak
```

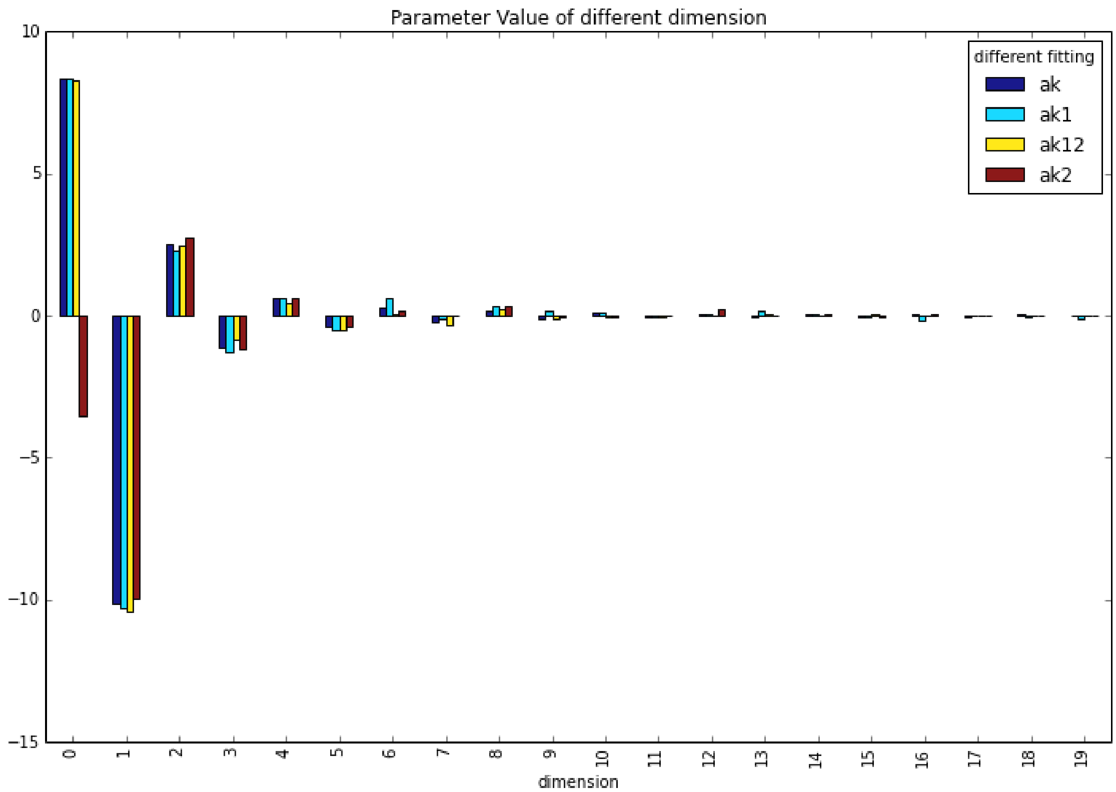
```
Populating the interactive namespace from numpy and matplotlib
[8.333333333333334, -10.13211836423378, 2.533029591058445, -1.125790929359309, 0.6332573977646112,
```

另外用智能算法根据 MSE(f(a_0,a_1,...a_{D-1},x)) 进行拟合，得到最优解记作 a_{k1}，用智能算法根据 MSE(f'(a_1,...a_{D-1},x)) 进行拟合，得到最优解记作 a_{k2}，联合以上两种 MSE 进行拟合，得到最优解记作 a_{k12}

下面图显示采取不同策略计算的最优解与 Ak 的对比

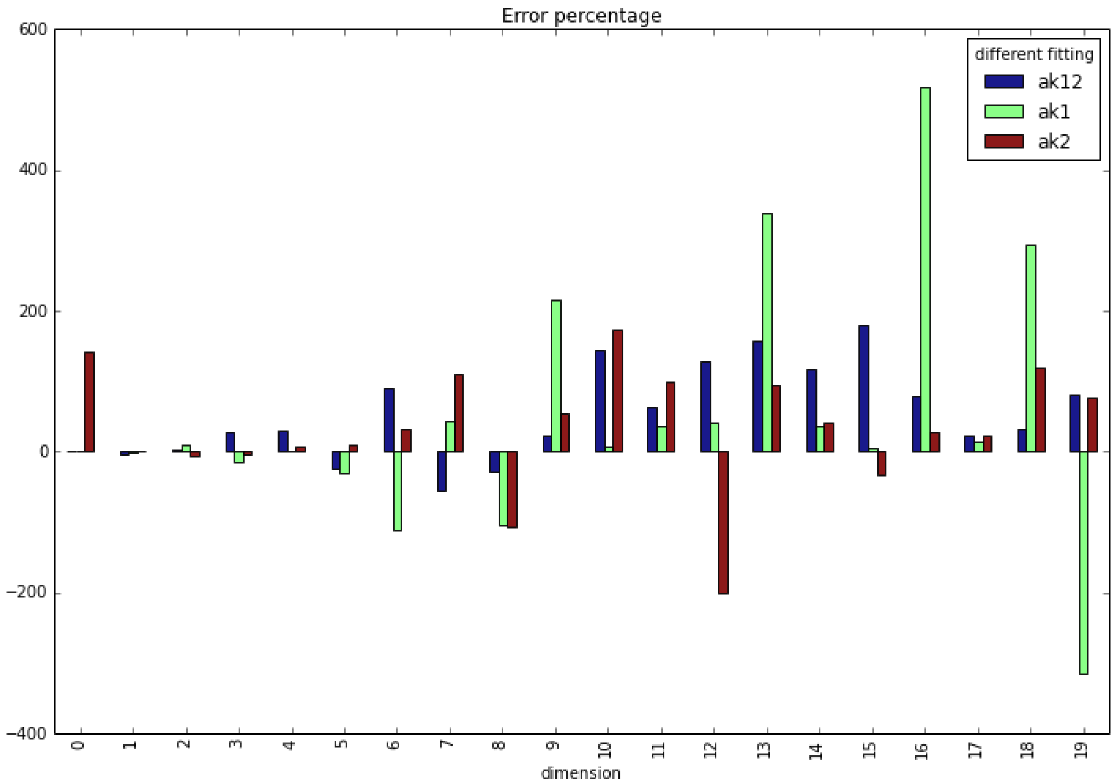
```
In [2]: ak1=[8.33281219 ,-10.318981, 2.27375948,-1.28375446, 0.6257736,-0.53357938, 0.59340663,-0.
        ak2=[-3.54610586e+00,-9.98962044e+00 ,2.71518383e+00,-1.17027024e+00, 5.93715316e-01,-3.69
```

```
ak12=[ 8.29574732e+00,-1.04340274e+01, 2.43998920e+00,-8.19756871e-01, 4.50149664e-01,-5.0
df=DataFrame({'ak':Ak,'ak1':ak1,'ak2':ak2,'ak12':ak12})
df.columns.name='different fitting'
df.index.name='dimension'
df.plot(kind='bar',stacked=False,colormap='jet',alpha=0.9,title='Parameter Value of differ
```

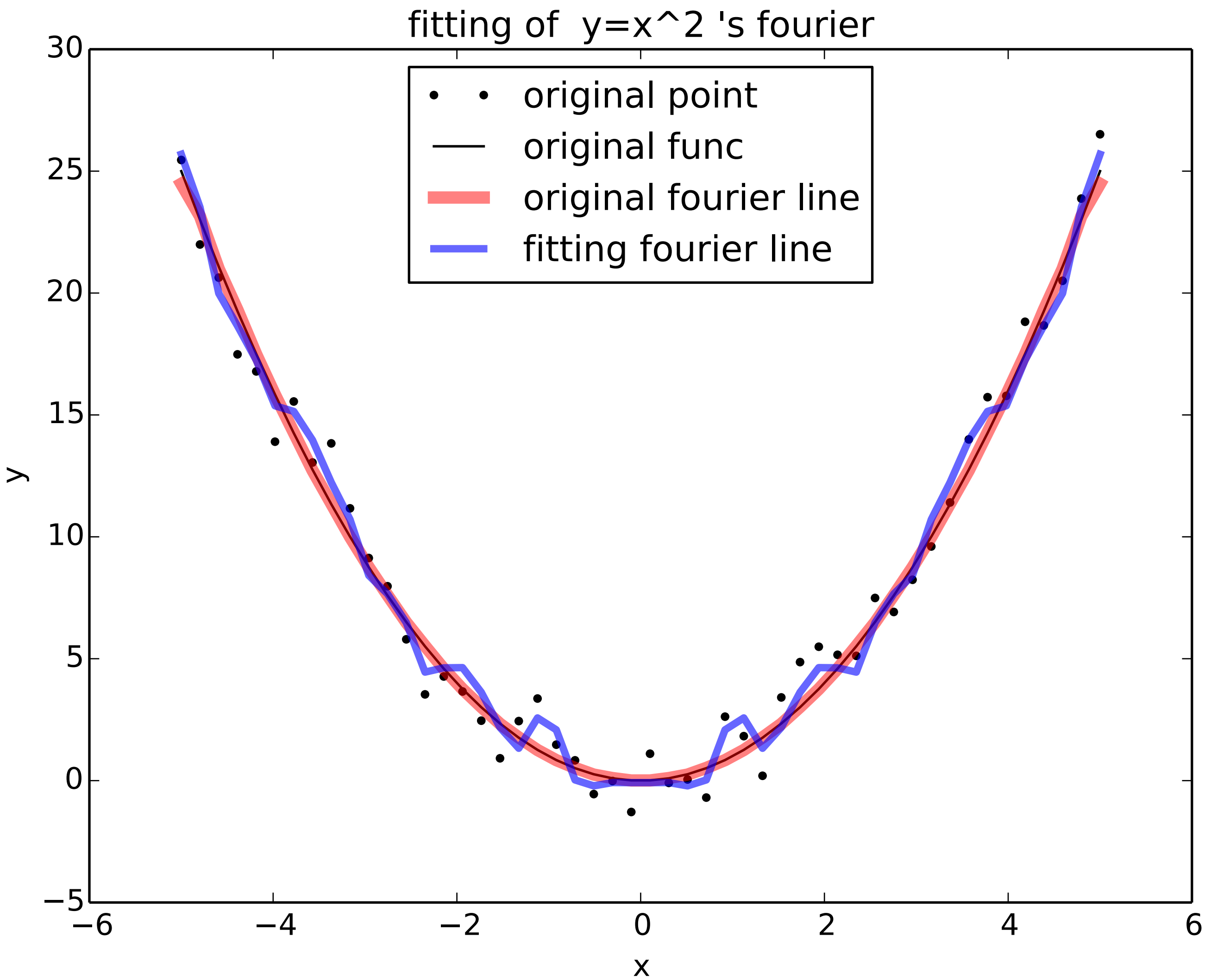


为对比方便，下图展示三种策略下的最优解在各维度上相对 ak 的误差百分比

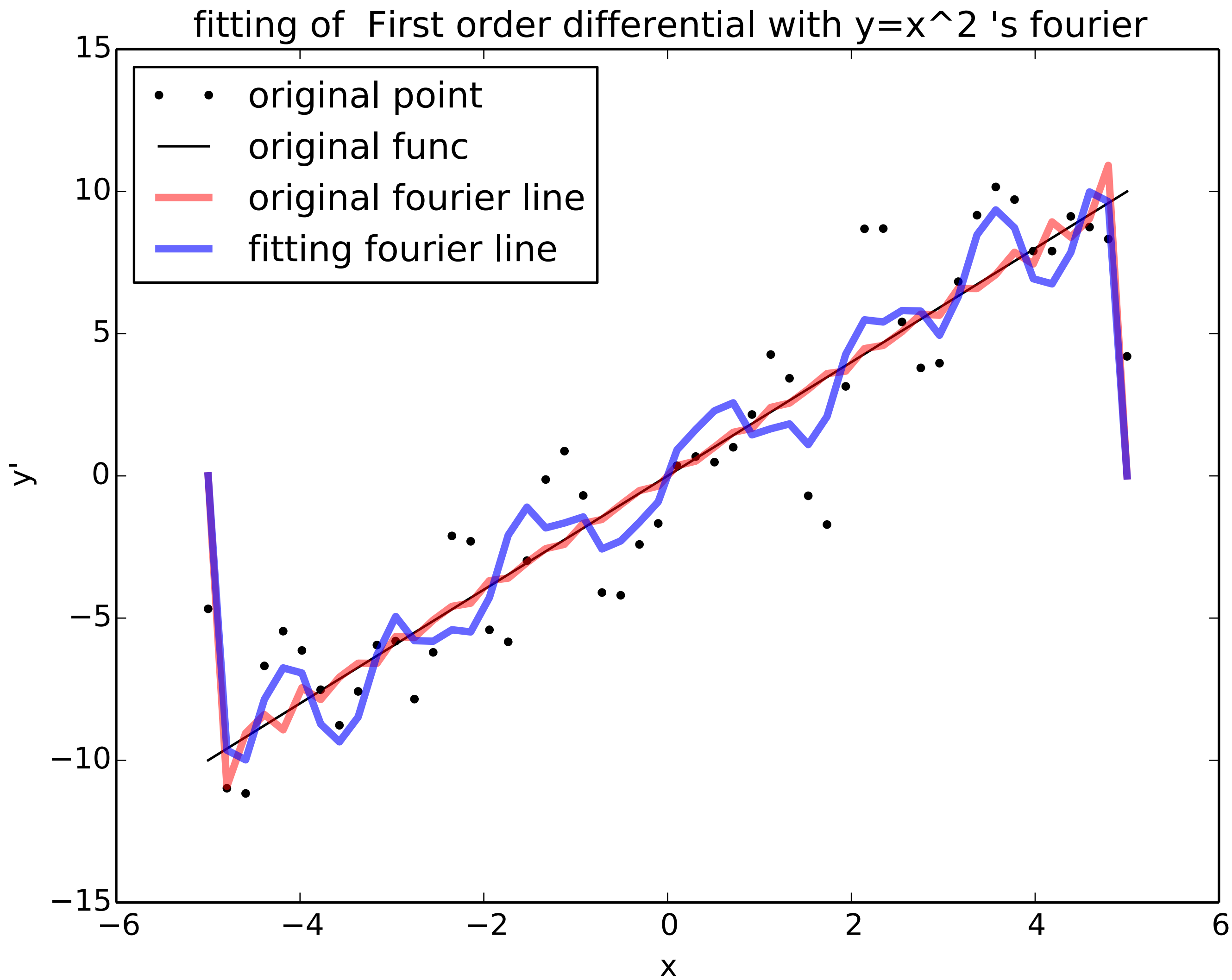
```
In [3]: b=df[['ak12','ak1','ak2']]
        c=df['ak']
        d=b.rsub(c,axis=0)
        e=d.div(c,axis=0)*100
        e.plot(kind='bar',stacked=False,colormap='jet',alpha=0.9,title='Error percentage');
```



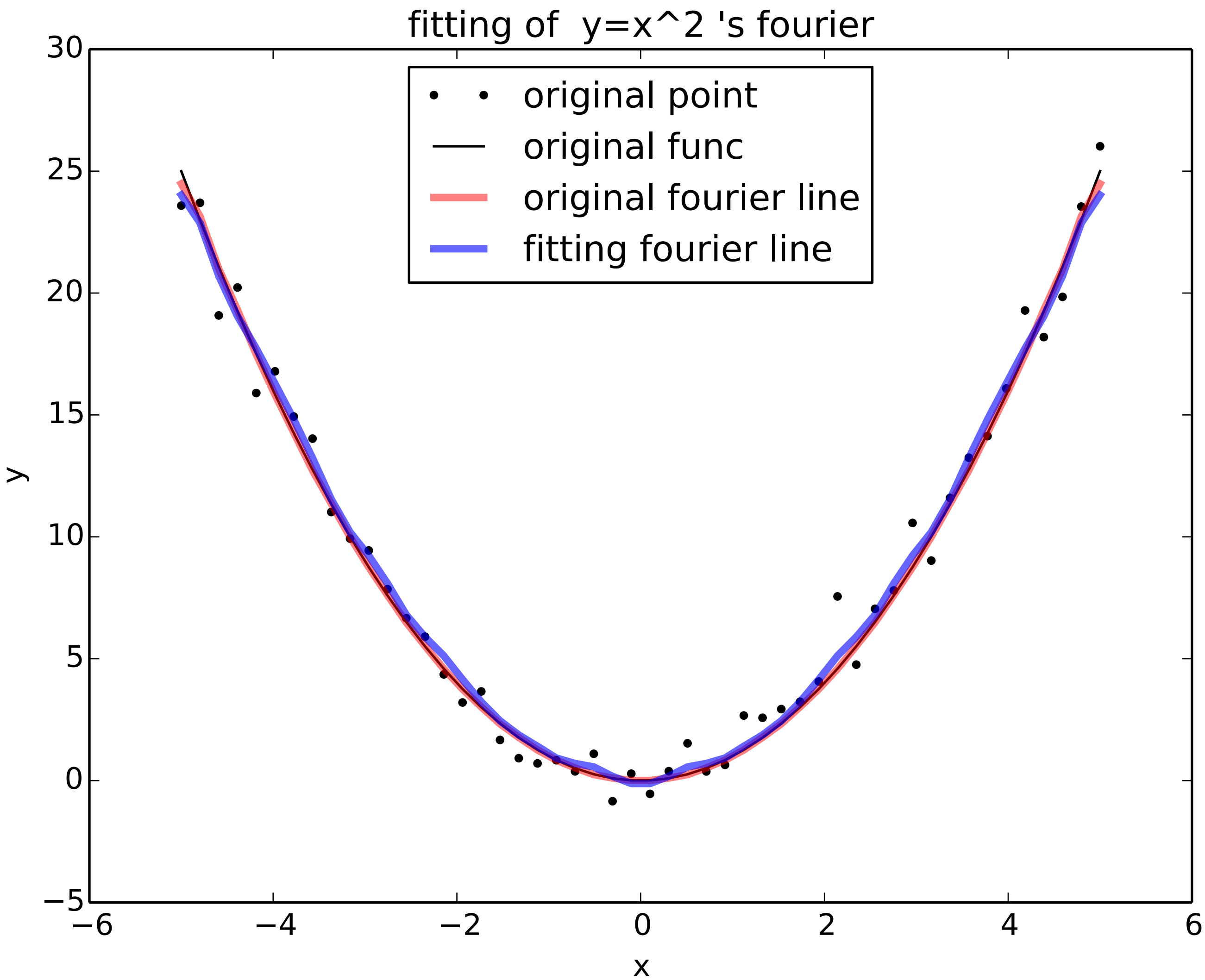
其中 $ak2$ 不对 a_0 进行拟合，其值没有参考意义。对比各图，可见 ak12 的相对误差要比 ak1 与 ak2 都要小. 以下为 ak1 拟合结果比较图，其中包括 $S1$ 数据点，x^2 原始函数与傅里叶展开的图像，以及拟合 ak1 参数的傅里叶展开图.



ak2 拟合结果比较图，其中包括 $S2$ 数据点，$2x$ 原始函数与傅里叶展开的图像，以及拟合 ak1 参数的傅里叶展开图

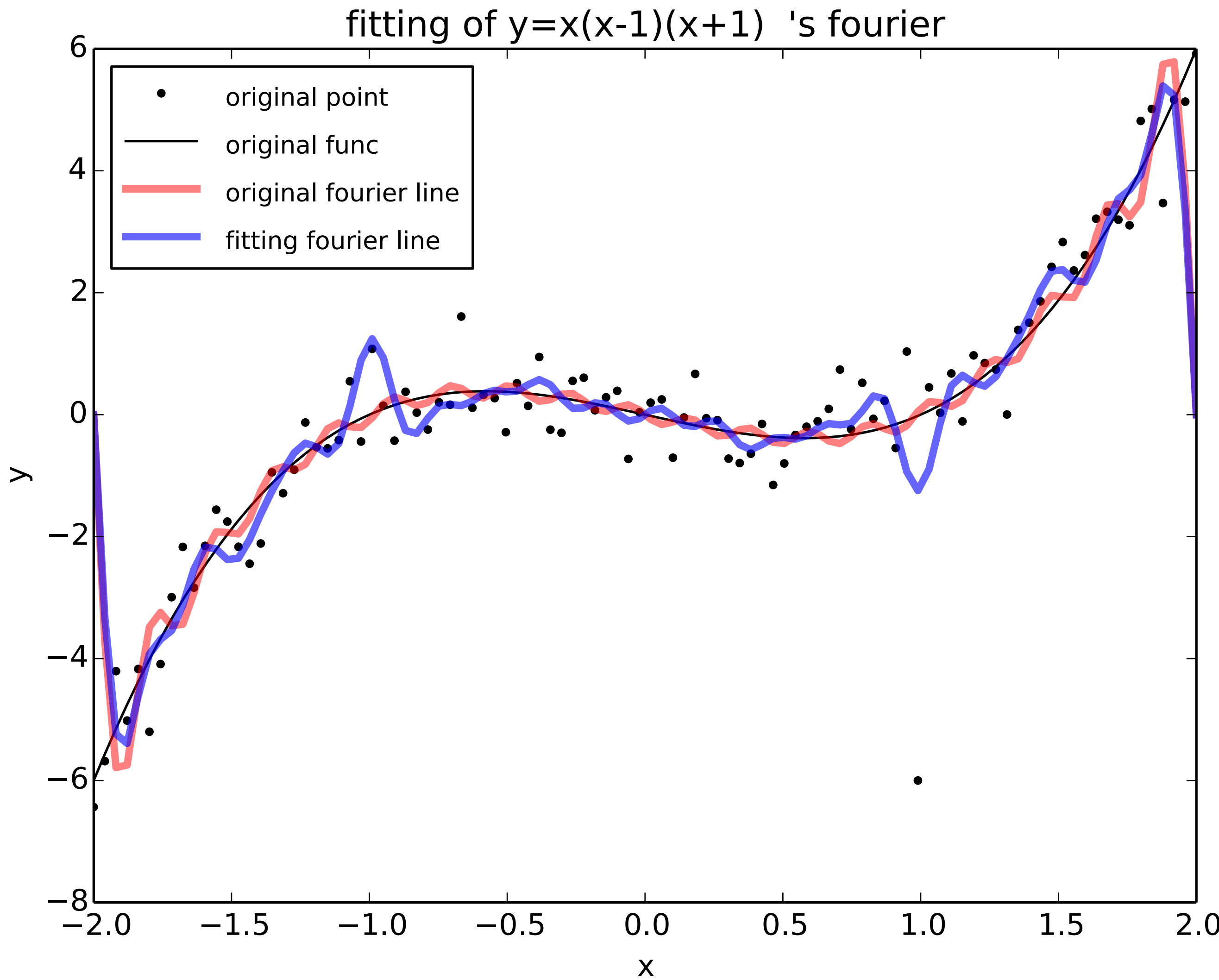


ak12 拟合结果比较图，各参数与第一幅图类似

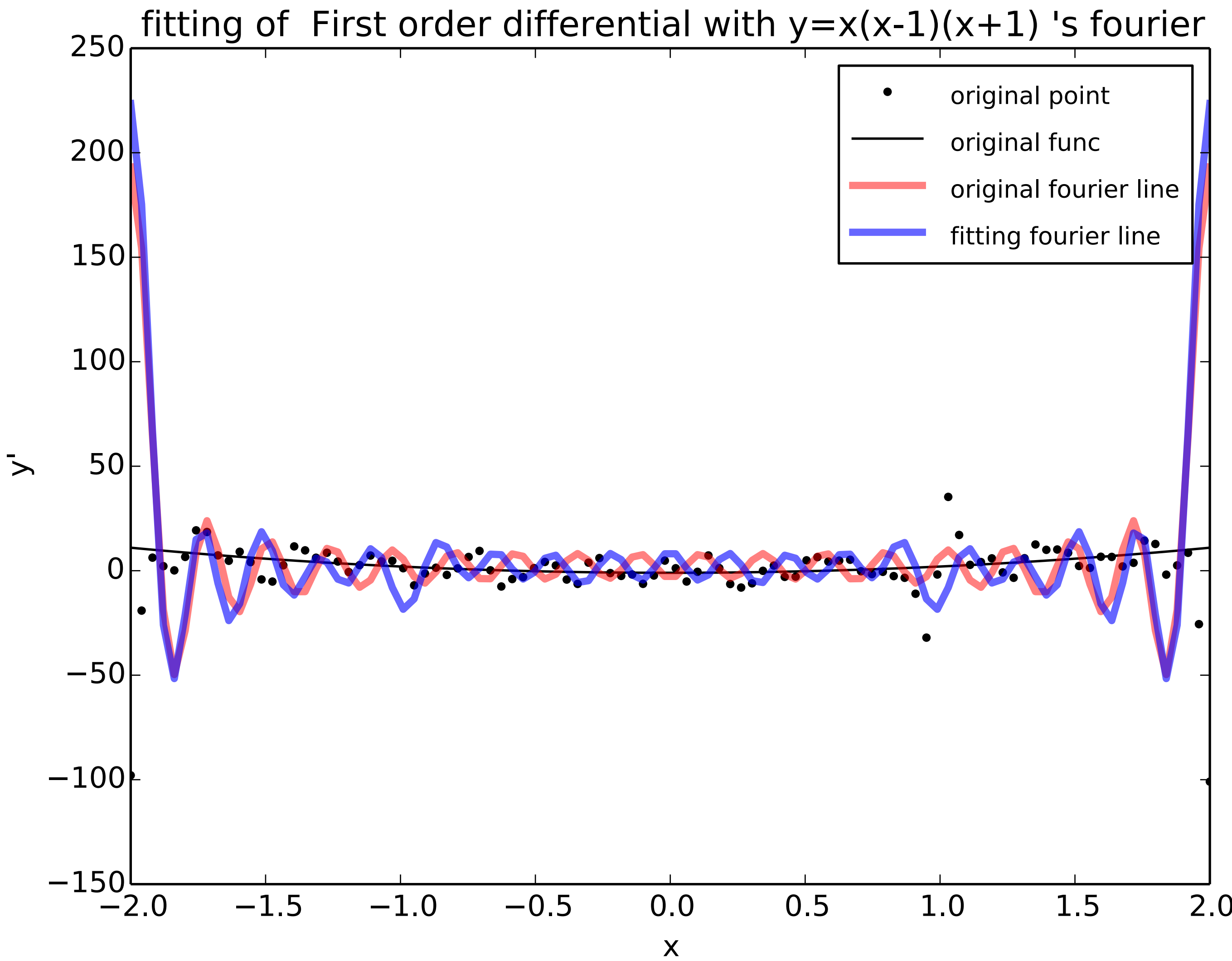

fitting of y=x^2 's fourier
original point
original func
original fourier line
fitting fourier line
x
y
30
25
20
15
10
5
0
−5
−6
−4
−2
0
2
4
6

1.4 测试函数：$y = x(x-1)(x+1)$

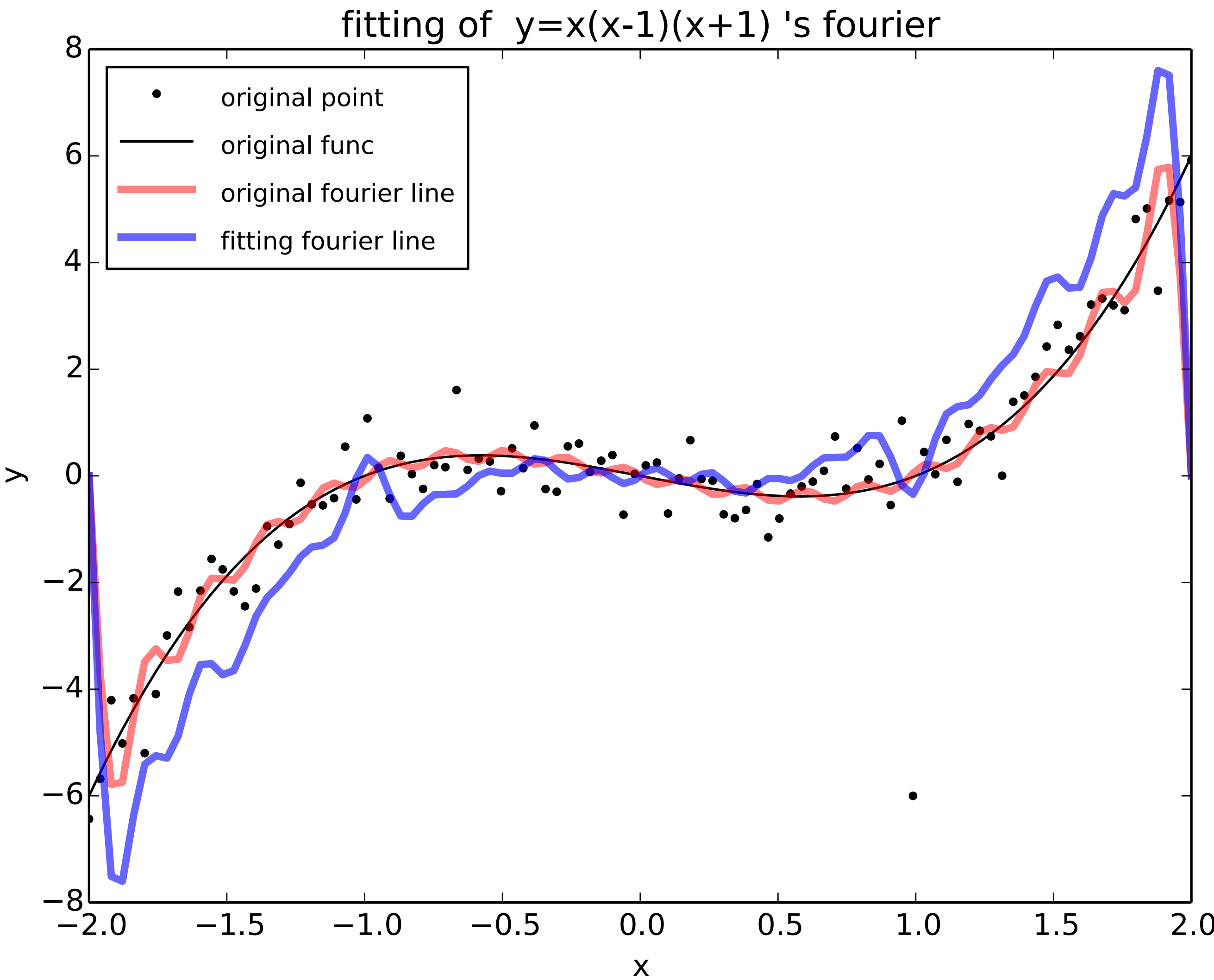
把 $y = x^2$ 替换成 $y = x(x-1)(x+1)$，N=100，增加噪点 (1,-6), 重复上述内容，可得



1.


fitting of First order differential with y=x(x-1)(x+1) 's fourier
original point
original func
original fourier line
fitting fourier line
y'
x
250
200
150
100
50
0
−50
−100
−150
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

2.



3.

很显然，这时候 ak12 拟合结果比 ak1 要差. 其原因在于第二幅图强调对原数据点拐点的拟合，权重集中于拐点处，因而两侧的权重较小。为了使第二幅图的影响权重在合理范围内，需要重构评价函数.

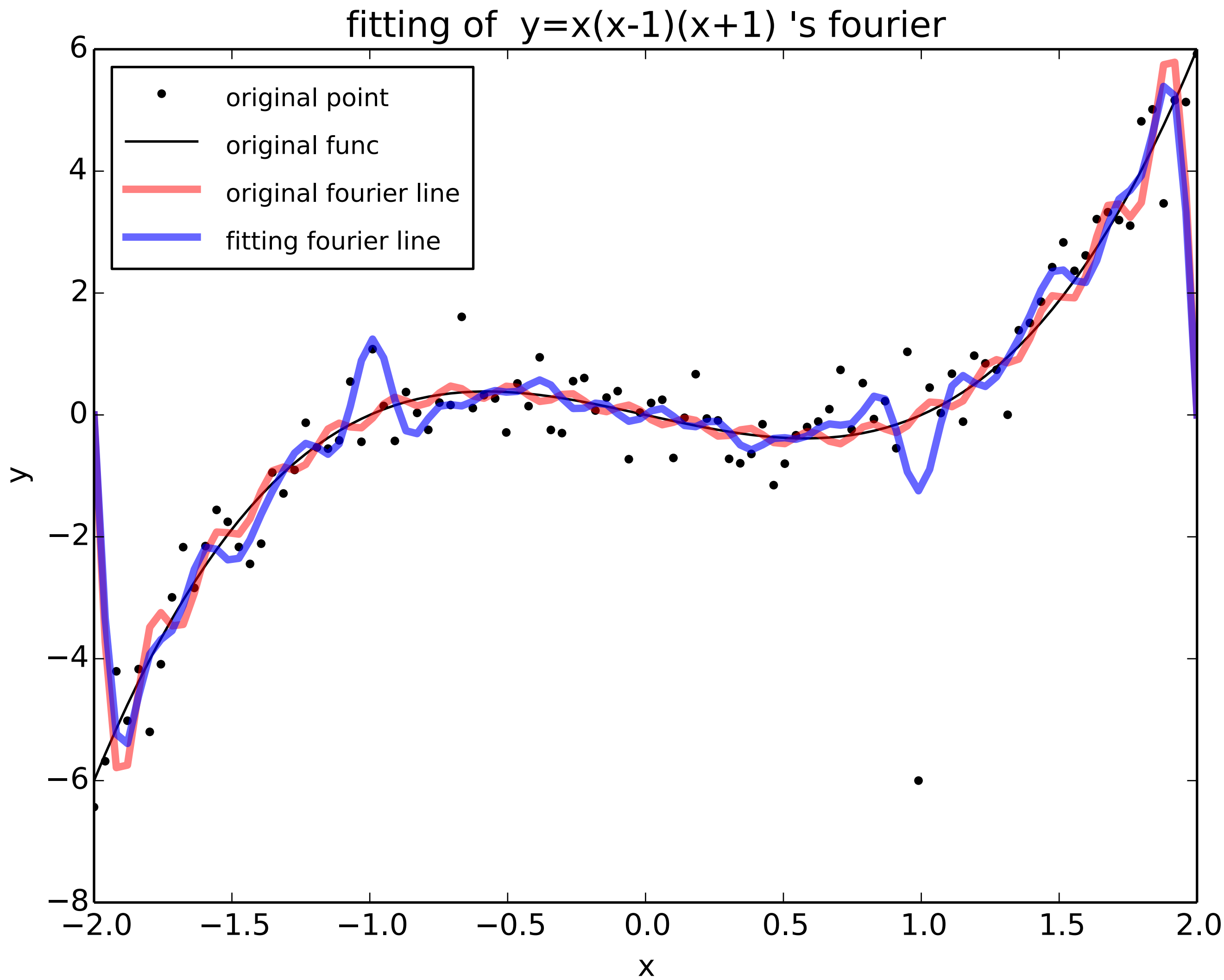
1.5 重构评价函数

为保证算法对 $f(x)$ 与 $f'(x)$ 都有较好拟合，并且两者的影响权重对结果影响均衡，构造评价函数

$$G'(a_0, a_1, ...a_{D-1}, \alpha) = (1-\alpha)MSE(f(a_0, a_1, ...a_{D-1}, x)) + \alpha MSE(f'(a_1, ...a_{D-1}, x)) = (1-\alpha)\frac{\sum_{i=0}^{N-1}(\frac{y_i - f(a_0, a_1, ...a_{D-1}, x_i)}{y_i}}{N}$$

$$s.t., \alpha \in [0, 1]$$

转化为求 $minG'(a_0, a_1, ...a_{D-1}, \alpha)$ 的解，其中取 0，这样就是求解 D+1 维最优解问题. 最终可求 α=0.000005213，也就是说第二张图的影响权重几乎为零，对应图像：



1.6 增加积分权重

由上可知 α 很小，也就是说导数部分对最终拟合的结果影响很小，引入积分权重进一步对上面函数进行拟合.

$$F(x)=\int_{-l}^{x}f(t)dt=-\frac{1}{2l}\int_{-l}^{l}xf(x)dx+\sum_{k=1}^{\infty}\frac{l}{k\pi}(a_k sin\frac{k\pi}{l}x-b_k cos\frac{k\pi}{l}x)+\frac{a_0(x+l)}{2}$$

可证：

$$\int_{-l}^{x}f(t)dt=\sum_{k=1}^{\infty}\frac{l}{k\pi}(a_k sin\frac{k\pi}{l}x-b_k(cos\frac{k\pi}{l}x+(-1)^{k+1})+\frac{a_0(x+l)}{2}$$

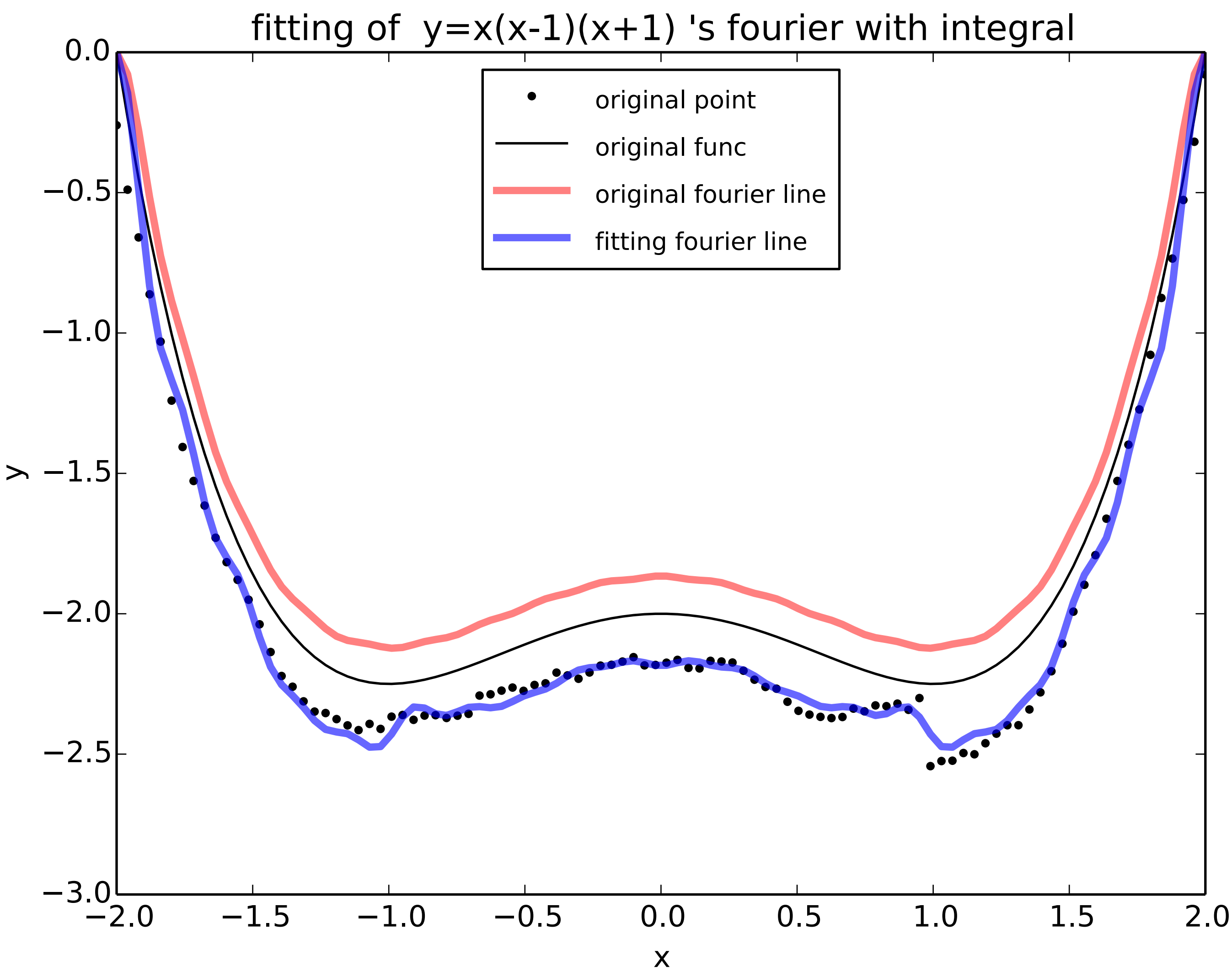
增加数据集

* S3={(x,F(x))|F(x)=integrate([f(x)]),x=linspace(-l,l,N)}

其中 linspace 为线性分布函数，integrate 对 x 点前面的所有点（包括自身）的积分增加评价权重：

$$MSE(F(a_0, a_1, ...a_{D-1}, x_0, ..., x_i)) = \frac{\sum_{i=0}^{N-1}(\frac{F_i - F(a_0, a_1, ...a_{D-1}, x_0, ..., x_i)}{y_i})^2}{N}$$

使用该 MSE 作为评价函数可得如下图:

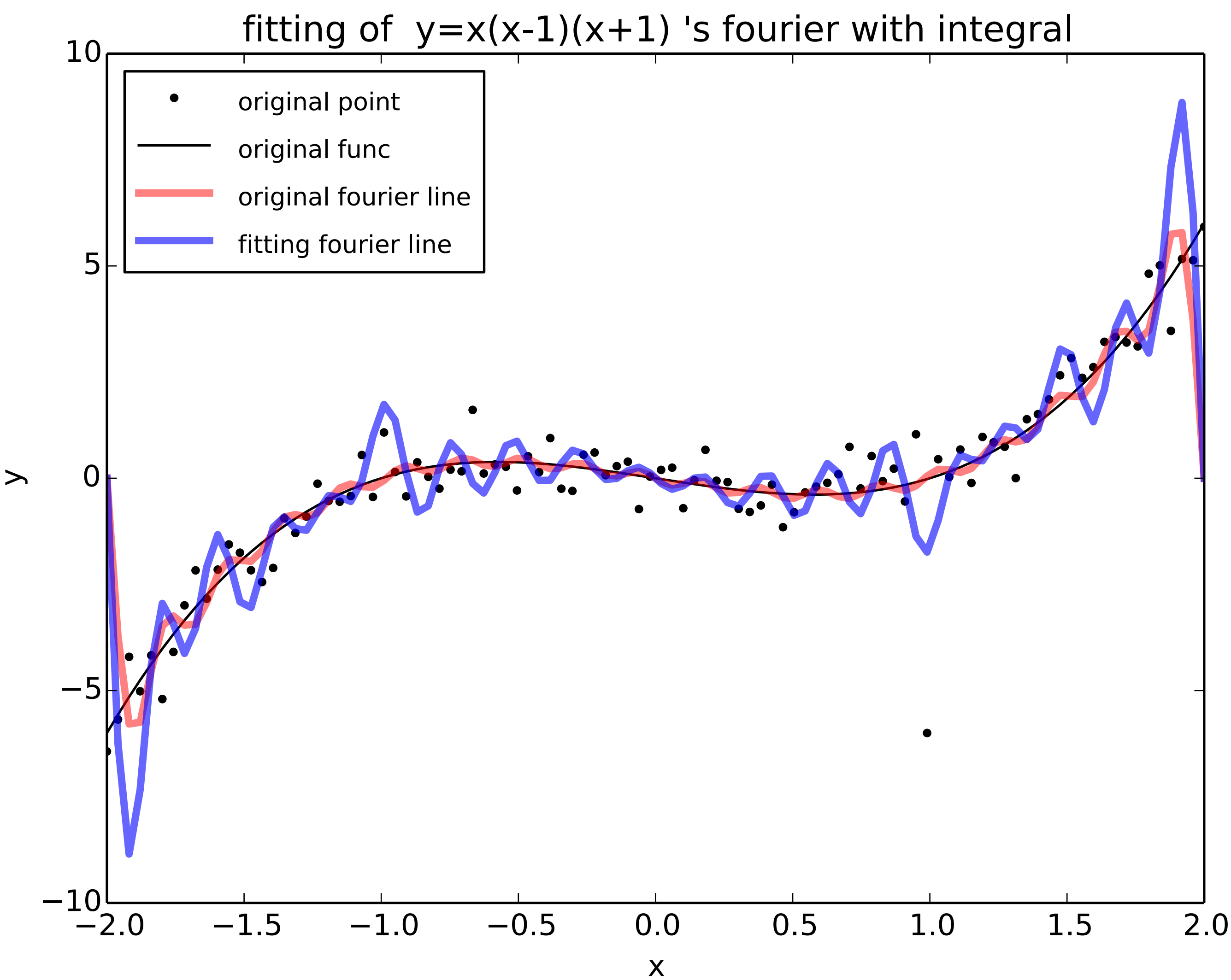


新评价函数：

$$G''(a_0, a_1, ...a_{D-1}, \alpha) = (1-\alpha-\beta)MSE(f(a_0, a_1, ...a_{D-1}, x)) + \alpha MSE(f'(a_1, ...a_{D-1}, x)) + \beta MSE(F(a_0, a_1, ...a_{D-1}, x_0, ..$$

$$s.t., \alpha, \beta, \alpha + \beta \in [0, 1]$$

通过计算可得 $\alpha = 5.3292029e{-}06, \beta = 0.998920117$, 也就是说这个函数增大积分权重的影响可以得到最好结果，如图所示：


fitting of y=x(x-1)(x+1) 's fourier with integral
original point
original func
original fourier line
fitting fourier line
10
5
0
−5
−10
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
x
y