傅立叶变换光谱测量技术实验报告

黄润华
Ocean University of China
email@noreply.com

杨超
Ocean University of China
email@somewhere.com

摘要—本实验报告为傅立叶变换光谱测量技术第四次实验报告，本报告采用 Python 仿真同时考虑有限扫描长度和采样间隔误差两因素影响下的傅里叶变换光谱测量系统的光谱测量曲线。实验仿真误差选取是随机、线性或正弦变化的误差，仿真采样间隔为 79.1nm。本仿真以 632.8nm 的 HeNe 激光和 532nm 的 YAG 激光为例。

Index Terms—Optical spectrum, python, fft

I. 实验目的

1. 仿真多种正弦形式的扫描长度误差与采样间隔误差同时叠加对傅里叶变换光谱测量曲线的影响;
2. 仿真扫描长度误差与采样间隔误差累积叠加对傅里叶变换光谱测量曲线的影响;

II. 实验原理

A. 分辨率和分辨能力

光谱仪的光谱分辨率是其定性区分彼此非常接近的两个光谱峰的能力的量度。通常，光谱分辨率由仪器线形状（ILS）的半峰全宽（FWHM）表示，光谱仪的输出光谱具有纯单色输入辐射。

光谱分辨率 R 由下式定义

$$R = \frac{\sigma_{max}}{\delta\sigma} \tag{1}$$

其中 σ_{max} 是光谱仪设计运行的最大波数。

1) 分辨率与最大 OPD 之间的关系: 根据公式

$$B_c(\sigma) = B_{reL}(\sigma) + iB_{ioL}(\sigma) = \int_{-\infty}^{\infty} I(x)\,\mathrm{rect}(x)e^{-i2\pi ax}dx \tag{2}$$

$$\begin{cases} cB(\sigma) = 2\,|\int_{-\infty}^{+\infty} I(x)\,\mathrm{rect}(x)e^{-2\pi ax}dx \\ \phi(\sigma) = \arctan\left[\frac{B_{ioL}(\sigma)}{B_{reL}(\sigma)}\right] \end{cases} \quad (\sigma \geq 0) \tag{3}$$

其中矩形函数定义为

$$\mathrm{rect}(x) = \begin{cases} 1 & |x| < L \\ 0 & |x| > L \end{cases}$$

因此，理论上 FTS 的光谱分辨率取决于可移动镜的扫描范围

$$B_e(\sigma) = \begin{cases} llB(\sigma)/2 & (\sigma \geq 0) \\ B(-\sigma)/2 & (\sigma \leq 0) \end{cases} \tag{4}$$

理论上，完美单色线的干涉图可以用来描述

$$I(x) = 2cos(2\pi\sigma_0 x) \tag{5}$$

将此表达式替换为方程 $B(\sigma)$、$\phi(\sigma)$, 考虑到实际上 $\sigma > 0$, 可以得到 ILS $B(\sigma)$ 的表达式:

$$B_{IIS}(\sigma) = 2L\frac{\sin\left[2\pi\left(\sigma_0+\sigma\right)L\right]}{2\pi\left(\sigma_0+\sigma\right)L} + \frac{\sin\left[2\pi\left(\sigma_0-\sigma\right)L\right]}{2\pi\left(\sigma_0-\sigma\right)L} \tag{6}$$

$$\approx 2L \times \frac{\sin\left[2\pi\left(\sigma_0-\sigma\right)L\right]}{2\pi\left(\sigma_0-\sigma\right)L} \tag{7}$$

$$= 2L\sin c\left[2\pi\left(\sigma_0-\sigma\right)L\right] \tag{8}$$

如图??所示，光谱从一条线 (δ 函数) 扩展到 *sinc* 函数形状 (仪器线形)。

$$(\delta\sigma)_{linewidth} = \frac{1.21}{2L} \tag{9}$$

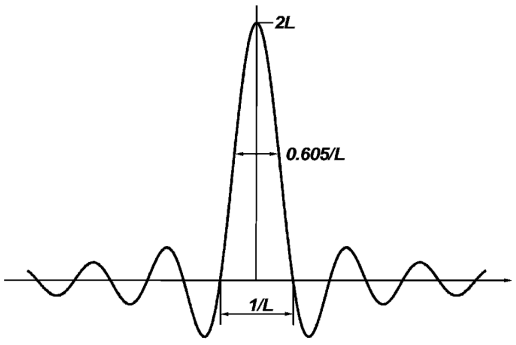

图 1. $B_{IIS}(\sigma)$

2) 分离共振: 分辨率可以通过分离图??所示光谱中相同强度 (或共振) 的两条单色线 (波数和) 来描述。如果两个线峰值之间的下降幅度大于线峰值的 20%，则可以声称解决了两个共振，此时

$$(\delta\sigma)_{separation} = \frac{1.46}{2L} \tag{10}$$

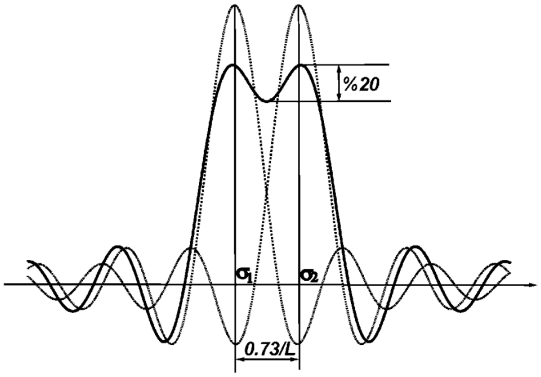

图 2. 分离共振解析结果

3) 瑞利准则: 瑞利准则分离两个 ILS 的峰值，使得一个共振的最大值落在另一个共振的零点处，因此，仪器线形状决定的分辨率是

$$\delta\sigma = \frac{1}{2L} \tag{11}$$

B. 分辨率与发散角度之间的关系

图??是迈克尔逊干涉仪的等效图，根据图??与图??可以计算出 OPD 与夹角 θ 之间的关系

$$OPD = 2 \times \frac{x/2}{\cos\theta} - x\tan\theta\sin\theta = x\cos\theta \tag{12}$$

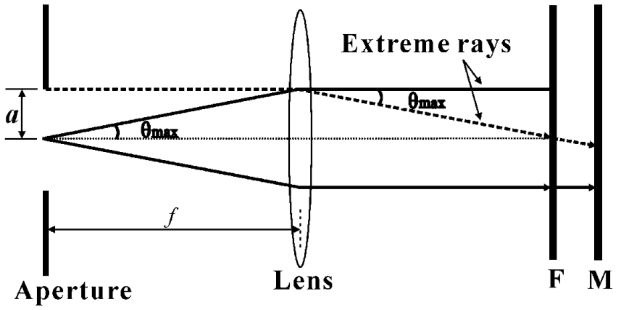

图 3. 迈克尔逊干涉仪的等效图

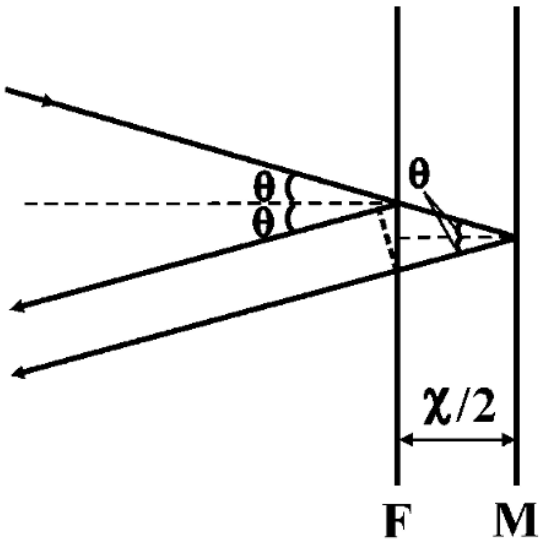

图 4. 迈克尔逊干涉仪接收端放大图

我们可以得到扩展源的归一化干涉图，对着一个立体角 Ω_{max} 是

$$I(x,\Omega_{\max}) = \frac{1}{\Omega_{\max}}\int_0^{\infty} B(\sigma)\int_0^{\Omega_{\text{mix}}} \cos(2\pi\sigma x\cos\theta)d\Omega d\sigma$$

根据立体角的公式，上述式子可以化简为

$$I(x,\Omega_{\max}) = \int_0^{\infty} B(\sigma)sinc\frac{\Omega_{\max}\sigma x}{2}\cos[2\pi\sigma x - \frac{\Omega_{\max}\sigma x}{2}]d\sigma$$

对于单色光源其满足

$$B(\sigma) = \delta(\sigma - \sigma_0)$$

其仪器线轮廓可以写为

$$\begin{cases} B_{Div-1LS}(\sigma) = [\pi/(\sigma_0\Omega_{\max})]\,\text{rect}(\sigma_1,\sigma_2) \\ \qquad\qquad +[\pi/(\sigma_0\Omega_{\max})]\,\text{rect}(-\sigma_2,-\sigma_1) \\ \sigma_2 = \sigma_0 - \sigma_0\Omega_{\max}/2\pi \\ \sigma_1 = \sigma_0 \end{cases}$$

于是可以得到

$$B_{Div-1LS}(\sigma) = [\pi/(\sigma_0\Omega_{\max})]\,\text{rect}(\sigma_1,\sigma_2) \tag{13}$$

$$\bar{\sigma} = \sigma_0[1 - (\Omega_{\max}/4\pi)] \tag{14}$$

总的波数差或分辨率是

$$\delta\sigma = \frac{\sigma_0 \Omega_{max}}{2\pi}$$

考虑到可移动镜的扫描长度有限，实际 FTS 的 ILS 应该是

$$B_{ILS}(\sigma) = B_{Div-ILS}(\sigma) * 2L \sin c[2\pi\sigma L] \tag{15}$$

其中 L 是可移动镜的最大位移，$*$ 是卷积算子。图??中两个极端射线的 OPD 之间的差异公式如下:

$$\Delta OPD = 2L - 2L\cos\theta_{\max} \approx L\theta_{\max}^2 = L\frac{a^2}{f^2}$$

其中 a 是入口孔径的半径，f 是准直器的焦距。当 ΔOPD 等于 $\lambda/2$ 时，两条光线异相，它们之间会发生破坏性叠加。对于宽带辐射输入，存在的最短波长决定了 FTS 的 ΔOPD 的最大有效值，如下式所示:

$$\Delta OPD \leq \frac{\lambda_{min}}{2} = \frac{1}{2\sigma_{max}} \tag{16}$$

我们可以得到 a 的值应满足以下不等式，以获得大于 R 的分辨率:

$$a \leq \frac{f}{\sqrt{R}} \tag{17}$$

C. 傅里叶变换光谱仪的优点

与色散光谱仪相比，FTS 具有许多广泛宣传的优点。然而，只有吞吐量（Jacquinot）和多路复用（Felgett）优势是 FTS 操作原理所固有的，而不是特定的工程设计。

1) 吞吐量或 Jacquinot 优势: 吞吐量优势是 FTS 可以具有大的圆形入口孔，其面积比分散光谱仪中的狭缝的面积大得多，以获得相同的分辨率。吞吐量定义为，其中是限制孔径的面积，并且是图??中所示的准直或聚焦光学器件所对应的立体角。

FTS 的最大吞吐量由下式给出

$$T_{FTS} = A\Omega_s = \frac{\pi}{R}A_{mirror} \tag{18}$$

其中 A_{mirror} 是镜子的投影面积。

对于光栅光谱仪，实现分辨率 R 的能量吞吐量受其狭缝区域和准直光学系统的限制，并由下式给出:

$$T_{grating} = \frac{l}{f_g R}A_{grating} \tag{19}$$

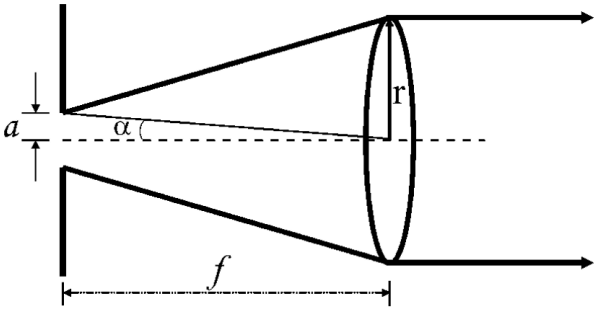

图 5. 孔径和立体角的关系图

其中 f_g 是准直光学系统的焦距，l 是狭缝的高度，$A_{grating}$ 是光栅的投影面积。对于光栅光谱仪，$\frac{l}{f_g R}$ 小于 1/20。

因此，对于相同的分辨率和类似的仪器尺寸，FTS 比光栅光谱仪具有高 60 倍的能量收集能力。因此，Jacquinot 的优势使得 FTS 更适合于弱信号测量，其中检测器噪声占主导地位，频谱信噪比（SNR）与吞吐量成比例地增加。

2) Multiplex 或 Felgett 的优势: 多路复用的优点是 FTS 在扫描周期内同时观察来自给定频谱的整个范围的所有频谱信息。因此，通过使用 FTS 和使用单色仪可获得的 SNR 的比率:

$$\frac{SNR_{FTS}^S}{SNR_{monochromator}} = N^{\frac{1}{2}} \tag{20}$$

FTS 的多重优势仅存在于红外和远红外信号的测量中，并且在可见 - 紫外信号的检测中丢失，因为在红外检测中，与信号电平无关的检测器噪声占主导地位。量子噪声在可见光 - 紫外信号检测中占主导地位。

3) Connes 的优势: FTS 的波数范围来自 He-Ne 激光条纹，其作为每次扫描中采样位置的内部参考。这种激光的波数非常准确，并且非常稳定。因此，干涉仪的波数校准更精确，并且具有比分散仪器的校准好得多的长期稳定性.

III. 实验内容

仿真同时考虑有限扫描长度和采样间隔误差两因素影响下的傅里叶变换光谱测量系统的光谱测量曲线。误差可以是随机、线性或正选变化的。以 632.8nm 的 He-Ne 激光和 532nm 的 YAG 激光为例。（采样间隔为 79.1nm）

IV. 实验结果

本实验报告以 632.8nm 的波长为例。

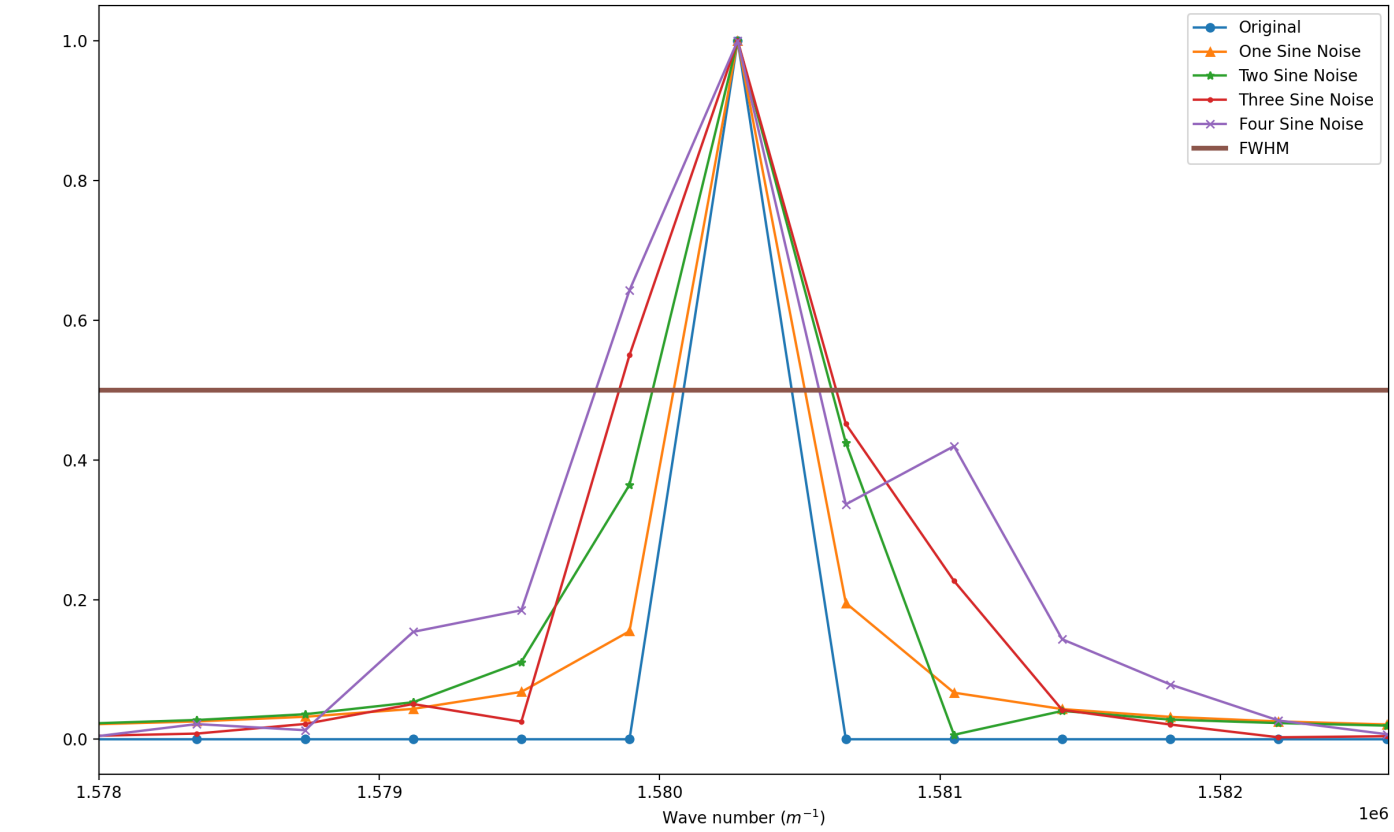

图 6. 原始信号的光谱曲线与叠加多种不同频率正弦噪声后光谱曲线的比较

A. 不同频率正弦波噪声的叠加结果

图片??、图??、图??与图??显示了本次仿真的不同频率正弦波噪声，其中图??为单一频率正弦波噪声叠加后的光谱曲线，图??为两个不同频率正弦波噪声叠加后的光谱曲线，图??为三个不同频率正弦波噪声叠加后的光谱曲线，图??为四个不同频率正弦波噪声叠加后的光谱曲线。从光谱曲线上可以看到，随着叠加的正弦噪声数的增加，光谱曲线的半峰全宽逐渐增大。

B. 噪声累积叠加结果

图片??、图??、图??与图??显示了四种不同幅度值的随机噪声的累积叠加结果。每张图片第一行代表理论扫描距离与叠加多次噪声后的扫描距离的比较，第二行左半部分代表代表未添加累积噪声的光谱曲线图，右半部分代表添加累积随机噪声后的光谱曲线图。

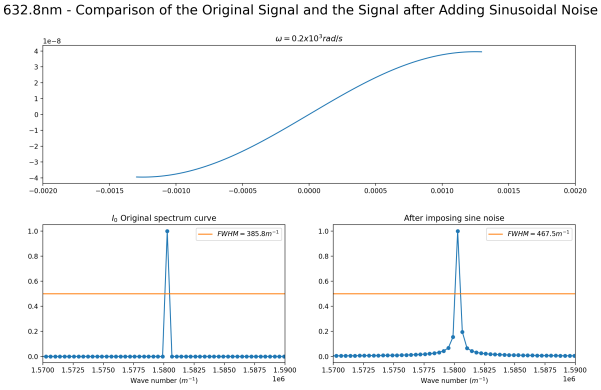

图 7. 单一频率正弦波噪声叠加后的光谱曲线

图片??、图??、图??与图??的结果表明，累积的随机误差会对光谱曲线的分辨率产生巨大的影响，一个极小的随机误差在不断迭代多次后会产生质的飞跃，具体

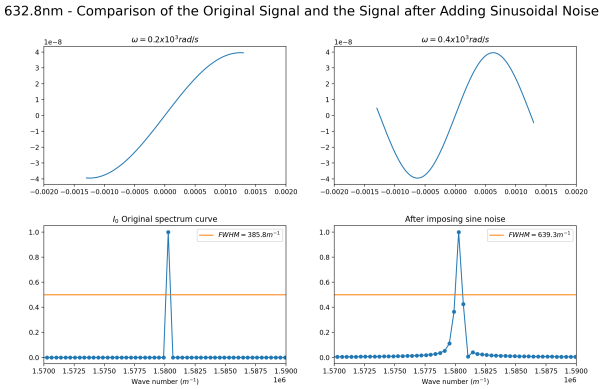

图 8. 两个不同频率正弦波噪声叠加后的光谱曲线

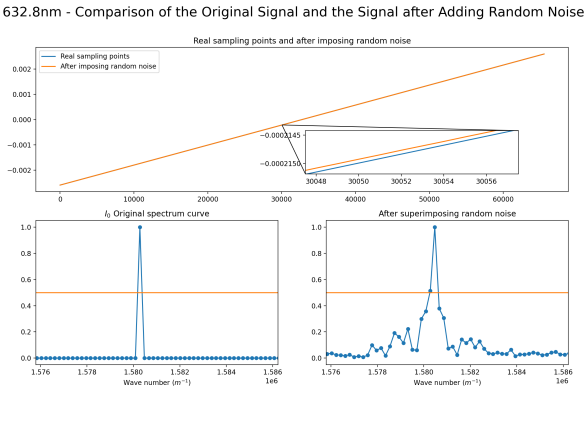

图 11. 随机噪声叠加结果 1(图片第一行代表理论扫描距离与叠加多次噪声后的扫描距离的比较，第二行左半部分代表代表未添加累积噪声的光谱曲线图，右半部分代表添加累积随机噪声后的光谱曲线图。)

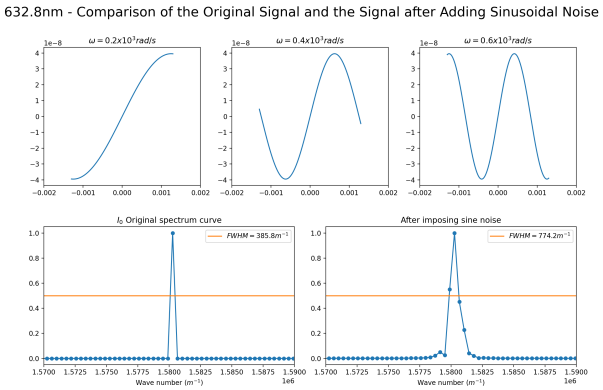

图 9. 三个不同频率正弦波噪声叠加后的光谱曲线

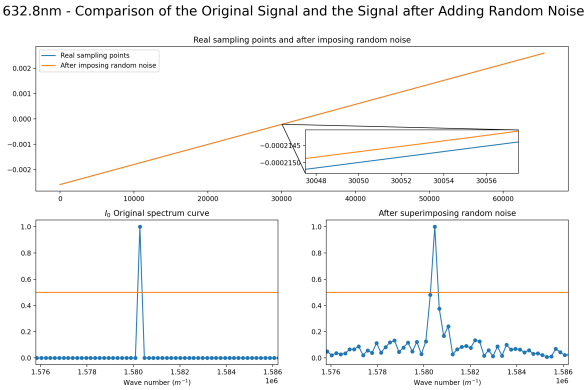

图 12. 随机噪声叠加结果 2(图片第一行代表理论扫描距离与叠加多次噪声后的扫描距离的比较，第二行左半部分代表代表未添加累积噪声的光谱曲线图，右半部分代表添加累积随机噪声后的光谱曲线图。)

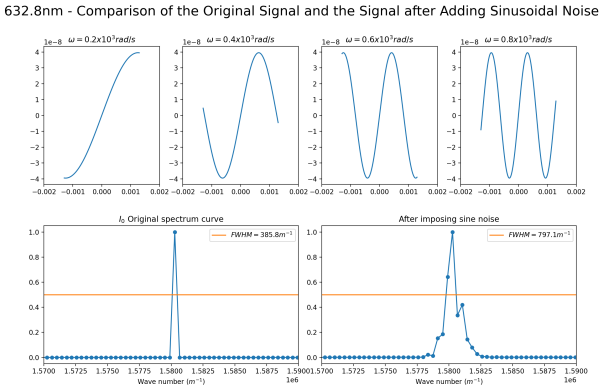

图 10. 四个不同频率正弦波噪声叠加后的光谱曲线

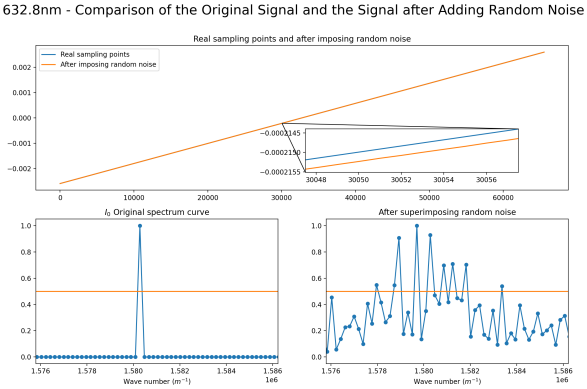

图 13. 线性噪声叠加结果 3(图片第一行代表理论扫描距离与叠加多次噪声后的扫描距离的比较，第二行左半部分代表代表未添加累积噪声的光谱曲线图，右半部分代表添加累积随机噪声后的光谱曲线图。)

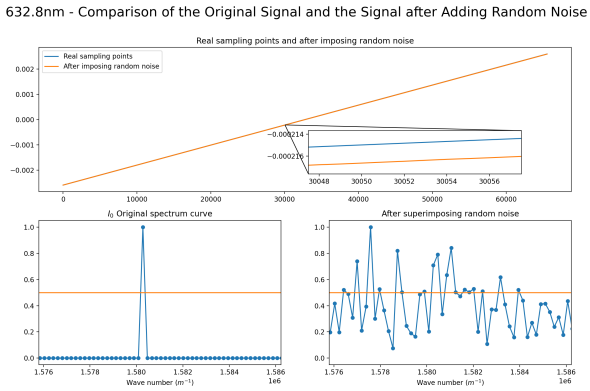

图 14. 线性噪声叠加结果 4(图片第一行代表理论扫描距离与叠加多次噪声后的扫描距离的比较，第二行左半部分代表代表未添加累积噪声的光谱曲线图，右半部分代表添加累积随机噪声后的光谱曲线图。)

表现在加噪声后的光谱曲线图的半峰全宽相比于原始光谱曲线图的半峰全宽加宽了较为大的范围。同时影响随着随机噪声幅度的增大而逐渐剧烈。

V. 结语

本实验完成了利用 Python 仿真同时考虑有限扫描长度和采样间隔误差两因素影响下的傅里叶变换光谱测量系统的光谱测量曲线。

附录 A
多个频率的正弦噪声叠加代码

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.fftpack import fft
from pylab import *
import matplotlib.gridspec as gridspec
# 本程序模拟有限扫描长度误差误差影响下的傅里叶变换光谱测量系统的光谱测量曲线
# 程序具体参数如下:
#   - 采样间隔选取 79.1nm
#   - 干涉图波长选取 632.8nm
#   - 干涉图采样点数选取 2^12 (该点数下仿真图像最佳)
#   - 本实验只叠加一种噪声——正弦噪声
#       - 频率分别为0.6*10^4, 1*10^4, 2*10^4与4*10^4 rad/s
#   - 实验目的在于模拟不同周期的正弦噪声同时叠加后对信号的影响
#   - 分别叠加2种、3种与4种噪声
def GetFFT(I0, Iw0, n0):
    Y0 = 2*abs(fft(I0,n0))
    Y0_max = max(Y0);
    Y0 = Y0/Y0_max
    Y0 = Y0[:int(n0/2)]

    Yw0 = 2*abs(fft(Iw0,n0))
    Yw0_max = max(Yw0);
    Yw0 = Yw0/Yw0_max
    Yw0 = Yw0[:int(n0/2)]
    return Y0, Yw0

# 波长为632.8nm
laimda0 = 632.8*10**(-9)
# 79.1nm的采样间隔
i = 79.1*10**(-9)
# 中心点的采样频
sigma0 = 1/laimda0
p1 = (-1)*(2**11)*79.1*10**(-9)
# 2**n个点
p2 = (2**11-1)*79.1*10**(-9)
# 无补零
x0 = np.arange(p1, p2, i)
n0 = n1 = 2**(int(np.log2(len(x0)))+1)
print("n0 length = %d" %n0)

# 此部分为叠加正弦噪声
noise_sin1 = laimda0/16*np.sin(2*np.pi
    *(0.6*10**4)*x0)
noise_sin2 = laimda0/16*np.sin(2*np.pi
    *(1*10**4)*x0)
noise_sin3 = laimda0/16*np.sin(2*np.pi
    *(2.5*10**4)*x0)
noise_sin4 = laimda0/16*np.sin(2*np.pi
    *(4*10**4)*x0)

I0 = np.cos(2*np.pi*sigma0*x0)
```

```
I_sin_all_1 = np.cos(2*np.pi*sigma0*(x0
    + noise_sin1))
I_sin_all_2 = np.cos(2*np.pi*sigma0*(x0
    + noise_sin1 + noise_sin2))
I_sin_all_3 = np.cos(2*np.pi*sigma0*(x0
    + noise_sin1 + noise_sin2 +
   noise_sin3))
I_sin_all_4 = np.cos(2*np.pi*sigma0*(x0
    + noise_sin1 + noise_sin2 +
   noise_sin3 + noise_sin4))

Y0, Y_sine_all_1 = GetFFT(I0,
   I_sin_all_1, n0)
Y0, Y_sine_all_2 = GetFFT(I0,
   I_sin_all_2, n0)
Y0, Y_sine_all_3 = GetFFT(I0,
   I_sin_all_3, n0)
Y0, Y_sine_all_4 = GetFFT(I0,
   I_sin_all_4, n0)

# 设置频谱图的横坐标
fs_0 = 1/i*np.arange(n0/2)/n0

best_Y0_average_range = 0.5*np.ones((Y0
   .size, 1))

# 只叠加一种波
figure(1)
gs = gridspec.GridSpec(2, 4)
gs.update(wspace=0.5, hspace=0.3)

subplot(gs[0, :4])
sin1, = plt.plot(x0, noise_sin1)
plt.title("$\omega=0.6x10^4 rad/s$")
plt.xlim(-0.00006, 0.00006)

subplot(gs[1, 0:2])
plt.plot(fs_0, Y0, marker='o', ms=5)
FWHM_original, = plt.plot(fs_0,
   best_Y0_average_range)
plt.title("$I_0$ Original spectrum
   curve")
plt.legend(handles=[FWHM_original],
   labels=['$FWHM = 3086.48 m^{-1}$'],
   loc='upper right')
plt.xlim(1.50*(10**6), 1.67*(10**6))
plt.xlabel('Wave number ($m^{-1}$)')

subplot(gs[1, 2:4])
plt.plot(fs_0, Y_sine_all_1, marker='o'
   , ms=5)
FWHM_sine_1, = plt.plot(fs_0,
   best_Y0_average_range)
plt.title("After imposing sine noise")
plt.legend(handles=[FWHM_sine_1],labels
   =['$FWHM = 3118.46 m^{-1}$'], loc='
   upper right')
plt.xlim(1.50*(10**6), 1.67*(10**6))
plt.xlabel('Wave number ($m^{-1}$)')

plt.suptitle("632.8nm - Comparison of
   the Original Signal and the Signal
   after Adding Sinusoidal Noise",
   fontsize = 20)

# 同时叠加两种波
figure(2)
gs = gridspec.GridSpec(2, 4)
gs.update(wspace=0.5, hspace=0.3)

subplot(gs[0, :2])
sin1, = plt.plot(x0, noise_sin1)
plt.title("$\omega=0.6x10^4 rad/s$")
plt.xlim(-0.00006, 0.00006)

subplot(gs[0, 2:4])
sin2, = plt.plot(x0, noise_sin2)
plt.title("$\omega=1x10^4 rad/s$")
plt.xlim(-0.00006, 0.00006)

subplot(gs[1, 0:2])
```

```
plt.plot(fs_0, Y0, marker='o', ms=5)
FWHM_original, = plt.plot(fs_0,
    best_Y0_average_range)
plt.title("$I_0$ Original spectrum
    curve")
plt.legend(handles=[FWHM_original],
    labels=['$FWHM = 3086.48 m^{-1}$'],
    loc='upper right')
plt.xlim(1.50*(10**6), 1.67*(10**6))
plt.xlabel('Wave number ($m^{-1}$)')

subplot(gs[1, 2:4])
plt.plot(fs_0, Y_sine_all_2, marker='o'
    , ms=5)
FWHM_sine_2, = plt.plot(fs_0,
    best_Y0_average_range)
plt.title("After imposing sine noise")
plt.legend(handles=[FWHM_sine_2],labels
    =['$FWHM = 3206.04 m^{-1}$'], loc='
    upper right')
plt.xlim(1.50*(10**6), 1.67*(10**6))
plt.xlabel('Wave number ($m^{-1}$)')

plt.suptitle("632.8nm - Comparison of
    the Original Signal and the Signal
    after Adding Sinusoidal Noise",
    fontsize = 20)

# 同时叠加三种波
figure(3)
gs = gridspec.GridSpec(2, 6)
gs.update(wspace=0.5, hspace=0.3)

subplot(gs[0, :2])
sin1, = plt.plot(x0, noise_sin1)
plt.title("$\omega=0.6x10^4 rad/s$")
plt.xlim(-0.00006, 0.00006)

subplot(gs[0, 2:4])
sin2, = plt.plot(x0, noise_sin2)
plt.title("$\omega=1x10^4 rad/s$")
plt.xlim(-0.00006, 0.00006)

subplot(gs[0, 4:6])
sin3, = plt.plot(x0, noise_sin3)
plt.title("$\omega=2.5x10^4 rad/s$")
plt.xlim(-0.00006, 0.00006)

subplot(gs[1, 0:3])
plt.plot(fs_0, Y0, marker='o', ms=5)
FWHM_original, = plt.plot(fs_0,
    best_Y0_average_range)
plt.title("$I_0$ Original spectrum
    curve")
plt.legend(handles=[FWHM_original],
    labels=['$FWHM = 3086.48 m^{-1}$'],
    loc='upper right')
plt.xlim(1.50*(10**6), 1.67*(10**6))
plt.xlabel('Wave number ($m^{-1}$)')

subplot(gs[1, 3:6])
plt.plot(fs_0, Y_sine_all_3, marker='o'
    , ms=5)
FWHM_sine_3, = plt.plot(fs_0,
    best_Y0_average_range)
plt.title("After imposing sine noise")
plt.legend(handles=[FWHM_sine_3],labels
    =['$FWHM = 3190.52 m^{-1}$'], loc='
    upper right')
plt.xlim(1.50*(10**6), 1.67*(10**6))
plt.xlabel('Wave number ($m^{-1}$)')

plt.suptitle("632.8nm - Comparison of
    the Original Signal and the Signal
    after Adding Sinusoidal Noise",
    fontsize = 20)

# 同时叠加四种波
figure(4)
gs = gridspec.GridSpec(2, 8)
gs.update(wspace=0.5, hspace=0.3)
```

```
subplot(gs[0, :2])
sin1, = plt.plot(x0, noise_sin1)
plt.title("$\omega=0.6x10^4 rad/s$")
plt.xlim(-0.00006, 0.00006)

subplot(gs[0, 2:4])
sin2, = plt.plot(x0, noise_sin2)
plt.title("$\omega=1x10^4 rad/s$")
plt.xlim(-0.00006, 0.00006)

subplot(gs[0, 4:6])
sin3, = plt.plot(x0, noise_sin3)
plt.title("$\omega=2.5x10^4 rad/s$")
plt.xlim(-0.00006, 0.00006)

subplot(gs[0, 6:8])
sin4, = plt.plot(x0, noise_sin4)
plt.title("$\omega=4x10^4 rad/s$")
plt.xlim(-0.00006, 0.00006)

subplot(gs[1, 0:4])
plt.plot(fs_0, Y0, marker='o', ms=5)
FWHM_original, = plt.plot(fs_0,
    best_Y0_average_range)
plt.title("$I_0$ Original spectrum
    curve")
plt.legend(handles=[FWHM_original],
    labels=['$FWHM = 3086.48 m^{-1}$'],
    loc='upper right')
plt.xlim(1.50*(10**6), 1.67*(10**6))
plt.xlabel('Wave number ($m^{-1}$)')

subplot(gs[1, 4:8])
plt.plot(fs_0, Y_sine_all_4, marker='o'
    , ms=5)
FWHM_sine_4, = plt.plot(fs_0,
    best_Y0_average_range)
plt.title("After imposing sine noise")
plt.legend(handles=[FWHM_sine_4],labels
    =['$FWHM = 3198.12 m^{-1}$'], loc='
    upper right')
plt.xlim(1.50*(10**6), 1.67*(10**6))
plt.xlabel('Wave number ($m^{-1}$)')

plt.suptitle("632.8nm - Comparison of
    the Original Signal and the Signal
    after Adding Sinusoidal Noise",
    fontsize = 20)

# 将所有波形放在一起
figure(5)
pic, = plt.plot(fs_0, Y0, marker='o',
    ms=5)
pic1, = plt.plot(fs_0, Y_sine_all_1,
    marker='^', ms=5)
pic2, = plt.plot(fs_0, Y_sine_all_2,
    marker='*', ms=5)
pic3, = plt.plot(fs_0, Y_sine_all_3,
    marker='.', ms=5)
pic4, = plt.plot(fs_0, Y_sine_all_4,
    marker='x', ms=5)
FWHM_sine_5, = plt.plot(fs_0,
    best_Y0_average_range, linewidth=3)
plt.legend(handles=[pic, pic1, pic2,
    pic3, pic4, FWHM_sine_5],labels=['
    Original', 'One Sine Noise', 'Two
    Sine Noise', 'Three Sine Noise', '
    Four Sine Noise', 'FWHM'], loc='
    upper right')
plt.xlim(1.50*(10**6), 1.67*(10**6))
plt.xlabel('Wave number ($m^{-1}$)')
plt.suptitle("632.8nm - Comparison of
    the Original Signal and the Signal
    after Adding Sinusoidal Noise",
    fontsize = 20)

plt.show()
```

附录 B
累加随机噪声代码

```
import numpy as np
import matplotlib.pyplot as plt
```

```
from scipy.fftpack import fft
from pylab import *
import matplotlib.gridspec as gridspec
from scipy import signal
import random
from mpl_toolkits.axes_grid1.
    inset_locator import mark_inset
from mpl_toolkits.axes_grid1.
    inset_locator import inset_axes

# 本程序模拟有限扫描长度误差误差影响下
    的傅里叶变换光谱测量系统的光谱测量曲
    线
# 程序具体参数如下:
#   - 采样间隔选取 79.1nm
#   - 干涉图波长选取 632.8nm
#   - 干涉图采样点数选取 2^12 (该点数下
    仿真图像最佳)
#   - 本实验只叠加一种噪声——随机噪声 1
#   - 实验目的在于模拟累积的随机噪声叠
    加后对信号的影响, 同时在此基础上观察
    扫描长度增加对信号点影响

def GetFFT(I0, Iw0, n0):
    Y0 = 2*abs(fft(I0,n0))
    Y0_max = max(Y0);
    Y0 = Y0/Y0_max
    Y0 = Y0[:int(n0/2)]

    Yw0 = 2*abs(fft(Iw0,n0))
    Yw0_max = max(Yw0);
    Yw0 = Yw0/Yw0_max
    Yw0 = Yw0[:int(n0/2)]
    return Y0, Yw0



# 波长为632.8nm
laimda0 = 632.8*10**(-9)
# 79.1nm的采样间隔
i = 79.1*10**(-9)
# 中心点的采样频
sigma0 = 1/laimda0
p1 = (-1)*(2**15)*79.1*10**(-9)
# 2**n个点
p2 = (2**15-1)*79.1*10**(-9)
# 无补零
x0 = np.arange(p1, p2, i)
x0_error = np.arange(p1, p2, i)

n0 = n1 = 2**(int(np.log2(len(x0)))+1)
# n0 = len(x0)
print("n0 length = %d" %n0)

#  此部分为叠加线性噪声
I0 = np.cos(2*np.pi*sigma0*x0)

print(x0)

for j in range(1, len(x0)):
    x0_error[j] = x0_error[j-1] +
        random.uniform(-laimda0/280,
        laimda0/280) + i

print("
    ----------------------------------------
    ")

print(x0_error)

Iw2 = np.cos(2*np.pi*sigma0*(x0_error))

Y0, Yw2 = GetFFT(I0, Iw2, n0)

# 设置频谱图的横坐标
fs_2 = 1/i*np.arange(n0/2)/n0
p_2 = np.arange(len(x0))
best_Y0_average_range = 0.5*np.ones((
    Yw2.size, 1))

figure(1)
ax = plt.subplot(2,1,1)
axins_plot1, = plt.plot(p_2, x0)
```

```
axins_plot2, = plt.plot(p_2, x0_error)
plt.title("Real sampling points and
    after imposing random noise")

plt.legend(handles=[axins_plot1,
    axins_plot2],labels=['Real sampling
    points', 'After imposing random
    noise'], loc='upper left')

# 嵌入绘制局部放大图的坐标系
axins = inset_axes(ax, width="40%",
    height="30%",loc='lower left',
                   bbox_to_anchor=(0.5,
                       0.1, 1, 1),
                   bbox_transform=ax.
                       transAxes)

# 在子坐标系中绘制原始数据
axins.plot(p_2, x0)
axins.plot(p_2, x0_error)



# 设置放大区间
zone_left = 30050
zone_right = 30055

# 坐标轴的扩展比例（根据实际数据调整）
x_ratio = 0.5 # x轴显示范围的扩展比例
y_ratio = 0.5 # y轴显示范围的扩展比例

# X轴的显示范围
xlim0 = p_2[zone_left]-(p_2[zone_right
    ]-p_2[zone_left])*x_ratio
xlim1 = p_2[zone_right]+(p_2[zone_right
    ]-p_2[zone_left])*x_ratio

# Y轴的显示范围
y = np.hstack((x0[zone_left:zone_right
    ], x0_error[zone_left:zone_right]))
ylim0 = np.min(y)-(np.max(y)-np.min(y))
    *y_ratio
ylim1 = np.max(y)+(np.max(y)-np.min(y))
    *y_ratio

# 调整子坐标系的显示范围
axins.set_xlim(xlim0, xlim1)
axins.set_ylim(ylim0, ylim1)

# 建立父坐标系与子坐标系的连接线
# loc1 loc2: 坐标系的四个角
# 1（右上）2（左上）3(左下) 4(右下)
mark_inset(ax, axins, loc1=3, loc2=1,
    fc="none", ec='k', lw=1)



subplot(2,2,3)
plt.plot(fs_2, Y0, marker='o', ms=5)
plt.plot(fs_2,best_Y0_average_range)
plt.title("$I_0$ Original spectrum
    curve")
plt.xlim(1.5758*(10**6),
    1.58622*(10**6))
plt.xlabel('Wave number ($m^{-1}$)')

subplot(2,2,4)
plt.plot(fs_2, Yw2, marker='o', ms=5)
plt.plot(fs_2,best_Y0_average_range)
plt.title("After superimposing random
    noise")
plt.xlim(1.5758*(10**6),
    1.58622*(10**6))
plt.xlabel('Wave number ($m^{-1}$)')


plt.suptitle("632.8nm - Comparison of
    the Original Signal and the Signal
    after Adding Random Noise", fontsize
     = 20)

plt.show()
```