## Report of Bandlimited signal recovery from zero-crossings

吴天宇 12334125

## 0.1题目

**Bandlimited signal recovery from zero-crossings.** Let  $y \in \mathbb{R}^n$  denote a bandlimited signal, which means that it can be expressed as a linear combination of sinusoids with frequencies in a band:

$$y_t = \sum_{j=1}^B a_j \cos \left(rac{2\pi}{n}(f_{\min}+j-1)t
ight) + b_j \sin \left(rac{2\pi}{n}(f_{\min}+j-1)t
ight), \quad t=1,\dots,n,$$

where  $f_{\min}$  is the lowest frequency in the band, B is the bandwidth, and  $a,b\in\mathbb{R}^B$  are the cosine and sine coefficients, respectively. We are given  $f_{\min}$  and B, but not the coefficients a,b or the signal y.

We do not know y, but we are given its sign s = sign(y), where  $s_t = 1$  if  $y_t \ge 0$  and  $s_t = -1$  if  $y_t < 0$ . (Up to a change of overall sign, this is the same as knowing the 'zero-crossings' of the signal, i.e., when it changes sign. Hence the name of this problem.)

We seek an estimate  $\hat{y}$  of y that is consistent with the bandlimited assumption and the given signs. Of course we cannot distinguish y and  $\alpha y$ , where  $\alpha>0$ , since both of these signals have the same sign pattern. Thus, we can only estimate y up to a positive scale factor. To normalize  $\hat{y}$ , we will require that  $||\hat{y}||_1=n$ , i.e., the average value of  $|y_i|$  is one. Among all  $\hat{y}$  that are consistent with the bandlimited assumption, the given signs, and the normalization, we choose the one that minimizes  $||\hat{y}||_2$ .

(a) Show how to find  $\hat{y}$  using convex or quasiconvex optimization.

(b) Apply your method to the problem instance with data in zero\_crossings\_data\*. The data files also include the true signal y (which of course you cannot use to find  $\hat{y}$ ). Plot  $\hat{y}$  and y, and report the relative recovery error,  $||\hat{y} - y||_2 / ||y||_2$ . Give one short sentence commenting on the quality of the recovery.

## 0.2题目翻译

**从零交叉恢复带限信号。** 设  $y\in\mathbb{R}^n$  表示一个带限信号,这意味着它可以表示为一个频带内正弦波的线性组合:

$$y_t = \sum_{j=1}^B a_j \cos \left(rac{2\pi}{n}(f_{\min}+j-1)t
ight) + b_j \sin \left(rac{2\pi}{n}(f_{\min}+j-1)t
ight), \quad t=1,\dots,n,$$

其中  $f_{\min}$  是频带中的最低频率,B 是带宽, $a,b\in\mathbb{R}^B$  分别是余弦和正弦系数。我们被给定了  $f_{\min}$  和 B,但没有给出系数 a,b 或信号 y。

我们不知道 y,但我们得到了它的符号  $s=\mathrm{sign}(y)$ ,其中  $s_t=1$  如果  $y_t\geq 0$  并且  $s_t=-1$  如果  $y_t<0$ 。(直到整体符号的改变,这和知道信号的'零交叉'是一样的,即,当它改变符号时。因此这个问题的名称。)

我们寻求一个与带限假设和给定符号一致的 y 的估计  $\hat{y}$ 。当然,我们不能区分 y 和  $\alpha y$ ,其中  $\alpha>0$ ,因为这两个信号有相同的符号模式。因此,我们只能估计 y 直到一个正的比例因子。为了规范化  $\hat{y}$ ,我们将要求  $\|\hat{y}\|_1=n$ ,即  $|y_i|$  的平均值是一。在所有与带限假设、给定符号和规范化一致的  $\hat{y}$  中,我们选择最小化  $\|\hat{y}\|_2$  的一个。

(a) 展示如何使用凸优化或准凸优化找到  $\hat{y}$ 。

(b) 将您的方法应用于  $zero\_crossings\_data\_*$  中的数据问题实例。数据文件还包括真实信号 y(当然您不能使用它来找到  $\hat{y}$ )。绘制  $\hat{y}$  和 y,并报告相对恢复误差, $\|\hat{y}-y\|_2/\|y\|_2$ 。简短评注恢复的质量。

## 1.1解答

(a) 此问题的估计可以表示为  $\hat{y}=Ax$ ,其中  $x=(a,b)\in\mathbb{R}^{2B}$  表示余弦和正弦系数向量。定义矩阵  $A=[C\quad S]\in\mathbb{R}^{n imes2B}$ ,其中矩阵  $C,S\in\mathbb{R}^{n imes B}$  的元素分别定义为:

$$C_{tj} = \cosigg(rac{2\pi(f_{\min}+j-1)t}{n}igg), \quad S_{tj} = \sinigg(rac{2\pi(f_{\min}+j-1)t}{n}igg),$$

为确保  $\hat{y}$  与 s 的符号一致,必须满足约束  $s_t a_t^T x \geq 0$  (对于  $t=1,\ldots,n$  成立,其中  $a_1^T,\ldots,a_n^T$  是矩阵 A 的行)。此外为实现适当的规范化,还需要线性等式约束  $\|\hat{y}\|_1 = s^T A x = n$  (尽管通常  $l_1$ -范数等式约束非凸,但在此情境下由于符号确定故是凸的)。

综上所述,得到的优化问题是凸的,具体表述如下:

$$egin{array}{lll} \min & \|Ax\|_2 \ ext{s.t.} & s_t a_t^T x \geq 0, & t = 1, \ldots, n \ & s^T A x = n. \end{array}$$

解决此优化问题后,可以得到  $\hat{y}=Ax^*$ ,其中  $x^*$  是该问题的解。

(b) 使用 python 调用 cvxpy 库求解该优化问题,恢复误差(RRMS)为0.12083136047814856。

zero\_crossings\_data.py 模块提供实验数据和相关参数。在信号处理的研究和实验中,数据的准备是基础且关键的一步。该模块定义函数 zero\_crossings\_data 并返回了信号的长度、最小频率、带宽、信号的符号数组 以及真实信号的样本值。

optimization\_solve.py 模块使用cvxpy库函数解决带限信号恢复问题的优化问题,以从给定的零交叉信息中恢复带限信号。此模块包含了定义目标函数、设置约束条件以及调用优化求解器。

```
In [ ]:
        import numpy as np
        import cvxpy as cvx
        def optimization_solve(n, f_min, B, s, y):
            # 构造矩阵 A,其列为带限正弦波
            C = np.zeros((n, B))
            S = np.zeros((n, B))
             for j in range(B):
                C[:, j] = np.cos(2 * np.pi * (f min + j) * np.arange(1, n + 1) / n)
                S[:, j] = np.sin(2 * np.pi * (f_min + j) * np.arange(1, n + 1) / n)
            A = np.hstack((C, S))
             # 最小化目标函数. 同时考虑 L1 规范化和符号约束
            x = cvx.Variable(2 * B)
             obj = cvx.norm(A @ x)
             constraints = [cvx.multiply(s, A @ x) >= 0, s.T @ (A @ x) == n]
             problem = cvx.Problem(cvx.Minimize(obj), constraints)
             problem.solve(solver=cvx.ECOS)
            y_recovered = A @ x.value
             # RRMS (Relative Root Mean Square Error) 误差计算
             RRMS = np.linalg.norm(y - y recovered) / np.linalg.norm(y)
             print('Optimize Success')
             return y_recovered, RRMS
```

results\_output.py 模块处理和展示输出结果,绘制图表并保存数据。

```
In []:
         import matplotlib.pyplot as plt
         import numpy as np
         import pandas as pd
         import os
         def plot_and_save_with_csv(y1_data, title, xlabel, ylabel, output_folder_path, fig_size=(24, 5), plot_type='plot', y2_data=None, y2_label=None, legend1=None, legend2=None):
             # 确保输出文件夹存在
            if not os.path.exists(output folder path):
                os.makedirs(output_folder_path)
             # 绘制图像
            fig, ax1 = plt.subplots(figsize=fig_size)
            if plot type == 'plot':
                ax1.plot(y1_data, color='tab:blue', label=legend1)
             elif plot type == 'stem':
                 ax1.stem(y1_data, linefmt='tab:blue', label=legend1)
             ax1.set xlabel(xlabel)
             ax1.set ylabel(ylabel)
             ax1.tick_params(axis='y')
             # 第二个 Y 轴的处理
            if y2_data is not None and y2_label is not None:
                ax2 = ax1.twinx()
                 ax2.plot(y2_data, '--', color='tab:red', label=legend2)
                ax2.set ylabel(y2 label)
                ax2.tick params(axis='y')
            plt.title(title)
             # 显示图例
             if legend1 is not None:
                 ax1.legend(loc='upper left')
             if y2 data is not None and legend2 is not None:
                 ax2.legend(loc='upper right')
             plt.tight_layout()
             # 保存图像
            safe_title = title.replace(' ', '_').replace('.', '').replace('-', '_')
             image file name = f"{safe title}.jpg"
             plt.savefig(os.path.join(output folder path, image file name))
             plt.show()
             # 合并数据并保存到 CSV
             csv file name = f"{safe title}.csv"
             if y2 data is not None:
                # 确保数据是列表格式,即使只有一个元素
                y1 data = [y1 data] if np.isscalar(y1_data) else y1_data
                y2_data = [y2_data] if np.isscalar(y2_data) else y2_data
                combined data = pd.DataFrame({legend1: y1 data, legend2: y2_data})
                y1_data = [y1_data] if np.isscalar(y1_data) else y1_data
                combined_data = pd.DataFrame({legend1: y1_data})
             combined data.to csv(os.path.join(output folder path, csv file name), index=False, header=True)
```

main.py 是整个程序的入口点,负责协调上述三个模块的功能,完成从零交叉恢复带限信号的整个流程。在 main.py 中先调用 zero\_crossings\_data.py 来获取必要的数据和参数,然后利用 optimization\_solve.py 中的算法对数据进行处理和分析,最后通过 results\_output.py 模块来展示和评估结果。

```
import os
from module.zero_crossings_data import zero_crossings_data
from module.optimization_solve import optimization_solve
from module.results_output import plot_and_save_with_csv

n, f_min, B, s, y = zero_crossings_data()

y_recovered, RRMS = optimization_solve(n, f_min, B, s, y)

# 輸出路径
current_dir = os.path.dirname(os.path.realpath(__file__))
results_output_folder_path = os.path.join(current_dir, 'outputs/results')
error_output_folder_path = os.path.join(current_dir, 'outputs/rerors')

# 輸出結果
# 绘制網路信号y和估计信号y_recovered的图像
plot_and_save_with_csv(y, 'original_and_recovered_bandlimited_signals', 'Sample_Index', 'bandlimited_signals', results_output_folder_path, y2_data=y_recovered, y2_label='bandlimited_signals', plot_and_save_with_csv(y, 'original_and_recovered_bandlimited_signals', 'sample_Index', 'bandlimited_signals', results_output_folder_path, y2_data=y_recovered, y2_label='bandlimited_signals', plot_and_save_with_csv(RRMS, 'RRMS_Error', 'Iteration', 'Error', error_output_folder_path, legendl='RRMS')
```

