科学与社会研讨课报告
基于树莓派平台的地震仪设计
和利用互相关函数计算台站钟差及格林函数的实践

作者：廖荣　　指导老师：李俊伦　　助教：刘煜川、郭畅、陈国艺

2022 年 6 月 8 日

目录

摘要：作为中国科学技术大学新生“科学与社会”研讨课报告，本文利用树莓派平台、三维检波器和模数转换芯片搭建了简易地震仪；并在实地数据测量中利用互相关函数得到了自制地震仪与具有 GPS 自动授时功能的专业地震仪之间的台站钟差；进一步利用互相关技术处理三个专业地震仪的同一时段噪声记录，得到了台站对之间的格林函数。

关键词： **地震仪**，**背景噪声**，**互相关**，**台站钟差**，**格林函数**

1 引言

2 基于树莓派平台的地震仪设计

2.1 硬件设计

2.1.1 硬件概况

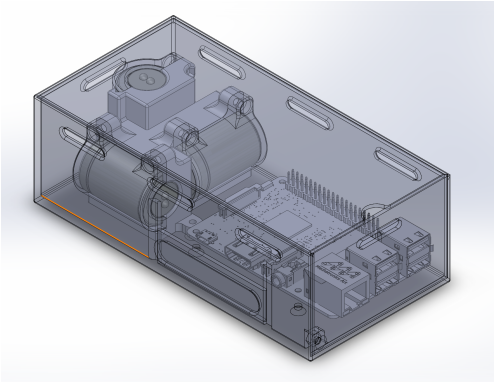


图 1: 自制地震仪内部结构

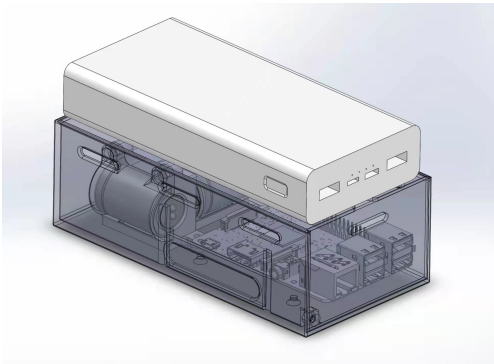


图 2: 自制地震仪整体结构

图 3: 地震检波器

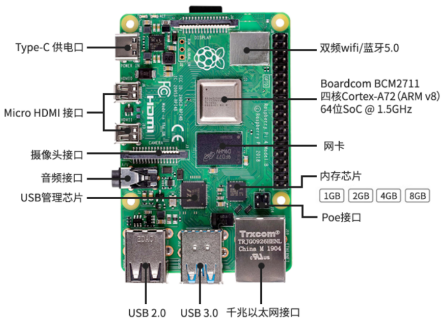


图 4: 树莓派平台

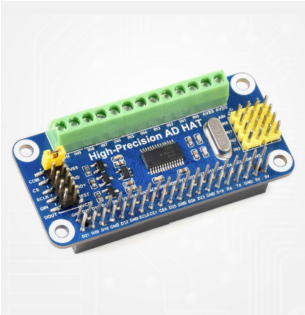


图 5: 模数转换模块

自制地震仪由六部分组成，三个方向互相垂直的检波器，树莓派，模数转换模块，外接电源，导线和 3D 打印外壳。检波器系统由保定兆峰传感设备公司生产的两个横波检波器和一个垂直检波器组成，详细参数均为自然频率 $4.5Hz$, 开路电阻 450Ω。树莓派平台选用 Raspberry Pi4 Model B。模数转换模块为大陆胜公司生产的适配树莓派的 10 通道高精度模数扩展板，搭载 ADS1263 芯片。外接电源为小米公司生产的 20000mAh、USB-C 移动电源。外壳由聚合物 3D 打印得到。

2.1.2 硬件工作原理

自制地震仪的工作原理核心分为两部分，一部分为检波器对地震信号的记录和转换，另一部分为搭载 ADS 模块的树莓派平台对电压信号的处理。

电动式地震检波器的外形和结构如图6所示。它由永久磁铁、线圈和弹簧片组成，磁铁具有很强的磁性，它是地震检波器的关键部件；线圈由铜漆包线绕在框架上而成，有两个输出端，它也是地震检波器的关键部件；弹簧片由特制的磷青铜做成一定的形状，具有线性弹性系数，它使线圈与塑料盖连在一起，使线圈与磁铁形成一个相对运动体 (惯性体)。当地面存在机械振动时，线圈对磁铁做相对运动而切割磁力线，根据电磁感应原理，线圈中产生感生电动势，且感生电动势的大小与线圈和磁铁的相对运动速度成正比，线圈输出的模拟电信号与地面机械振动的速度变化规律是一致的。

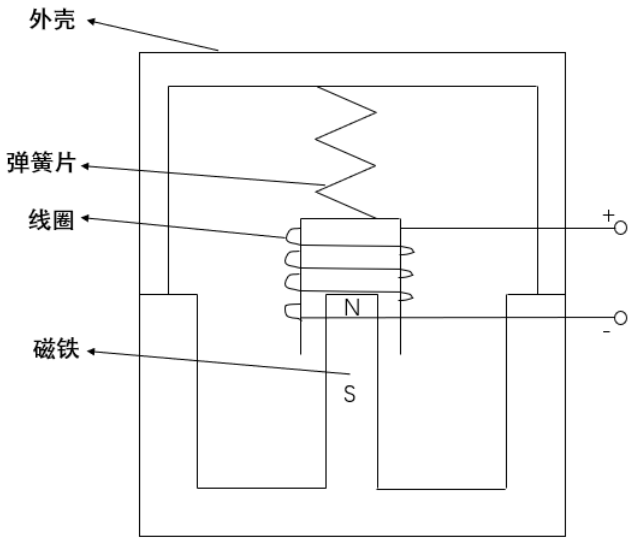


图 6: 检波器结构原理示意图 [1]

检波器接收地面机械振动后通过两个输出端输出的模拟电信号直接由导线传入 ADS 模数转换模块，ADS 芯片将模拟电信号转换成为数字信号后，将信号传到树莓派平台内存中，由树莓派 CPU 控制将数据写入磁盘或送入管道进行绘图操作。

2.2 软件设计

软件部分主要由”read_and_save” 数据读写存储函数和”plot_pipeline” 绘图函数组成。

2.2.1 数据读写存储函数

”read_and_save” 数据读写存储函数完成从 ADS 芯片读取数据并将其发送到 pipeline 文件的任务。其具体设计思路如下：

首先初始化硬件，检查 ADS 芯片程序工作和 pipeline 管道文件打开是否正常。

初始化硬件后，在第一次采样前，使用微秒量级精度 timeval 结构体记录当前系统时间为采样初始时间，将其向上取整到整秒位并加上系统初始化所需要的时间 2s，记为结构体变量 tbegin，后将初始时间调整到北京时区和 ISO 单位制并存入文件。

计算采样起始时间后，设计了 fetchdata 函数实现定时采样。后在记录采样时间时，对每一个采样点，记录采样开始时间和采样结束时间，并以采样起始时间 tbegin 为 0 时刻记录相对时间。最终每个采样点的数据形式为“开始时间-结束时间-电压值”。

采样完成后，选择将数据发送到缓冲区（当前默认不执行），发送到数据存储文件（当前默认执行），或是发送到管道文件 pipeline（当前默认执行）。当单个数据存储文件中存满 1000 个数据（按 100Hz 采样即十秒后），新建文件继续存储。当存储文件满十个后，从第一个文件开始刷新覆盖。

整个过程在一个循环结构中执行。

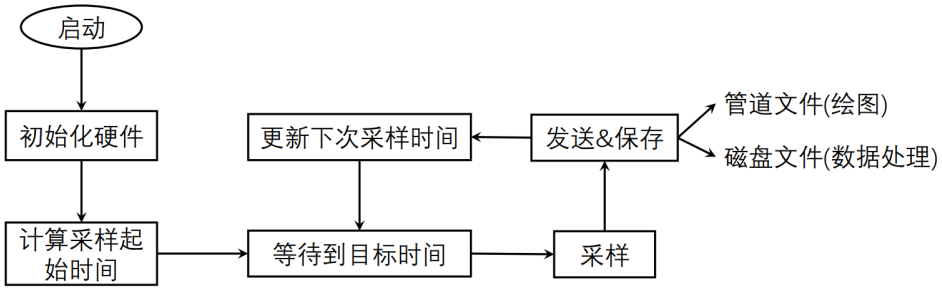


图 7: “read_and_save”数据读写存储函数流程图

给出代码如下 [2]：

```
/*
read_and_save.c
    read data from ADC and send to pipeline/disk.
    written by TA.
*/
#include <math.h>
#include <signal.h> //signal()
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/time.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <time.h>
#include <unistd.h>
#include "config.h"
#include "../include/ADS1263.h"
#define REF (REFERENCE_VOLTAGE)
#define CHANNEL 0

void Handler(int signo);
void fetchdata(struct timeval *tv, SAMPLE_VAR_TYPE *buf);
double timerdiff(struct timeval *a, struct timeval *b);
```

```
void writebuffer(SAMPLE_VAR_TYPE *t1, SAMPLE_VAR_TYPE *t2, SAMPLE_VAR_TYPE *w1);
int pipefile = 0, varsize = 0;

int main()
{
    // recorder part
    int i_plot = 0, i_data = 0, ifile = 0;
    SAMPLE_VAR_TYPE t1, t2,
        vt1[BUFFER_FILE_SIZE],
        vt2[BUFFER_FILE_SIZE],
        vw[BUFFER_FILE_SIZE],
        bufw[1], pipebuf[3];
    char filename[40],
        ISOFORMAT[DATE_STR_LEN] = "%Y/%m/%d %H:%M:%S",
        datestr[DATE_STR_LEN];
    FILE *fid = NULL;
    varsize = sizeof(t1);
    struct timeval tbegin, ttarget, treal, tstep;
    struct tm *ct;

    // init
    signal(SIGINT, Handler);
    DEV_Module_Init();
    ADS1263_SetMode(1); // set ADC to differential mode
    if (ADS1263_init() == 1) // exit when error
    {
        printf("\n END \n");
        DEV_Module_Exit();
        exit(0);
    }
    if (USE_PIPELINE_FILE) // open pipeline file
    {
        if (access(pipefilename, F_OK) == -1)
        {
            i_data = mkfifo(pipefilename, 0777);
            if (i_data != 0)
            {
                printf("Could not create Pipeline file\n");
                exit(0);
            }
        }
        printf("Opening pipeline file...\n");
```

```
    pipefile = open(pipefilename, O_WRONLY);
    if (pipefile == -1)
    {
        printf("Opening pipeline file failed, exit.\n");
        exit(0);
    }
}
// use tstep as tmp var, calc start time
timerclear(&tstep);
tstep.tv_sec = (long int)INIT_COST;
tstep.tv_usec = ((long int)(INIT_COST * 1e6)) % 1000000;
gettimeofday(&tbegin, NULL); // get current time
tbegin.tv_sec++;
tbegin.tv_usec = 0; // start from the whole second
timeradd(&tbegin, &tstep, &ttarget); // add starting time
tbegin = ttarget;
// print info to terminal
ct = localtime(&tbegin.tv_sec);
strftime(datestr, DATE_STR_LEN, ISOFORMAT, ct);
printf("Start recording after %fs at %s.%03d\n", INIT_COST,
       datestr, (int)(tbegin.tv_usec * 1e-6));
// write starttime to file
sprintf(filename, "../dat/starttime.txt");
fid = fopen(filename, "w");
fprintf(fid, "%s.%03d\n", datestr, (int)(tbegin.tv_usec * 1e-6));
fclose(fid);
// init tstep
timerclear(&tstep);
tstep.tv_usec = TIME_DELTA;
printf("Recording...\n");
i_data = 0;
while (TRUE)
{
    if (i_data == 0)
    {
        sprintf(filename, "../dat/data%d.dat", ifile);
        fid = fopen(filename, "wb");
    }
    // read from ADC
    treal = ttarget;
    fetchdata(&treal, bufw);
    t1 = (SAMPLE_VAR_TYPE)timerdiff(&ttarget, &tbegin); // theoretical time
```

```
        t2 = (SAMPLE_VAR_TYPE)timerdiff(&treal, &tbegin);   // real time
        timeradd(&ttarget, &tstep, &ttarget);
        // write to file
        // to terminal
        printf("%7.4f %7.4f %f\n", t1, t2, bufw[0]);
        // to plot buffer
        if (USE_BUFFER_FILE)
        {
            vt1[i_plot] = t1;
            vt2[i_plot] = t2;
            vw[i_plot] = bufw[0];
            i_plot = (i_plot + 1) % BUFFER_FILE_SIZE;
            if (i_plot == 0)
                writebuffer(vt1, vt2, vw);
        }
        // to data file
        fwrite(&t1, varsize, 1, fid);
        fwrite(&t2, varsize, 1, fid);
        fwrite(bufw, varsize, 1, fid);
        i_data = (i_data + 1) % DATA_FILE_SIZE;
        if (i_data == 0)
        {
            fclose(fid);
            ifile = (ifile + 1) % DATAFILE_NUM_MAX;
        }
        // to pipeline
        if (USE_PIPELINE_FILE)
        {
            pipebuf[0] = t1;
            pipebuf[1] = t2;
            pipebuf[2] = bufw[0];
            pipeline_send(pipefile, pipebuf, varsize*3);
        }
    }
    return 0;
}

void Handler(int signo)
{
    //System Exit
    if (USE_PIPELINE_FILE)
    {
```

```
        close(pipefile);
        unlink(pipefilename);
    }
    DEV_Module_Exit();
    printf("\nEND\n");
    exit(0);
}

void fetchdata(struct timeval *tv, SAMPLE_VAR_TYPE *buf)
{
    struct timeval ct;
    unsigned int slp;
    SAMPLE_VAR_TYPE dat;
    UDOUBLE ADC;
    // calculate how long to sleep
    gettimeofday(&ct, NULL);
    slp = (tv->tv_sec - ct.tv_sec) * 1e6 + tv->tv_usec - ct.tv_usec - TIME_EPS;
    if (slp > 0)
        usleep(slp);
    // wake up and read data
    ADC = ADS1263_GetChannalValue(CHANNEL);
    // transpose COUNT -> voltage
    if (ADC >> 31 == 1)
        buf[0] = -(REF * 2 - ADC * REF / 0x80000000);
    else
        buf[0] = ADC * REF / 0x7FFFFFFF;
    // write finish time
    gettimeofday(tv, NULL);
}

// calculate time difference between to time point
double timerdiff(struct timeval *a, struct timeval *b)
{
    return a->tv_sec - b->tv_sec + (a->tv_usec - b->tv_usec) * 1e-6;
}

// write data to buffer file(no longer used)
void writebuffer(SAMPLE_VAR_TYPE *t1, SAMPLE_VAR_TYPE *t2, SAMPLE_VAR_TYPE *w1)
{
    FILE *fp;
    fp = fopen(bufferfilename, "wb");
    fwrite(t1, varsize, BUFFER_FILE_SIZE, fp);
```

```
    fwrite(t2, varsize, BUFFER_FILE_SIZE, fp);
    fwrite(w1, varsize, BUFFER_FILE_SIZE, fp);
    fclose(fp);
}
```

2.2.2 绘图函数

“plot_pipeline”绘图函数完成从管道文件中读取数据，写入绘图库缓存并绘图的任务。其具体设计思路如下：

首先初始化缓存和 OpenGL 库，将所有绘图点数值归零。后从管道中读取数据，具体通过比对系统读取数据的前后时间差和管道文件更新的时间差（即采样频率的倒数）实现了当且仅当管道中有数据时获取数据并写入绘图库缓存。后调用 OpenGL 绘图库中的一系列函数完成了清屏，绘制参考线，设置窗口大小和位置，设置画面背景色，设置图线宽度和颜色，实现窗口中图像滚动刷新和设置退出绘图（关闭图层）键等操作。最终将电压值数据以随时间变更的地震波形式呈现在 Linux Ubuntu18.04 的用户屏幕上。

读取数据，写入内存和绘图函数封装在循环结构中，可通过指定键盘按键退出。

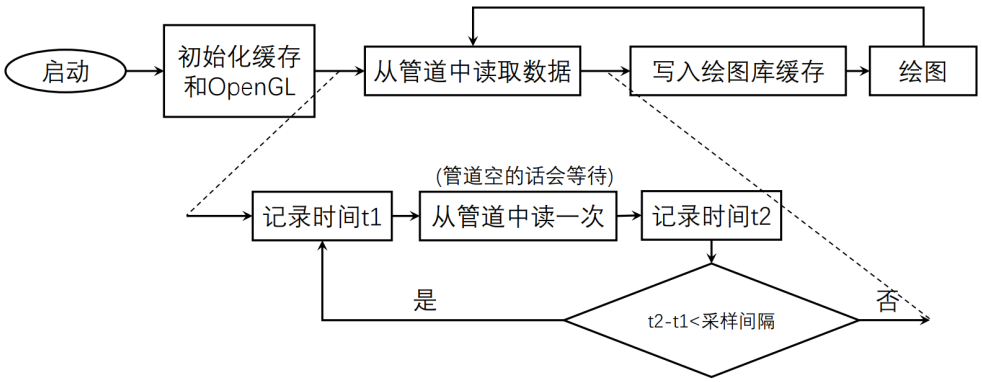


图 8: “plot_pipeline”绘图函数流程图

给出代码如下 [3]:

```
/*
plot_pipeline.c
    read waveform from pipeline file and plot
    written by TA
*/
#include <stdio.h> // standard library
#include <stdlib.h>
#include <GL/gl.h> // library of OpenGL
#include <GL/glu.h> // library of OpenGL
#include <GL/glut.h> // library of OpenGL
#include <math.h>
#include <unistd.h>
#include <sys/stat.h>
#include <sys/time.h>
```

```
#include <time.h>
#include <fcntl.h>
#include "config.h"
#define SAMPLE_VAR_NUM 3
#define SCALE_FACTOR 1.2
SAMPLE_VAR_TYPE tend = -1.0, t[PLOT_BUFFER_SIZE],
                w[PLOT_BUFFER_SIZE],
                refline[1];
int pipefile = 0;

 // read one sample form pipeline file
 // each sample contains 2 time and 1 data
void readonesample(void)
{
    int i;
    SAMPLE_VAR_TYPE t1, t2, tw, bf[SAMPLE_VAR_NUM];
    pipeline_receive(pipefile, bf, SAMPLE_VAR_NUM);
    t1 = bf[0];
    t2 = bf[1];
    for (i = 0; i < PLOT_BUFFER_SIZE - 1; i++)
    {
        w[i] = w[i + 1];
    }
    w[i] = bf[SAMPLE_VAR_NUM - 1];
    tend = t1;
    printf("o %.4f %.6f\n", tend, w[PLOT_BUFFER_SIZE - 1]);
}

void updatebuffer(void) // update buffer data
{
    struct timeval t1, t2;
    int tt, tr=TIME_DELTA/2;
    do{
        gettimeofday(&t1, NULL);
        readonesample();
        gettimeofday(&t2, NULL);
        tt = (t2.tv_sec-t1.tv_sec)*1000000+t2.tv_usec-t1.tv_usec;
    }while(tt<tr); // read sample until pipeline is empty
}

void display(void) /* function called whenever redisplay needed */
{
```

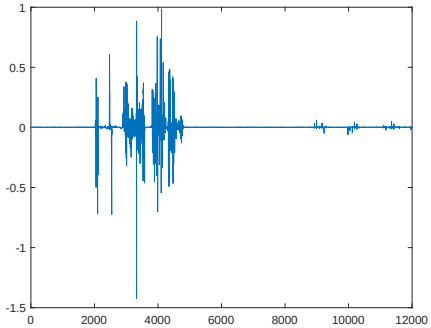
```
    int i;
    updatebuffer();
    glClear(GL_COLOR_BUFFER_BIT); // clear the display
    // add Reference lines
    glColor3f(0.8, 0.8, 0.8); // set color
    for (i = 0; i < 1; i++)
    {
        // top
        glBegin(GL_LINES);
        glVertex2d(t[0], refline[i] + WAVEAMP);
        glVertex2d(t[PLOT_BUFFER_SIZE - 1], refline[i] + WAVEAMP);
        glEnd();
        // bottom
        glBegin(GL_LINES);
        glVertex2d(t[0], refline[i] - WAVEAMP);
        glVertex2d(t[PLOT_BUFFER_SIZE - 1], refline[i] - WAVEAMP);
        glEnd();
        // center
        glBegin(GL_LINES);
        glVertex2d(t[0], refline[i]);
        glVertex2d(t[PLOT_BUFFER_SIZE - 1], refline[i]);
        glEnd();
    }
    // wave
    glColor3f(LINECOLOR_R, LINECOLOR_G, LINECOLOR_B);
    glBegin(GL_LINE_STRIP);
    for (i = 0; i < PLOT_BUFFER_SIZE; i++)
        glVertex2d(t[i], w[i] + refline[0]);
    glEnd();
    glFlush();
    glutSwapBuffers();
}

void keyCB(unsigned char key, int x, int y) // called on key press
{
    if (key == 'q')
    {
        close(pipefile);
        exit(0);
    }
}
```

```
int main(int argc, char *argv[])
{
    int win, i;
    SAMPLE_VAR_TYPE dt = 0.1;
    printf("Cleaning buffer...\n");
    t[0] = 0.0;
    for (i = 1; i < PLOT_BUFFER_SIZE; i++)
    {
        t[i] = t[i - 1] + dt; // fixed x coordinate
        w[i] = 0.0; // initial y coordinate
    }

    refline[0] = 0.0;

    // open pipeline file
    printf("Opening pipeline file...\n");
    pipefile = open(pipefilename, O_RDONLY);
    if (pipefile == -1)
    {
        printf("Opening pipeline file failed.\n");
        exit(0);
    }

    // plot
    glutInit(&argc, argv);                          // initial glut
    glutInitWindowPosition(WINDOW_X0, WINDOW_Y0);   // set window left top position
    glutInitWindowSize(WINDOW_WIDTH, WINDOW_HEIGHT); // set window size
    glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE);    // set display mode
    win = glutCreateWindow("plot pipeline");
    glClearColor(BACKGROUND_R, BACKGROUND_G, BACKGROUND_B, BACKGROUND_A);
    glLineWidth(WAVE_LINE_WIDTH);
    gluOrtho2D(t[0],
               t[PLOT_BUFFER_SIZE - 1],
               refline[0] - WAVEAMP * SCALE_FACTOR,
               refline[0] + WAVEAMP * SCALE_FACTOR);
    glutDisplayFunc(display);
    glutKeyboardFunc(keyCB);
    glutIdleFunc(display);
    glutMainLoop();
    return 0;
}
```

3 利用互相关函数计算台站钟差

地震记录的时间准确性对一些时间精度要求较高的地震研究至关重要，特别是对于地震预警、地震定位以及地壳结构层析成像等研究。地震台站授时主要由两部分完成，一是通过 GPS（全球定位系统）进行授时，二是由地震观测设备自带的晶振计时。本文涉及的专业地震仪具有 GPS 授时功能。但自制地震仪的授时来自其树莓派平台 CPU 的系统时间，其时间依赖连接互联网后的服务器授时，在野外工作条件下适用性差。近年来多项研究 [4-5] 均表明，可以使用互相关函数对背景噪声进行分析，从而计算台站钟差。

利用专业地震仪和自制地震仪于 2022 年 6 月 1 日 18:37:30-18:39:30 以 100Hz 采样的波形数据进行台站钟差分析，原始数据如下（两份数据非 1：1 比例）。



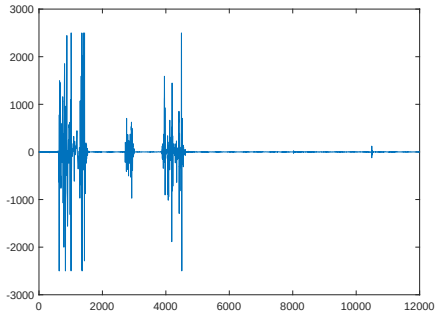


图 9: 自制地震仪采样数据（100Hz，2min）图 10: 专业地震仪采样数据（100Hz，2min）

由于自制地震仪并没有耦合地面设计，故可以直观看出，其所记录的人工激发的强震源信号与专业地震仪记录的波形差距很大，故直接观察相位差较为困难且会造成显著误差，本文选择使用 6000-8000 号背景噪声数据进行互相关分析，尝试获得台站钟差。

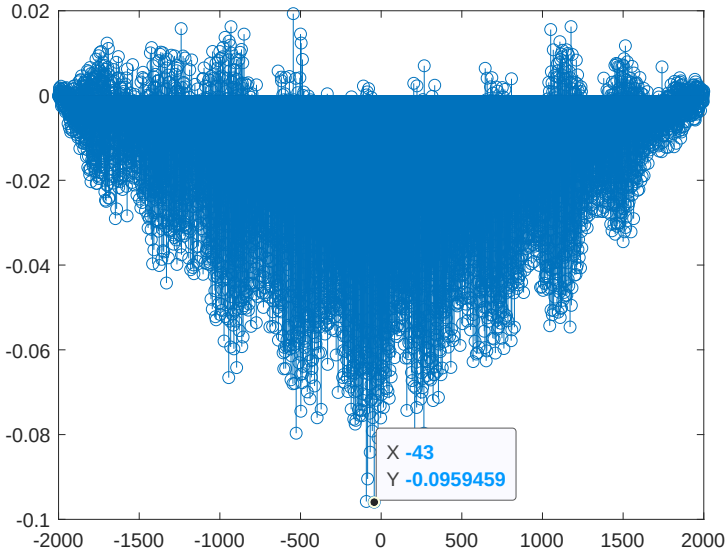


图 11: 自制地震仪与专业地震仪背景噪声互相关结果

可以看出，时间滞后分布结果较好，自制地震仪大致与专业地震仪计时相差 0.43s。

4 利用背景噪声互相关提取格林函数

研究表明，声学扩散场中两点之间的格林函数可以通过两点记录到的背景噪声信息进行互相关来提取。[6] 而固体地球物理学科中的背景噪声是指没有地震发生的情况下地震仪上记录到的随机波动信号，是一种连续稳定的自然源，包含了丰富的地下介质信息。1957 年地震学家 Aki 首次提出利用背景噪声信息来研究地球内部构造 [7]，而近年来众多地球物理地震成像研究 [8-9] 表明，对长时间的地震背景噪声进行互相关计算可以得到一个正负轴近似对称的波形。对其进行处理后，可以得到台站对之间的格林函数（包含台站对间地震波在介质中传播的所有信息）。

故本文尝试通过分析三台专业地震仪收集到的背景噪声信号，运用互相关函数处理得到台站对间的格林函数。本文实验中将三个专业地震仪台站各自相距 5m 呈正三角形状摆放并按 100Hz 的频率对背景噪声进行记录，示意图如下。

图 12: 背景噪声记录现场示意图

根据 Bensen 提出的标准化流程 [10]，通过背景噪声互相关提取格林函数的步骤为：

第一步首先对单个台站数据预处理，包括去趋势，去均值，去仪器响应，带通滤波处理等步骤，其中最重要的是时域归一化，一般采用“one-bit”正则化方法，目的是去除地震引起的畸变信号和附近噪声对台站互相关计算的影响，另外还应进行频谱白化，用于拓宽互相关信号的频谱，降低某一单频固定信号的干扰。最后把数据切割到以一天为长度。第二步为计算台站对间互相关，并把日相关性叠加到所需天数。第三步为测量组或相速度。第四步为误差分析和可接受测量值的选择。

由于本文的技术条件和数据长度有限，故只选择进行单台站数据预处理中的时域归一化和台站对间互相关计算两步计算，得到简易的格林函数。其计算过程如下 [11]：

对于两路平稳随机信号 $x(t)$ 和 $y(t)$, 其互相关函数通常用式 (1) 计算，即

$$R_{xy}(\tau) = \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} y(t)x(t-\tau)dt \tag{1}$$

(1) 式中，积分时间为无穷大，不符合实际计算情况，实际运算应在有限的积分时间 T 内计算互相关函数的估计值，即

$$\hat{R}_{xy}(\tau) = \frac{1}{T} \int_{0}^{T} y(t)x(t-\tau)dt \tag{2}$$

(2) 式中 $\hat{R}_{xy}(\tau)$ 表示随机信号 $x(t)$ 和 $y(t)$ 的互相关函数 $R_{xy}(\tau)$ 的估计值。为了能使该计算能够通过计算机程序执行，将式 (2) 离散化得到式 (3), 即

$$\hat{R}_{xy}(m) = \frac{1}{N}\sum_{n=0}^{N-1} y(n)x(n-m) \tag{3}$$

$$m = 0, 1, 2, \cdots, M-1$$

(3) 式中，$\hat{R}_{xy}(k)$ 表示时间序列 $x(n)$ 和 $y(n)$ 互相关的数值；$x(n)$ 和 $y(n)$ 式随机信号 $x(t)$ 和 $y(t)$ 的时间采样序列；采样周期为 T，$t = nT$;N 表示在有限的积分时间内的采样点数；m 为时间延迟的序号; 延迟时间 $\tau = mT$.

对时间序列做归一化，使得零滞后时的自相关为 1，即

$$\hat{R}_{xy,coeff}(m) = \frac{1}{\sqrt{\hat{R}_{xx}(0)\hat{R}_{yy}(0)}}\hat{R}_{xy}(m) \tag{4}$$

(4) 式即为包含时域归一化处理的台站对间互相关函数式，可在 matlab 中使用并处理数据。

给出 matlab 代码如下：

```
*/
   plot waveform from datafile
   and do correlations
   written by liaorong
/*
Seis_own=importdata('Seis_own.txt');
Seis_prof=importdata('Seis_prof.txt');
plot(Seis_own);
plot(Seis_prof);
Selected_Seis_own=Seis_own(6000:8000,1);
Selected_Seis_prof=Seis_prof(6000:8000,1);
[c,lags]=xcorr(Selected_Seis_own,Selected_Seis_prof,'normalized');
stem(lags,c);

sta1=importdata('sta1.txt');
sta2=importdata('sta2.txt');
sta3=importdata('sta3.txt');
plot(sta1);
plot(sta2);
plot(sta3);
[c12,lags12]=xcorr(sta1,sta2,'normalized');
[c13,lags13]=xcorr(sta1,sta3,'normalized');
[c23,lags23]=xcorr(sta2,sta3,'normalized');
stem(lags12,c12);
stem(lags13,c13);
stem(lags23,c23);
stem((lags12+lags13+lags23)/3,(c12+c13+c23)/3)
```

三台地震仪于 2022 年 6 月 1 日 18:47:00-18:57:00 以 100Hz 采样的波形原始数据如下。

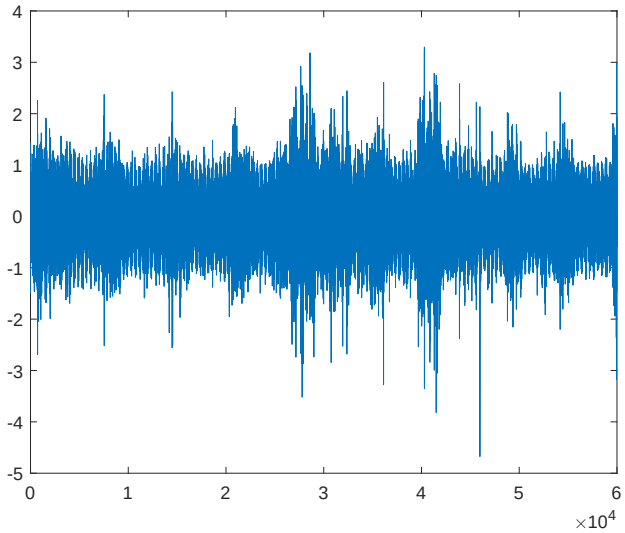


图 13: 1 号地震仪背景噪声波形数据

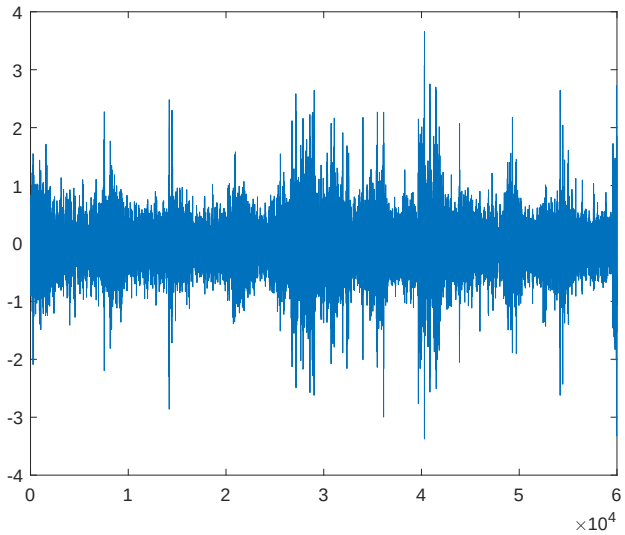


图 14: 2 号地震仪背景噪声波形数据

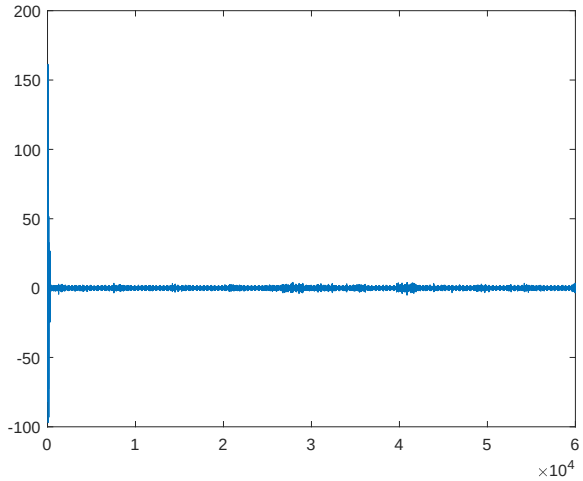


图 15: 3 号地震仪背景噪声波形数据

三台地震仪的背景噪声互相关结果及格林函数如下。

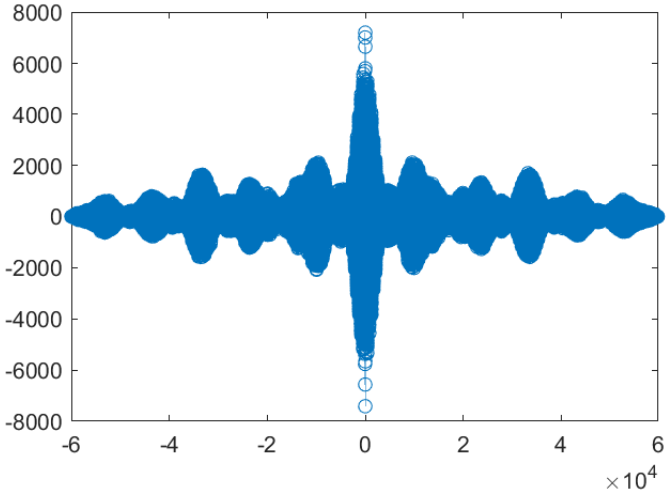


图 16: 1 号与 2 号地震仪背景噪声互相关

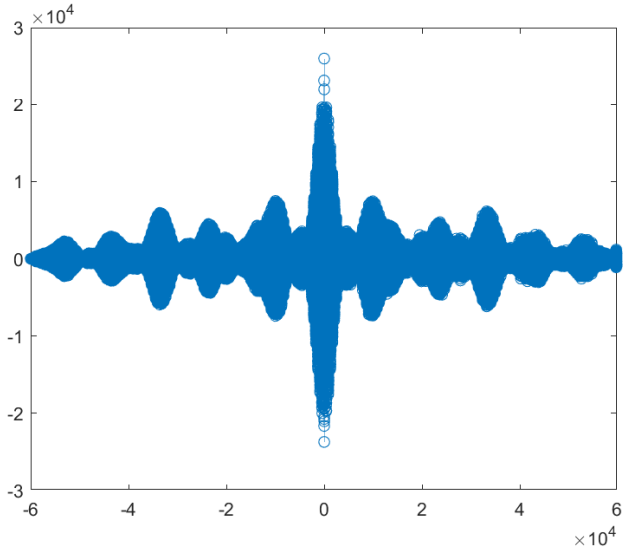


图 17: 1 号与 3 号地震仪背景噪声互相关

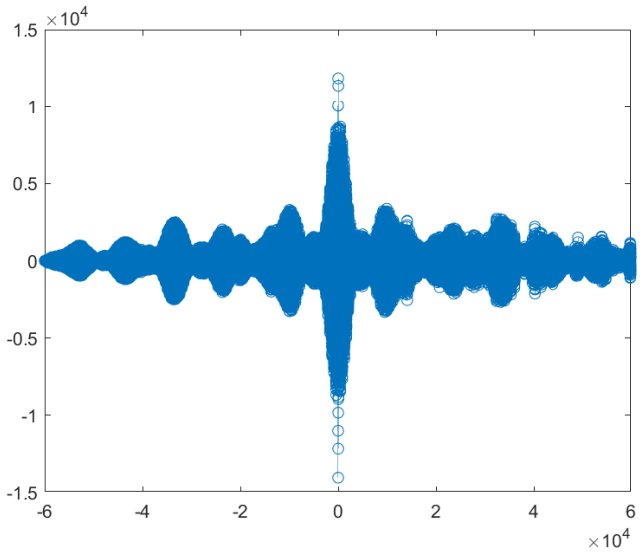


图 18: 2 号与 3 号地震仪背景噪声互相关

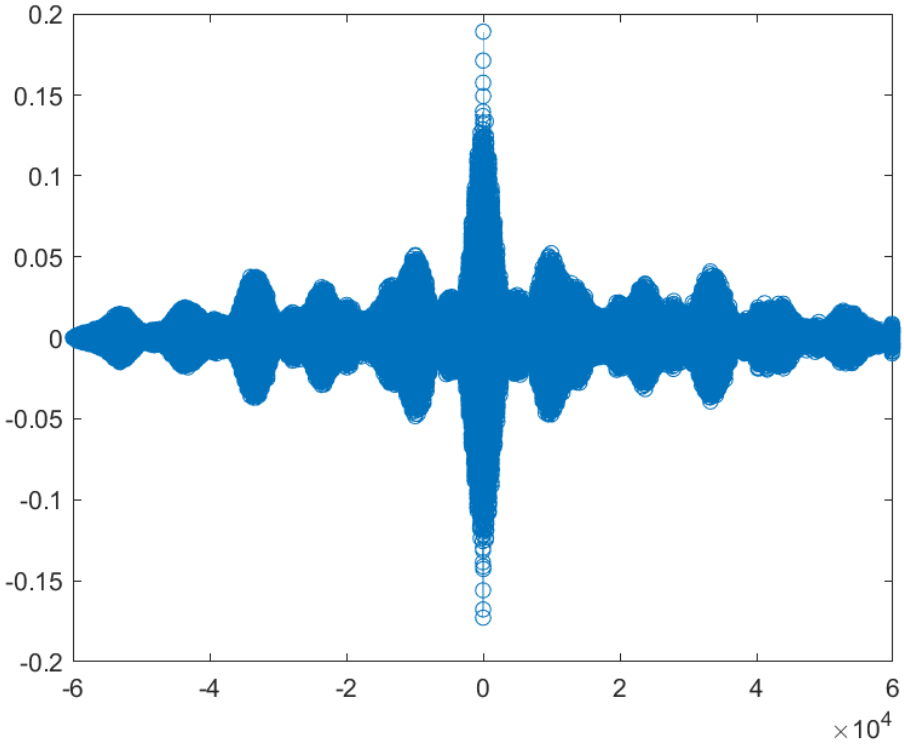


图 19: 三组互相关数据平均叠加结果（格林函数）

参考文献

[1] 袁子龙. 地震勘探仪器原理（第二版）[M]. 石油工业出版社，2016：14.

[2] Stephen Prata. C Primer Plus（第 6 版）中文版 [M]. 人民邮电出版社，2016.

[3] Richard S. Wright and Benjamin Lipchak and Nicholas Haemel. OpenGL SuperBible: Comprehensive Tutorial and Reference (4th Edition) [M]. 人民邮电出版社，2010.

[4] 王俊, 郑定昌, 詹小艳, 江昊琳, 缪发军, 高景春. 基于背景噪声互相关格林函数的单台时间误差估计 [J]. 地震学报,2013,35(06):888-901+938.

[5] 胡梦迪, 金星, 李军. 利用噪声互相关技术计算福建台网台站绝对钟差 [J]. 防灾减灾学报,2020,36(03):1-7.DOI:10.13693/j.cnki.cn21-1573.2020.03.001.

[6] Weavor R L,Lobkis O I. Ultrasonics without a source: Thermal fluctuation correlations at MHz frequencies [J]. Physical Review Letters,2001,87(13).

[7] Aki K. Space and time spectra of stationary stochastic waves, with special reference to microtremors [J]. Bulletin of the Earthquake Research Institute, 1957,35(3):415-456.

[8] Shapiro N M et al., Emergence of broadband Rayleigh waves from correlations of the ambient seismic noise[J], Geophys Res Lett, 2004, 31（7）, L07614（1-4）.

[9] Campillo M and Paul A. Long-range correlations in the diffuse seismic code[J]. Science,2003,299(5606:547-549).

[10] Bensen G D et al., Processing seismic ambient noise data to obtain reliable broad-band surface wave dispersion measurements[J]. Geophys J Int, 2007, 169（3）, 1239 -1260.

[11] 程佩青. 数字信号处理（第 2 版）[M]. 北京：清华大学出版社,2003:546-587.

A 致谢

本文是学生廖荣在李俊伦老师和刘煜川、郭畅、陈国艺三位助教的指导和帮助下完成的。衷心感谢李俊伦老师和三位助教一年间的辛勤付出，为我和其他八名大一年级的新生呈现了一堂近乎完美的新生“科学与社会”研讨课。研讨课期间，老师和助教充分考虑到大一新生数理基础和计算机能力尚不完善的情况，从最基础的数学工具 matlab 和 mathematica 开始教学，到 Linux 系统在虚拟机中的安装和使用，再到树莓派地震仪的具体搭建和调试，最终通过实地数据采样和分析完成了全部预定任务，让我和同学们在不断培养科研技能的同时，接触到了数字化、智能化地震勘探这一固体地球物理学科的前沿，收获颇丰。

另外让我感触最深的是，李俊伦老师用自己从南京大学电子信息学士到物理学硕士，再到 MIT 地球物理学博士的求学经历启发我们任何时候，任何阶段，做自己感兴趣的事业。这对刚入大学，对眼前事务和未来规划都一片迷茫却要做出一系列重要人生选择的新生无疑有着弥足珍贵的意义。对我个人而言，这激励我继续在基础学科领域学习研究，虽然不一定是固体地球物理学科，但往后的科研日子里，一定会记得我人生中第一堂科研课，第一位科研导师，第一个亲手完成的科研项目。