
程序编译：

编译./CRF_codes/src/下的程序时选用的 gcc 版本：应该没有影响，个人目前使用(load)的是 scl/gcc9.3，王新老师用的是 gcc4.8.5

编译 rjMCMCinv 时选用的 openmpi 版本：

```
module load openmpi/4.0.5_OPA_gcc9.3
```

编译完成后，请将/CRF_codes/rjMCMCinv 和/CRF_codes/bin/两个文件夹的绝对路径添加到环境变量。

可能的问题：

- (1) genCRF 运行报错：参见 **genCRF_修改记录.docx**
- (2) rjMCMCinv.o 编译出错：将 compile.sh 中的 forward_modeling.f90 修改为 forward_modeling_2.f90

程序运行：

数据准备（参见 CRF_readme.pdf）：

- (1) 接收函数存放方式：文件夹按事件分组，文件夹命名比较自由
- (2) 台站信息文件：共三列，stnm lat lon
- (3) 接收函数命名方式：由 proj_params.in 指定，其中一定要有 Z 分量/R 分量的区分标志以及台站名（见 proj_params.in）
- (4) P 波到时标记方式：在 t1（或 a 以及其他 tn）写入 P 波到时，然后在对应的 kt1（或 ka 以及其他 ktn）写入“P”，可参考 Step1_WritePArrival2SACheader.ipynb。注意，job_params.in 中的 marker，需要和这里写入 P 波到时的 sac 头段变量保持一致。

注意要准备足够长的 pre-event 的噪声段；谱白化要求出现在 2020 年的文章中，但似乎没有出现在 2021 年的文章中

执行 matlab 部分代码

在超算上执行（不带图形界面）：

sbatch run_matlab.sbatch, 该程序执行结束后，即可前往项目文件夹执行 fortran 部分

其中，run_matlab.sbatch 的内容为（xxxx 为用户名，后同）：

```
#!/bin/bash

#SBATCH --account=xxxx
#SBATCH --partition=sd530
#SBATCH --nodes=1
#SBATCH --ntasks=1

export MATLAB_PREFDIR=/tmp/xxxx_matlab

cd $SLURM_SUBMIT_DIR

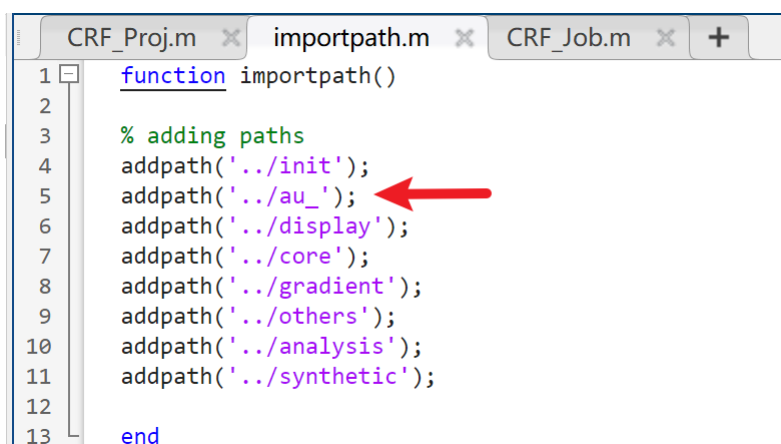
module load matlab/R2021a

matlab -nosplash -nodisplay -nodesktop -r "CRF_Proj;CRF_Job;CRF_McMC"
```

在包含图形界面的 **matlab** 执行（例如个人 PC，或者在计算节点打开 matlab 图形界面）：依次运行 CRF_proj，CRF_job 以及 CRF_McMC

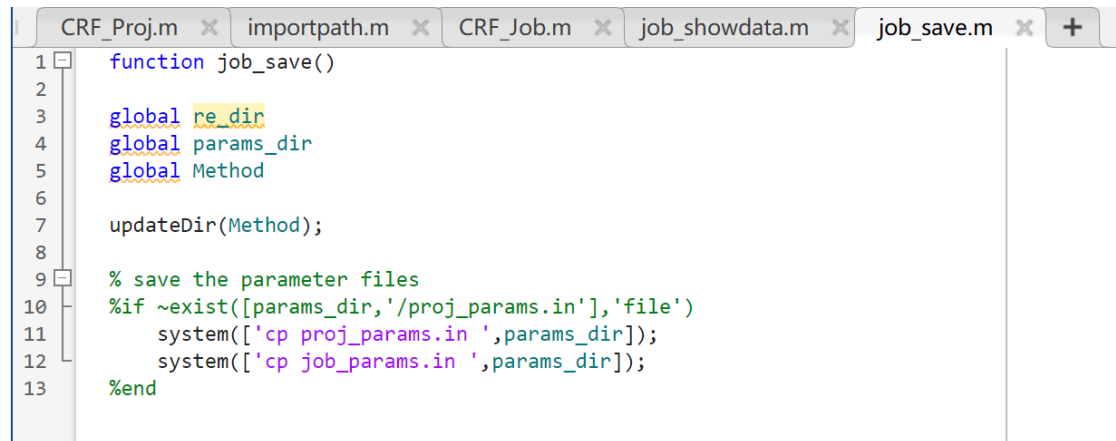
注意：原版程序似乎不是在 windows 下开发的，所以如果需要在 windows 下运行，需要修改几个地方。为了避免潜在的 BUG，可能尽量不要使用 windows 下运行的结果来执行后续 fortran 部分代码，windows 下运行 matlab 代码主要用于检查波形即可。不过，在目前的试验中，还没有出现将 matlab 运行结果上传到服务器后运行出错的情况出现。

(1) 由于 windows 下可能不支持 aux 作为文件夹名称，如果在 windows 下运行 matlab，文件夹 aux 可能被重命名为 au_，请调整 importpath.m 中的对应内容，将 aux 改为 au_



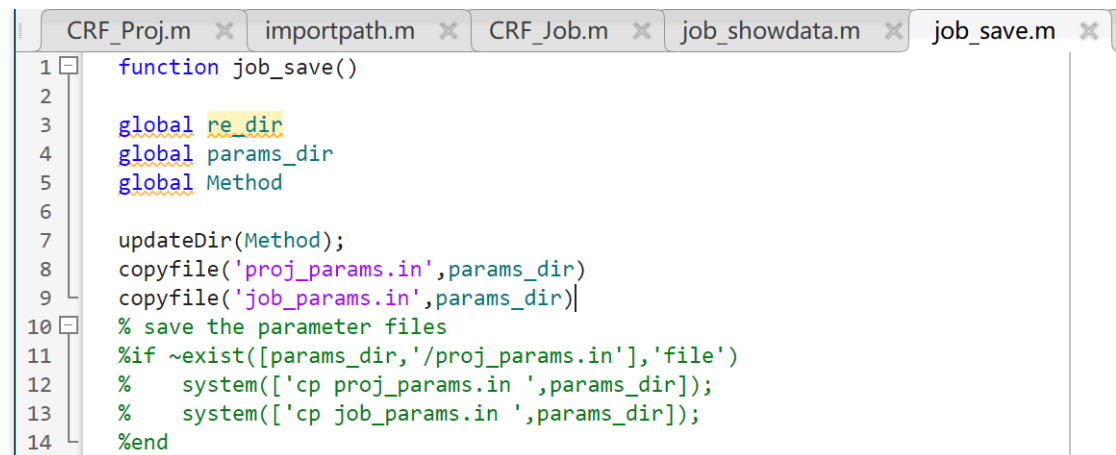
(2) 由于 windows 下没有 cp 命令，所以需要将 init/job_save.m 中复制文件的命令修改为 matlab 内置的命令：

原版：



```
1 function job_save()
2
3 global re_dir
4 global params_dir
5 global Method
6
7 updateDir(Method);
8
9 % save the parameter files
10 %if ~exist([params_dir,'/proj_params.in'],'file')
11     system(['cp proj_params.in ',params_dir]);
12     system(['cp job_params.in ',params_dir]);
13 %end
```

修改版：

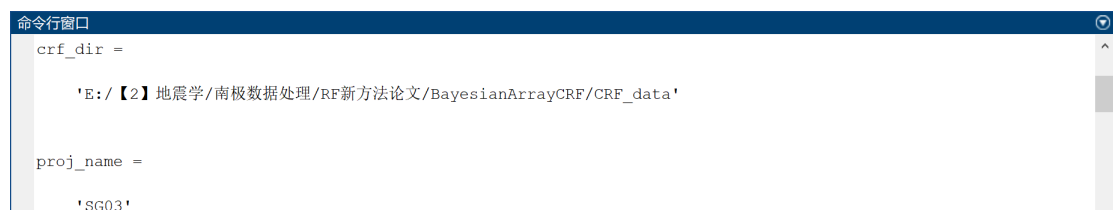


```
1 function job_save()
2
3 global re_dir
4 global params_dir
5 global Method
6
7 updateDir(Method);
8 copyfile('proj_params.in',params_dir)
9 copyfile('job_params.in',params_dir)
10 % save the parameter files
11 %if ~exist([params_dir,'/proj_params.in'],'file')
12     % system(['cp proj_params.in ',params_dir]);
13     % system(['cp job_params.in ',params_dir]);
14 %end
```

(1) CRF_proj:

需要已经设置好的 proj_params.in

输出：



```
命令行窗口
crf_dir =
    'E:/【2】地震学/南极数据处理/RF新方法论文/BayesianArrayCRF/CRF_data'

proj_name =
    'SG03'
```

...

```

rfile =
    'SG3.NAME.BHR'

eventlist =
    3×1 cell 数组
    {'20190512_1924'}
    {'20190526_0741'}
    {'20190530_0903'}

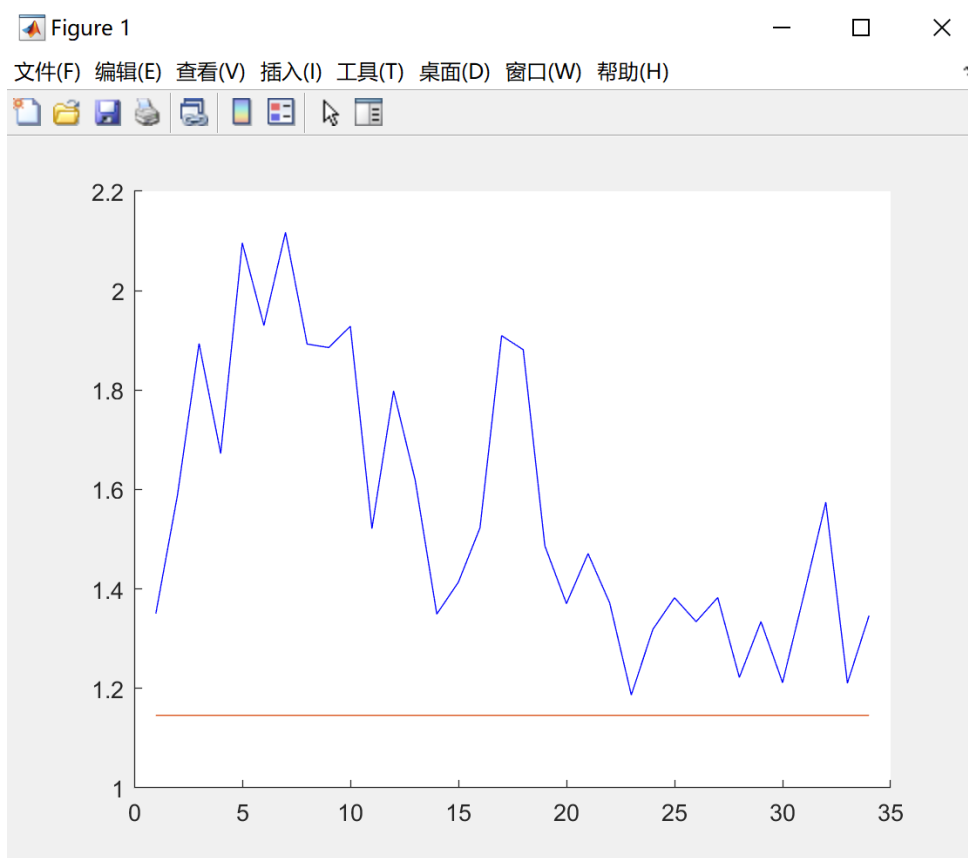
```

(2) **CRF_job:** (Figure 700+可用于异常地震波形检查，以供后续人为剔除异常数据)

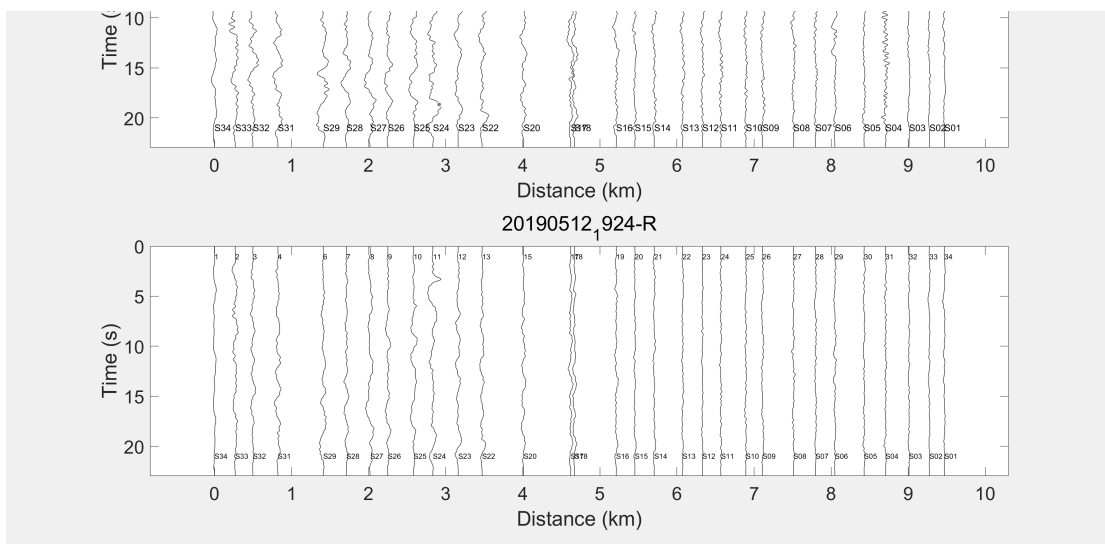
需要 job_params.in

输出:

1) **Figure 1:** 由函数 quality_control 调用的 amplitudeChecking_Z 绘制，横坐标为台站编号，纵坐标为标准差检查的结果。目前看起来该图是自动生成的，并不需要人为干预（因为每个地震事件都会生成一个这样的图，循环至下一事件时会清除该图重新绘制）



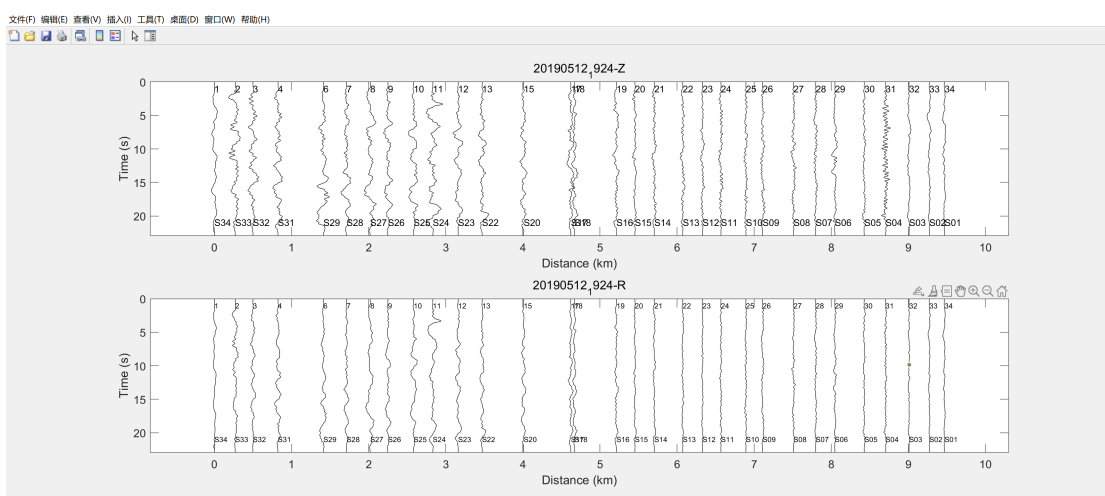
2) **Figure 600 (及以上):** 展示了 Z 分量及 R 分量的噪声波形数据
如果看起来图是这样的:



可以注释掉 `init/job_showdata` 中设置图窗位置的语句，也可以微调字体大小

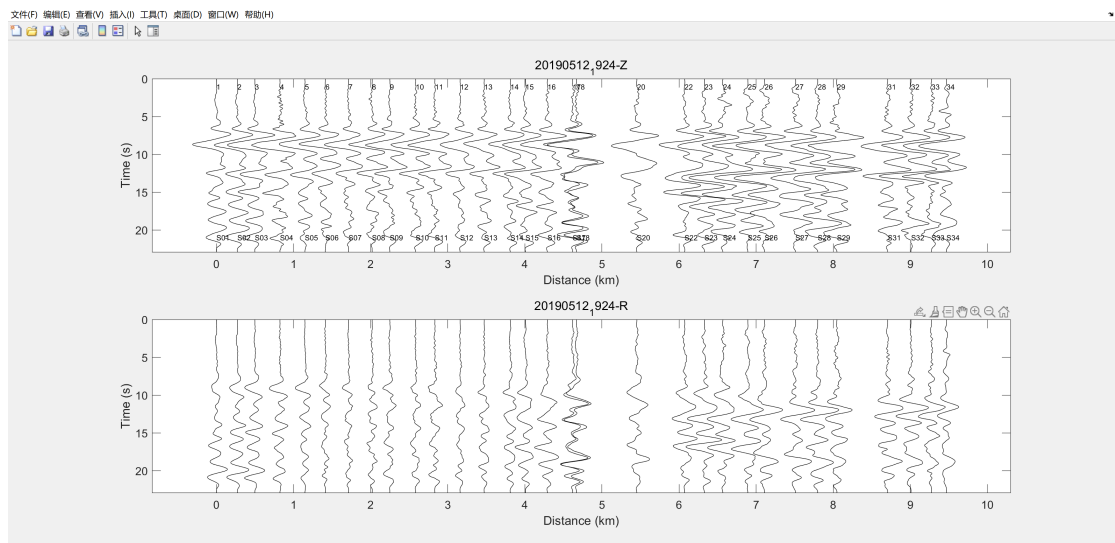
```
CRF_Proj.m  importpath.m  CRF_Job.m  job_showdata.m  +
8  global t_r
9  global evidlist
10 global rela_dist
11 global sta_flag
12
13 global newstnames
14 global scrsz
15 global DataZ_N
16 global DataR_N
17
18 %% show the noise
19 coe=waveform_scaling();
20 for iev=1:nsv
21     figure(600+iev);
22     clf;
23     %set(600+iev,'Position',[100 300 scrsz(3)/1.2 scrsz(4)/1.2]);
24     subplot(2,1,1);
25     hold on;
26     box on;
27     for ista=1:nsta
28         if sta_flag(iev,ista)==1
29             plot(DataZ_N(:,ista,iev)*coe+rela_dist(ista),t_z,'k');
30             text(rela_dist(ista),t_z(1)+1,num2str(ista),'fontsize',10);
31             text(rela_dist(ista),t_z(end)-2,newstnames{ista},'fontsize',10);
32         end
33     end
34 end
```

调整后的结果如下：



3) **Figure 700（及以上）**：展示了 Z 分量及 R 分量的事件波形数据，可以人为地检查（并手动剔除）可能存在的异常波形

类似地，如果图像显示有问题，可以注释掉 `set(700+iev,'Position',...)`



- 4) 如果 matlab 提示 “'cp' 不是内部或外部命令，也不是可运行的程序或批处理文件”，请按照本节开头的提示，修改 `init/job_save.m`。`job_save.m` 运行成功后，会将两个参数设置文件（*.in）复制到项目的 Params 文件夹（在 matlab 命令行输入 `global params_dir` 即可查看 Params 文件路径）

(3) CRF_McMC

直接运行即可。成功运行后，在项目的 Results 文件夹下（例如，`BayesianArrayCRF/CRF_data/SG03/NA/Results/209`），将生成用于计算 CRF 的资料，如下图所示：

RF新方法论文 > BayesianArrayCRF > CRF_data > SG03 > NA > Results > 209

在 209 中搜索

名称	修改日期	类型	大小
sub-15	2023/8/9 21:55	文件夹	
sub-16	2023/8/9 21:55	文件夹	
sub-17	2023/8/9 21:55	文件夹	
sub-18	2023/8/9 21:55	文件夹	
sub-19	2023/8/9 21:55	文件夹	
sub-20	2023/8/9 21:55	文件夹	
sub-21	2023/8/9 21:55	文件夹	
sub-22	2023/8/9 21:55	文件夹	
sub-23	2023/8/9 21:55	文件夹	
sub-24	2023/8/9 21:55	文件夹	
sub-25	2023/8/9 21:55	文件夹	
sub-26	2023/8/9 21:55	文件夹	
sub-27	2023/8/9 21:55	文件夹	
sub-28	2023/8/9 21:55	文件夹	
sub-29	2023/8/9 21:55	文件夹	
sub-30	2023/8/9 21:55	文件夹	
distALL.info	2023/8/9 21:55	INFO 文件	1 KB
ff.bin	2023/8/9 21:55	BIN 文件	4 KB
ff_sym.bin	2023/8/9 21:55	BIN 文件	4 KB
subArray.info	2023/8/9 21:55	INFO 文件	1 KB
win_map.mat	2023/8/9 21:55	MATLAB Data	1 KB
Ya.bin	2023/8/9 21:55	BIN 文件	4 KB
Ya_sym.bin	2023/8/9 21:55	BIN 文件	4 KB

执行 fortran 部分代码 (rjMCMCinv)

本步骤需要做超计算机上操作。首选需要在超计算机上运行的 matlab 部分，然后转至项目 Result 文件夹下提交作业。。

(1) 提交作业

提供了批量提交作业脚脚本。通过执行 `sh control_sbatch.sh` 即可批量执行。其中 `control_sbatch.sh` 的内容如下：

```
#!/bin/bash

for n in `ls | grep sub-`
do
echo $n

echo ""#!/bin/bash
#SBATCH --account=xxxx
#SBATCH --partition=sd530
#SBATCH --ntasks=30

module load openmpi/4.0.5_OPA_gcc9.3
"" > submit_one_job.sbatch

echo 'cd $SLURM_SUBMIT_DIR' >> submit_one_job.sbatch
```

```

echo ""cp *.bin *.info ./${n}/data/

cd ./${n}/
mpirun -np 30 rjMCMCinv.o"" >> submit_one_job.sbatch

sbatch submit_one_job.sbatch

done

```

请注意，在导入模块时，openmpi 的版本尽量与编译时的 openmpi 版本保持一致。另外，根据王新老师的建议，mpirun -np 100 rjMCMCinv.o 或许是比较合适的选择（相应地，--ntasks 可能也需要改成 100），但 -np 30 也可以跑出一个还行的结果。

运行结束后，可以观察程序输出（利用 tail *out）：

```

-bash-4.2$ ls
209.zip          slurm-4407517.out  slurm-4407529.out  sub-10  sub-21  sub-5
control_sbatch.sh slurm-4407518.out  slurm-4407530.out  sub-11  sub-22  sub-6
distALL.info      slurm-4407519.out  slurm-4407531.out  sub-12  sub-23  sub-7
ff.bin            slurm-4407520.out  slurm-4407532.out  sub-13  sub-24  sub-8
ff_sym.bin        slurm-4407521.out  slurm-4407533.out  sub-14  sub-25  sub-9
slurm-4407510.out  slurm-4407522.out  slurm-4407534.out  sub-15  sub-26  subArray.info
slurm-4407511.out  slurm-4407523.out  slurm-4407535.out  sub-16  sub-27  submit_one_job.sbatch
slurm-4407512.out  slurm-4407524.out  slurm-4407536.out  sub-17  sub-28  win_map.mat
slurm-4407513.out  slurm-4407525.out  slurm-4407537.out  sub-18  sub-29  Ya.bin
slurm-4407514.out  slurm-4407526.out  slurm-4407538.out  sub-19  sub-3   Ya_sym.bin
slurm-4407515.out  slurm-4407527.out  slurm-4407539.out  sub-2   sub-30
slurm-4407516.out  slurm-4407528.out  sub-1              sub-20  sub-4

```

（执行 tail *out）

```

==> slurm-4407538.out <==
sample:      10000 /      10000
number of phase:      15 like:  74738.871023507876
*****
sample:      10000 /      10000
number of phase:      15 like:  67467.573833993956
*****
sample:      10000 /      10000
number of phase:      15 like:  59510.159497485460
*****
time taken by the code was  277.493652      seconds

==> slurm-4407539.out <==
sample:      10000 /      10000
number of phase:      15 like:  58008.523135027404
*****
sample:      10000 /      10000
number of phase:      15 like:  53090.473895477873
*****
sample:      10000 /      10000
number of phase:      15 like:  43971.079418397916
*****
time taken by the code was  265.487244      seconds
-bash-4.2$

```

以上的程序输出意味着程序没有由于 runtime error 停止。

接下来可以进入一个 subarray，检查接收函数是否正常输出：

```

-bash-4.2$ cd sub-z
-bash-4.2$ ls
average.out      data            mcmcLayer.out  posterigor.out
bestmodel.out   mcmcHis.out    params.in      results
-bash-4.2$

```

执行 `genCRF average.out ./data/WaveRZ.xyz 0.05 rf_plot.info`

```

-bash-4.2$ ls
average.out      RF12_rf2std.sac  RF15_R.sac      RF4_rf2std.sac  RF7_R.sac
bestmodel.out   RF12_Rrf1std.sac RF15_Z.sac      RF4_Rrf1std.sac RF7_Z.sac
data            RF12_Rrf2std.sac RF1_rf1std.sac  RF4_Rrf2std.sac RF8_rf1std.sac
mcmcHis.out     RF12_R.sac       RF1_rf2std.sac  RF4_R.sac       RF8_rf2std.sac
mcmcLayer.out   RF12_Z.sac       RF1_Rrf1std.sac RF4_Z.sac       RF8_Rrf1std.sac
params.in       RF13_rf1std.sac  RF1_Rrf2std.sac RF5_rf1std.sac  RF8_Rrf2std.sac
posterigor.out  RF13_rf2std.sac  RF1_R.sac       RF5_rf2std.sac  RF8_R.sac
results         RF13_Rrf1std.sac RF1_Z.sac       RF5_Rrf1std.sac RF8_Z.sac
RF10_rf1std.sac RF13_Rrf2std.sac RF2_rf1std.sac  RF5_Rrf2std.sac RF9_rf1std.sac
RF10_rf2std.sac RF13_R.sac       RF2_rf2std.sac  RF5_R.sac       RF9_rf2std.sac
RF10_Rrf1std.sac RF13_Z.sac       RF2_Rrf1std.sac RF5_Z.sac       RF9_Rrf1std.sac
RF10_Rrf2std.sac RF14_rf1std.sac  RF2_Rrf2std.sac RF6_rf1std.sac  RF9_Rrf2std.sac
RF10_R.sac      RF14_rf2std.sac  RF2_R.sac       RF6_rf2std.sac  RF9_R.sac
RF10_Z.sac      RF14_Rrf1std.sac RF2_Z.sac       RF6_Rrf1std.sac RF9_Z.sac
RF11_rf1std.sac RF14_Rrf2std.sac RF3_rf1std.sac  RF6_Rrf2std.sac rf_plot.info
RF11_rf2std.sac RF14_R.sac       RF3_rf2std.sac  RF6_R.sac       slope1std.info
RF11_Rrf1std.sac RF14_Z.sac       RF3_Rrf1std.sac RF6_Z.sac       slope2std.info
RF11_Rrf2std.sac RF15_rf1std.sac  RF3_Rrf2std.sac RF7_rf1std.sac
RF11_R.sac      RF15_rf2std.sac  RF3_R.sac       RF7_rf2std.sac
RF11_Z.sac      RF15_Rrf1std.sac RF3_Z.sac       RF7_Rrf1std.sac
RF12_rf1std.sac RF15_Rrf2std.sac RF4_rf1std.sac  RF7_Rrf2std.sac

```

出现以上文件，则说明结果提取成功。

后处理：绘图

(1) 快速提取结果

后续许多绘图代码的运行都依赖于 sac 格式的结果。下面提供了快速执行 genCRF 以获取 sac 结果的代码（run_genCRF_allsub.sh）

```
#!/bin/bash

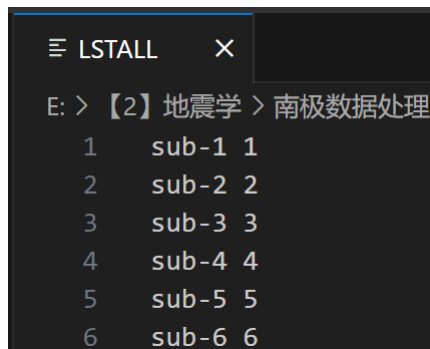
for n in `ls | grep sub-`
do

cd ${n}
genCRF average.out ./data/WaveRZ.xyz 0.05 rf_plot.info
cd ..

done
```

(2) PlotALLrf.csh: 绘制 CRF 接收函数波形，以及 CRF 的斜率信息

运行该代码需要一个名为 LSTALL 的文件，该文件的形式如下



```
E: > 【2】地震学 > 南极数据处理
1    sub-1 1
2    sub-2 2
3    sub-3 3
4    sub-4 4
5    sub-5 5
6    sub-6 6
```

可以执行提供的脚本 get_LSTALL.sh 来获取该文件：

```
#!/bin/bash

for n in `ls | grep sub-`
do

m=`echo ${n} | tr -cd "[0-9]" `
echo ${n} ${m}
echo ${n} ${m} >> LSTALL_temp

done

sort -n -k 2 LSTALL_temp > LSTALL

rm -f LSTALL_temp
```

此外，该代码需要在所有的子文件夹下都已经生成 `rf1std.sac` 文件。如果缺少这些文件，可以使用 `run_genCRF_allsub.sh` 生成这些文件。

请注意，在绘制右图的（下侧绘图代码）时候，绘制线状的斜率信息时，可能使用到了以 `i` 为单位的方式。因此，如果修改了测线距离，可能需要修改相应的与计算单位为 `i` 的 `shift` 有关的代码。

原始绘图代码根据个人所使用的条件不同，可能需要测试及调整，以下是在个人使用 GMT4 绘图时调整后的设置。

主要变动包括：

- 1) 删除 `module load GMT/4.5.16` 相关的部分，因为在武大超算上，没有提供 GMT 模块。直接使用自行安装的 GMT4 即可
- 2) 删除一些关于 GMT 字体相关的设置的内容（自己的环境下没有这些东西），改为用 `gmtset` 手动设置字体大小
- 3) 删除了所有 `gmt` 引导命令，因为考虑到使用的是 GMT4
- 4) 删除了绘制方块的命令

```
#!/bin/csh -f
#####
# Load default GMTSET                                     #
gmtset HEADER_FONT_SIZE 16
gmtset LABEL_FONT_SIZE 14
gmtset ANOT_FONT_SIZE 12
#####
# EQ ID, and the output will be ID-map.ps                 #
set PS = CRFALLrf.ps
#####
psbasemap -JX2.3i/6.0i -R-0.2/5.0/0/9.5 -Ba0 -K -P -X0.5i -Y1.5i >
${PS}
psbasemap -JX -R -Ba1f0.2:"Time (sec)"/a2f1:"Distance (km)"/:. "95%
Confidence level":SE -K -O >> ${PS}
cat LSTALL | awk '{print $2}' > dir_lst
foreach line (`cat dir_lst`)
set subd = `echo $line`
set dist = `cat distALL.info | awk '{if($3=='${subd}') print $1}'`
set mid = `cat ./sub-${subd}/rf_plot.info | grep Midpoint | awk '{print
$2}'`
echo ./sub-${subd}/RF${mid}_rf1std.sac 0 $dist | pssac2 -J -R -Ent-3 -
G255/0/0/0/-0.2/5.0 -g0/0/255/0/-0.2/5.0 -M5/0 -W0.2p -K -O >> ${PS}
end
```

```

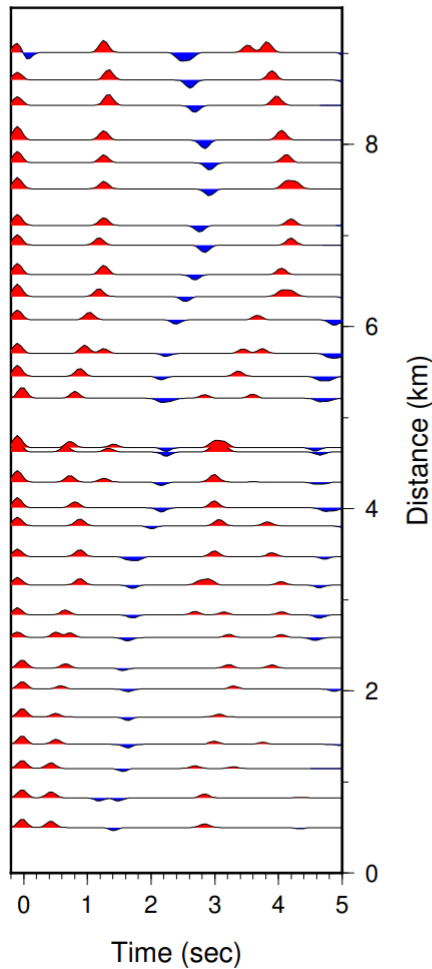
psbasemap -JX2.3i/6.0i -R-0.2/5.0/0/9.5 -Ba0 -K -P -X3.5i -O >> ${PS}
psbasemap -JX -R -Ba1f0.2:"Time (sec)":/a2f1:"Distance (km)":::"95%
Confidence level":SE -K -O >> ${PS}
set scale = `echo 6 9.5 0.5 | awk '{print $3*$1/$2}'`
foreach line (`cat dir_lst`)
set subd = `echo $line`
set dist = `cat distALL.info | awk '{if($3=='${subd}') print $1}'`
set shift = `echo 6 9.5 $dist | awk '{print ($3-0.25)*$1/$2}'`
psxy -T -JX2.3i/${scale}i -R-0.2/5.0/-0.25/0.25 -Y${shift}i -K -O >>
${PS}
cat ./sub-${subd}/slope1std.info | awk '{print $2,$1}' | psxy -J -R -
W0.5p/black -m -K -O >> ${PS}
psxy -T -J -R -Y-${shift}i -K -O >> ${PS}
end

ps2pdf ${PS}
rm -rf ${PS}

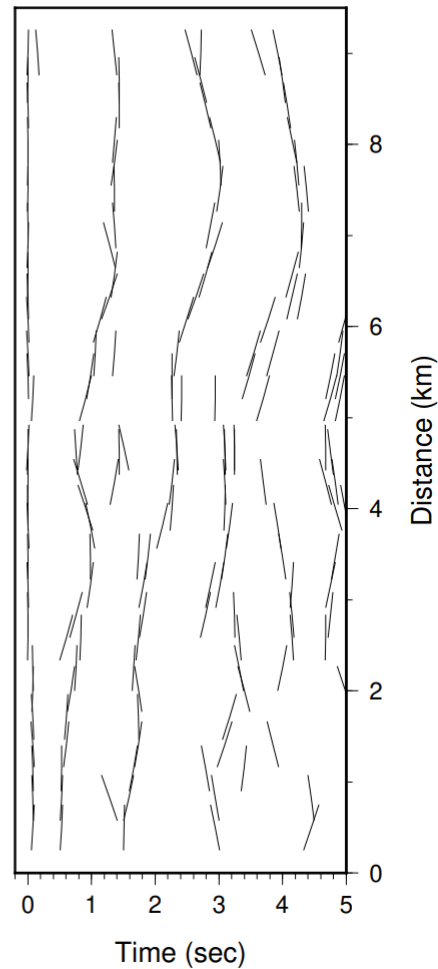
```

绘图效果如下所示：

95% Confidence level



95% Confidence level



请注意，如果使用自己的数据，那么可能需要调整绘图的 range。

(3) plotALLimage.csh: 绘制接收函数剖面

如果运行脚本时，提示以下错误：

```
-bash-4.2$ genCRFstack distALL.info 0.1 0.02 10 10.0 mesh.xyz
./sub-1/sub-1_average.out
At line 9 of file length.f90 (unit = 101, file = 'mesh.xyz')
Fortran runtime error: File './sub-1/sub-1_average.out' does not exist
```

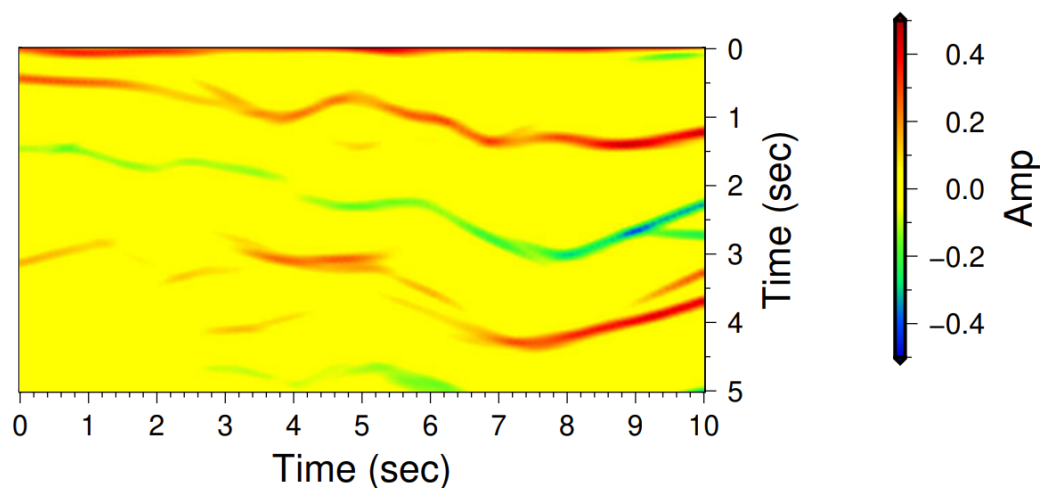
这可能是由于此前 average.out 的名称为 sub-n_average.out 造成的。一种可能的解决方法是，将 genCRFstack.f90 中第 74 行的 tmpfile，由原本的：

```
tmpfile = "./sub-1/TRIM(nfile)"/"sub-1/TRIM(nfile)"/"_average.out"
```

修改为：

```
tmpfile = "./sub-1/TRIM(nfile)"/"sub-1/TRIM(nfile)"/"average.out"
```

此后即可正常执行，运行结果为：



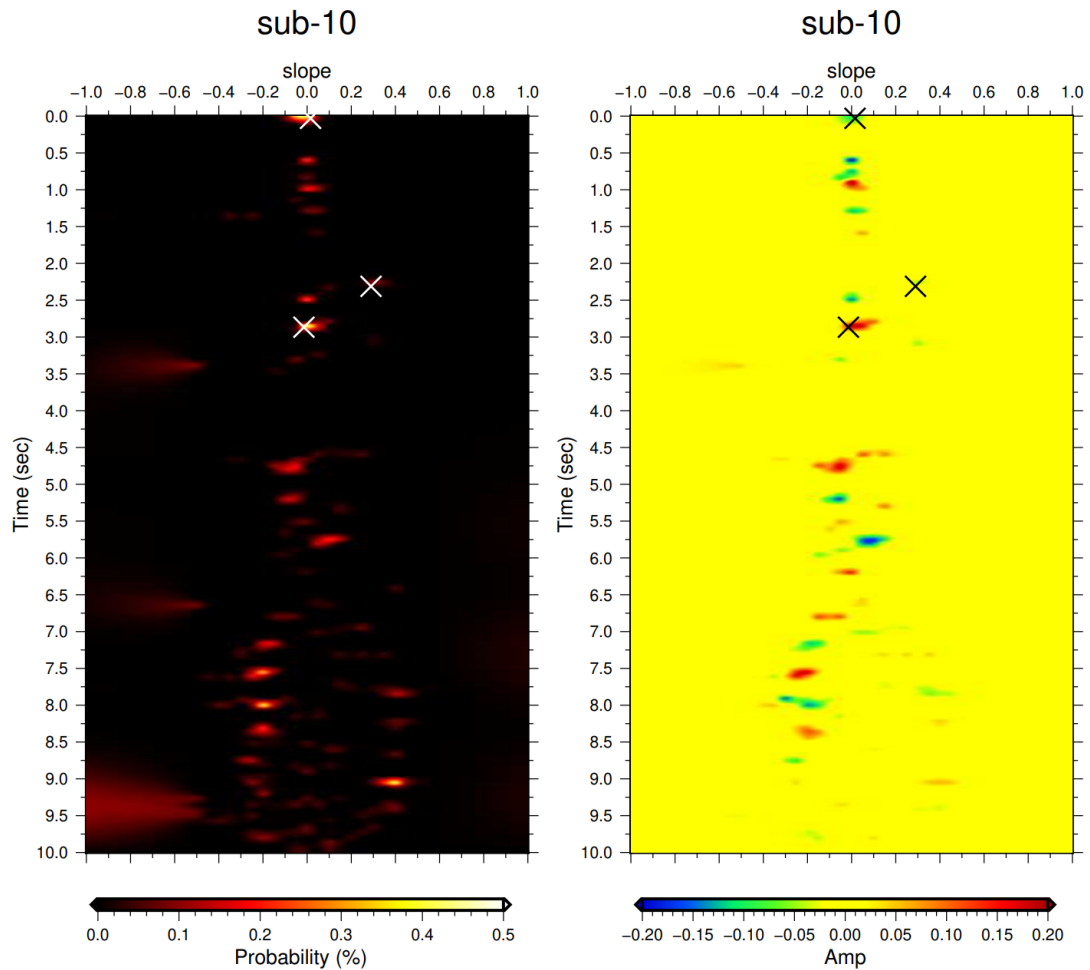
(注意，横坐标单位可能不是 Time (sec)而是 Distance(km)，需要修改)

对应的绘图代码修改为：

```
#!/bin/csh -f
#####
# Load default GMTSET
#####
set PS = CRFimage.ps
#####
genCRFstack distALL.info 0.1 0.02 10 10.0 mesh.xyz
gmt psbasemap -JX4.0i/-2.0i -R0.0/10.0/0/5.0 -Ba0 -K -P -X0.5i -Y1.5i >
${PS}
gmt psbasemap -JX -R -Ba1f0.2:"Time (sec)":/a1f0.5:"Time (sec)":SE -K
-O >> ${PS}
#cat distALL.info | awk '{print -0.3,$1}' | gmt psxy -J -R -Ss0.05i -
Ggray -W0.5p -N -K -O >> ${PS}
gmt makecpt -D -Cseis -I -T-0.5/0.5/0.01 -Z > amp.cpt
cat mesh.xyz | awk '{print $2+0.25,$1,$3}' | gmt surface -R -I0.05/0.05
-G1sigma.grd
gmt grdsample 1sigma.grd -G1sigmaI.grd -I0.01
gmt grdimage 1sigmaI.grd -J -R -Camp.cpt -K -O >> ${PS}
gmt psscale -Camp.cpt -D13/0.5+w5.0/0.15+e -Ba0.2f0.1:"Amp": -K -O >>
${PS}

ps2pdf ${PS}
rm -rf ${PS}
```

(4) plotMCMC.csh: 绘制 MCMC 输出的后验概率相关的信息



对应的代码修改为:

```
#!/bin/csh -f
#####
# EQ ID, and the output will be ID-map.ps  #
gmt gmtset FONT_ANNOT 8p
gmt gmtset FONT_LABEL 10p
gmt gmtset FONT_TITLE 16p
gmt gmtset MAP_ANNOT_OFFSET 4p
gmt gmtset MAP_LABEL_OFFSET 4p

set name = $1
set PS = CRFpdf.ps
#####
gmt psbasemap -JX3i/-5i -R-1.0/1.0/0/10.0 -
Ba0.2f0.1:"slope":/a0.5f0.25:"Time (sec)":NsWe:."$name": -K -P -Y5.5i >
${PS}
gmt makecpt -D -Chot -T0/0.5/0.01 -Z > pdf.cpt
set max = `cat posterigor.out | awk '{if($1>0.2) print $4}' | sort -g |
tail -1`
```

```

set min = `cat posterigor.out | awk '{if($1>0.2) print $4}' | sort -g |
head -1`
cat posterigor.out | awk '{print $2,$1,($4-`${min}`)/(`${max}`-
`${min}`)}' > PDF.norm
cat PDF.norm | gmt blockmean -R-1.0/1.0/0/10.0 -I0.05/0.05 | gmt
surface -Ll0.001 -Lu0.5 -R -I0.05/0.05 -GPDF.grd
gmt grdsample PDF.grd -GPDF_i.grd -I0.01
gmt grdimage PDF_i.grd -J -R -Cpdf.cpt -K -O >> ${PS}
cat bestmodel.out | awk '{if(NR>1) print $2,$1}' | gmt psxy -J -R -
Sx0.2i -W1p,white -Gwhite -K -O -N >> ${PS}
gmt psscale -Cpdf.cpt -Dx0.2/-1.0+w7.0/0.2+h+e -Ba0.1f0.02:"Probability
(%)" -K -O >> ${PS}

gmt psbasemap -JX3i/-5i -R-1.0/1.0/0/10.0 -
Ba0.2f0.1:"slope":/a0.5f0.25:"Time (sec)":NsWe:."$name": -K -O -
X3.7i >> ${PS}
gmt makecpt -D -Cseis -T-0.2/0.2/0.01 -Z -I > pdf.cpt
cat posterigor.out | awk '{print $2,$1,$3}' > PDF.norm
cat PDF.norm | gmt blockmean -R-1.0/1.0/0/10.0 -I0.05/0.05 | gmt
surface -Ll-0.2 -Lu0.2 -R -I0.05/0.05 -GPDF.grd
gmt grdsample PDF.grd -GPDF_i.grd -I0.01
gmt grdimage PDF_i.grd -J -R -Cpdf.cpt -K -O >> ${PS}
cat bestmodel.out | awk '{if(NR>1) print $2,$1}' | gmt psxy -J -R -
Sx0.2i -W1p,black -Gblack -K -O -N >> ${PS}
gmt psscale -Cpdf.cpt -Dx0.2/-1.0+w7.0/0.2+h+e -Ba0.05f0.01:"Amp": -K -
O >> ${PS}

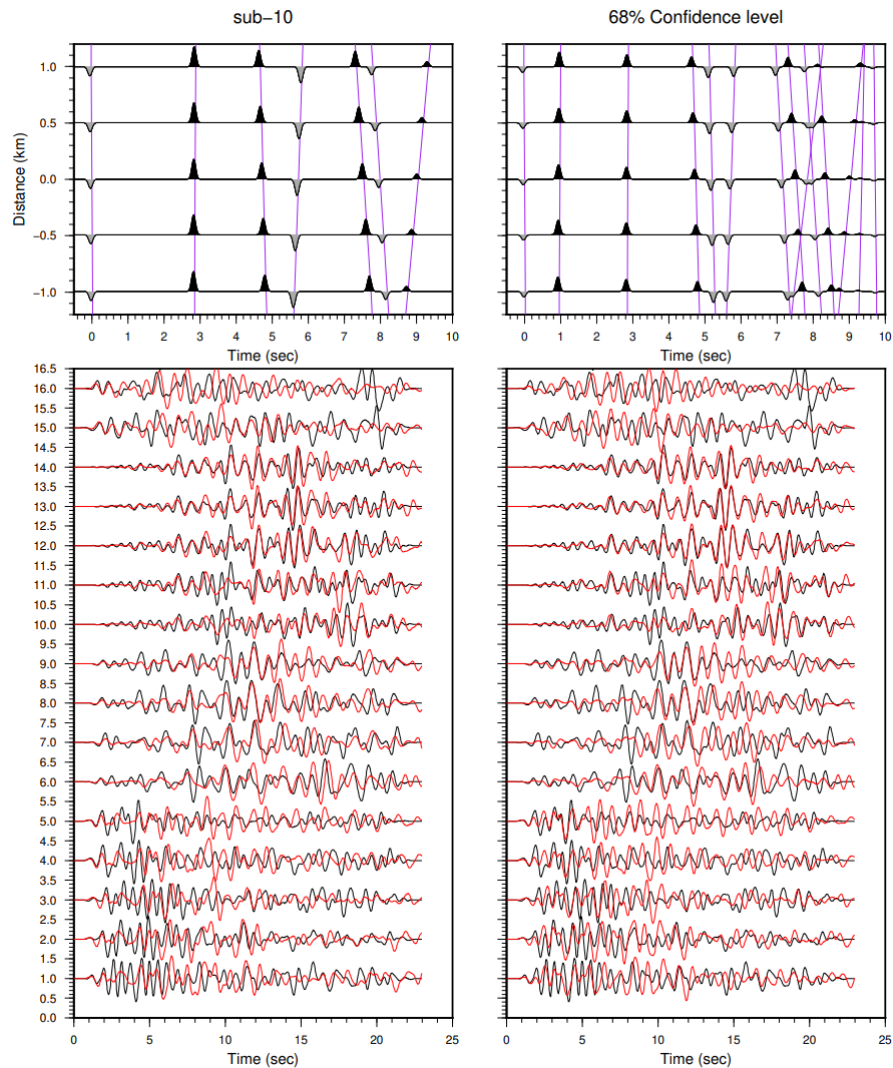
ps2pdf ${PS}
rm -rf ${PS}

```

(5) plotResult.csh: 绘制接收函数结果、原始波形以及对波形的拟合情况

该代码在绘图时可能需要调整绘图的范围。例如，台间距不同的话，上面两图的 Distance 的最佳距离也可能会不一样。而如果出现 pssac2: No points to plot – RFxx_R.sac 的话，则可能是由于下面两图的绘图范围设置小了，导致有些原始地震记录绘制不上。

（由于上两图绘制结果中，可能包含多个地震处理出来的接收函数的叠放的结果，因此当地震较多时，有可能出现正负涂色些许错位等等的问题）



修改的代码为

```
#!/bin/csh -f
#####
# EQ ID, and the output will be ID-map.ps    #
gmtset PAPER_MEDIA A4
gmtset HEADER_FONT_SIZE 12
gmtset LABEL_FONT_SIZE 10
gmtset ANOT_FONT_SIZE 8
gmtset LABEL_OFFSET 0p

set PS = CRFresult.ps
set name = $1
genCRF average.out ./data/WaveRZ.xyz 0.05 rf_plot.info
#####
psbasemap -JX3.5i/2.5i -R-0.5/10.0/-1.2/1.2 -Ba0 -K -P -X0.6i -Y8.5i >
${PS}
psbasemap -JX -R -Ba1f0.2:"Time (sec)":/a0.5f0.1:"Distance
(km)":SW:."$name": -K -O >> ${PS}
```

```

cat slope2std.info | awk '{print $2,$1}' | psxy -J -R -W0.5p/purple -m
-K -O >> ${PS}
cat rf_plot.info | awk '{if(NR>3) print $2"_rf2std.sac",0,$1}' | pssac2
-J -R -Ent-3 -G0/0/0/0/-0.5/10.0 -g170/170/170/0/-0.5/10.0 -C-0.5/10.0
-M3/0 -W0.5p -K -O >> ${PS}

psbasemap -JX3.5i/2.5i -R-0.5/10.0/-1.2/1.2 -Ba0 -K -O -X4.0i >> ${PS}
psbasemap -JX -R -Ba1f0.2:"Time (sec)":/a0.5f0.1:"Distance (km)":::"68%
Confidence level":Sw -K -O >> ${PS}
cat slope1std.info | awk '{print $2,$1}' | psxy -J -R -W0.5p/purple -m
-K -O >> ${PS}
cat rf_plot.info | awk '{if(NR>3) print $2"_rf1std.sac",0,$1}' | pssac2
-J -R -Ent-3 -G0/0/0/0/-0.5/10.0 -g170/170/170/0/-0.5/10.0 -C-0.5/10.0
-M5/0 -W0.5p -K -O >> ${PS}
##### Waveform fits

psbasemap -JX3.5i/6i -R0.0/25.0/0/16.5 -Ba0 -K -O -Y-6.5i -X-4.0i >>
${PS}
psbasemap -JX -R -Ba5f1:"Time (sec)":/a0.5f0.1Sw -K -O >> ${PS}
cat rf_plot.info | awk 'BEGIN{i=1} {if(NR>3) print $2"_R.sac",0,i++}' |
pssac2 -J -R -Ent-3 -M0.4 -W0.5p -K -O >> ${PS}
cat rf_plot.info | awk 'BEGIN{i=1} {if(NR>3) print
$2"_Rrf2std.sac",0,i++}' | pssac2 -J -R -Ent-3 -M0.4 -W0.5p/red -K -
O >> ${PS}

psbasemap -JX3.5i/6i -R0.0/25.0/0/16.5 -Ba0 -K -O -X4.0i >> ${PS}
psbasemap -JX -R -Ba5f1:"Time (sec)":/a0.5f0.1Sw -K -O >> ${PS}
cat rf_plot.info | awk 'BEGIN{i=1} {if(NR>3) print $2"_R.sac",0,i++}' |
pssac2 -J -R -Ent-3 -M0.4 -W0.5p -K -O >> ${PS}
cat rf_plot.info | awk 'BEGIN{i=1} {if(NR>3) print
$2"_Rrf1std.sac",0,i++}' | pssac2 -J -R -Ent-3 -M0.4 -W0.5p/red -K -
O >> ${PS}

ps2pdf ${PS}
rm -rf ${PS}

```

另一个修改较多的版本或可尝试自动确定最佳绘图范围：

```

#!/bin/csh -f
#####
# EQ ID, and the output will be ID-map.ps      #
gmtset PAPER_MEDIA A4
gmtset HEADER_FONT_SIZE 12
gmtset LABEL_FONT_SIZE 10
gmtset ANOT_FONT_SIZE 8

```

```

gmtset LABEL_OFFSET 0p

set PS = CRFresult.ps
set name = $1
genCRF average.out ./data/WaveRZ.xyz 0.05 rf_plot.info
#####Get Plot Range#####
set Range2=`awk 'NR==1 {print $2+1}' rf_plot.info`
set Range1=`awk 'BEGIN{max=-999} END{printf "%.1f",max+0.2} NR==3 {for
(i=1; i<=NF; i++) if (sqrt($i*$i) > max) max = sqrt($i*$i)}'
rf_plot.info`

#####
psbasemap -JX3.5i/2.5i -R-0.5/10.0/-${Range1}/${Range1} -Ba0 -K -P -
X0.6i -Y8.5i > ${PS}
psbasemap -JX -R -Ba1f0.2:"Time (sec)"/a0.2f0.1:"Distance
(km)":SW:."$name": -K -O >> ${PS}
cat slope2std.info | awk '{print $2,$1}' | psxy -J -R -W0.5p/purple -m
-K -O >> ${PS}
cat rf_plot.info | awk '{if(NR>3) print $2"_rf2std.sac",0,$1}' | pssac2
-J -R -Ent-3 -G0/0/0/0/-0.5/10.0 -g170/170/170/0/-0.5/10.0 -C-0.5/10.0
-M3/0 -W0.5p -K -O >> ${PS}

psbasemap -JX3.5i/2.5i -R-0.5/10.0/-${Range1}/${Range1} -Ba0 -K -O -
X4.0i >> ${PS}
psbasemap -JX -R -Ba1f0.2:"Time (sec)"/a0.2f0.1:"Distance (km)":::"68%
Confidence level":Sw -K -O >> ${PS}
cat slope1std.info | awk '{print $2,$1}' | psxy -J -R -W0.5p/purple -m
-K -O >> ${PS}
cat rf_plot.info | awk '{if(NR>3) print $2"_rf1std.sac",0,$1}' | pssac2
-J -R -Ent-3 -G0/0/0/0/-0.5/10.0 -g170/170/170/0/-0.5/10.0 -C-0.5/10.0
-M5/0 -W0.5p -K -O >> ${PS}
##### Waveform fits

psbasemap -JX3.5i/6i -R0.0/25.0/0/${Range2} -Ba0 -K -O -Y-6.5i -X-
4.0i >> ${PS}
psbasemap -JX -R -Ba5f1:"Time (sec)"/a1.0f0.5Sw -K -O >> ${PS}
cat rf_plot.info | awk 'BEGIN{i=1} {if(NR>3) print $2"_R.sac",0,i++}' |
pssac2 -J -R -Ent-3 -M0.4 -W0.5p -K -O >> ${PS}
cat rf_plot.info | awk 'BEGIN{i=1} {if(NR>3) print
$2"_Rrf2std.sac",0,i++}' | pssac2 -J -R -Ent-3 -M0.4 -W0.5p/red -K -
O >> ${PS}

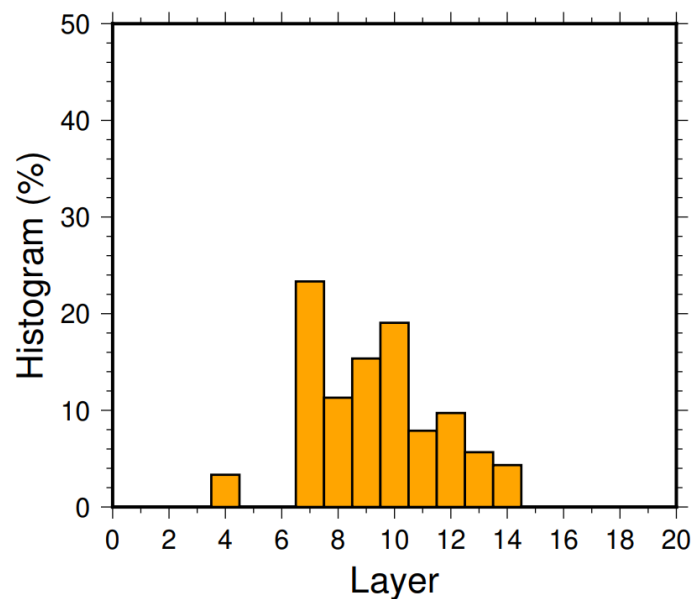
psbasemap -JX3.5i/6i -R0.0/25.0/0/${Range2} -Ba0 -K -O -X4.0i >> ${PS}
psbasemap -JX -R -Ba5f1:"Time (sec)"/a1.0f0.5Sw -K -O >> ${PS}

```

```
cat rf_plot.info | awk 'BEGIN{i=1} {if(NR>3) print $2"_R.sac",0,i++}' |
pssac2 -J -R -Ent-3 -M0.4 -W0.5p -K -O >> ${PS}
cat rf_plot.info | awk 'BEGIN{i=1} {if(NR>3) print
$2"_Rrf1std.sac",0,i++}' | pssac2 -J -R -Ent-3 -M0.4 -W0.5p/red -K -
O >> ${PS}

ps2pdf ${PS}
rm -rf ${PS}
```

(6) plotLayer.csh: 绘制 mcmcLayer.out 的直方图



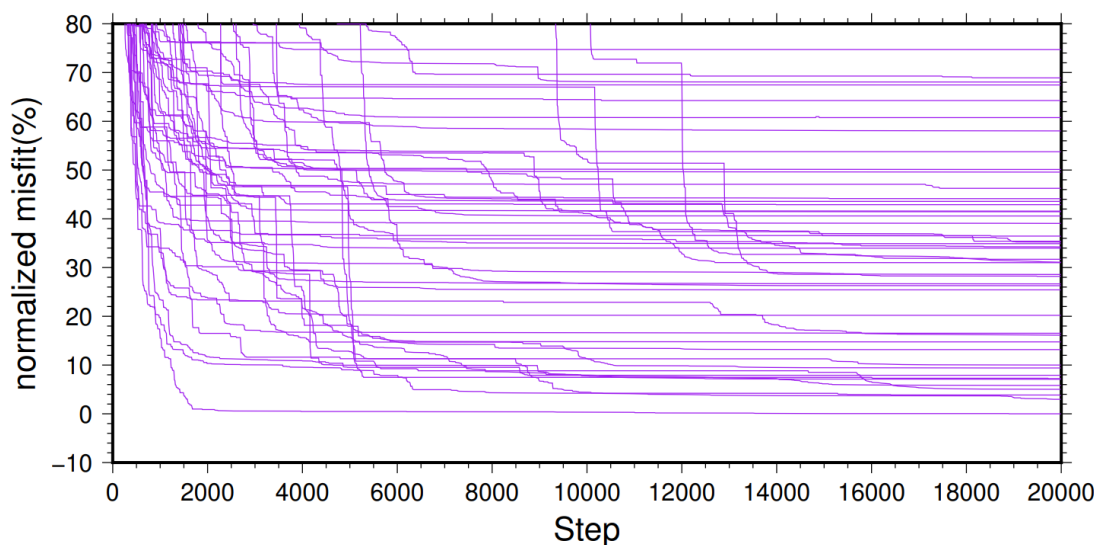
修改后的代码为:

```
#!/bin/csh -f
set PS = CRFLayer.ps

gmt psbasemap -JX3.5i/3i -R0/20/0/50 -Ba2f1:"Layer":/a10f2:"Histogram
(%)" :nSWe -K > ${PS}
cat mcmcLayer.out | awk '{if(NR>=2) print $2}' | gmt pshistogram -J -R
-F -L1p -W1 -Gorange -Z1 -K -O >> ${PS}

ps2pdf ${PS}
rm -rf ${PS}
```

(7) plotStep.csh: 绘制 MCMC 误差收敛情况



其修改后的代码为：

（注意，由于新版代码中 `mcmcStep.out` 改为了 `mcmcHis.out`，因此这里修改了低6行的输入文件名）

```
#!/bin/csh -f

set PS = CRFstep.ps
set BEST = `cat bestmodel.out | head -1 | awk '{print $2}'`
gmt psbasemap -JX6.5i/3i -R0/20000/-10/80 -
Ba2000f500:"Step":/a10f2:"normalized misfit(%)":nSWe -K > ${PS}
cat mcmcHis.out | awk '{print $1,100*($2-'${BEST}')/'${BEST}'}' | gmt
psxy -J -R -W0.5p,purple -K -O >> ${PS}

ps2pdf ${PS}
rm -rf ${PS}
```

其他：

部分需要注意的参数说明（`proj_params.in/job_params.in`）

(1) `proj_params.in`

- 1) `dist_mode` 与 `ref_lat` 和 `ref_lon`: `dist_mode=1` 是指根据经纬度来算台站之间的距离，此时需要给定一个参考的经纬度。对于线性台阵，应选择为线性台阵某一端最开始的台站的经纬度。

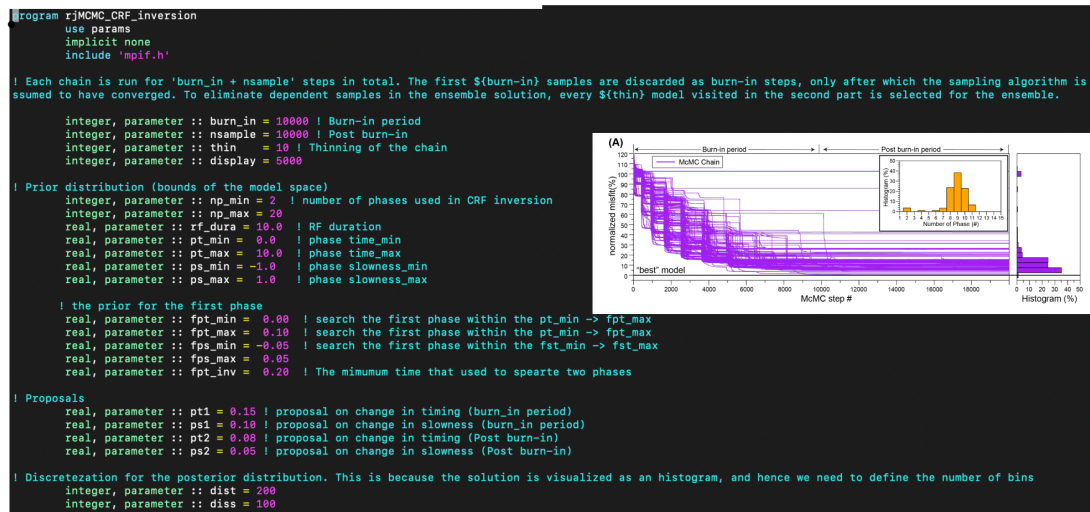
- 2) events: 需要和存放接收函数的文件夹的名字相同
- 3) rfile/zfile: 一定要注意前缀、后缀要跟实际文件名匹配。否则会在 job_loaddata 时出错

(2) job_params.in

- 1) job_num: 该变量是通过数值型存储的，因此如果设置以 0 开头的 job number，实际创建存放 results 的文件夹时，开头的 0 会消失
- 2) marker: 需要和 P 波到时的 sac 头段变量保持一致
- 3) polar: 在 init/job_loaddata.m 中发挥作用。需要设置 polar 为 0 来保证不翻转 R 分量极性（为其他值时，翻转 R 分量极性）。
- 4) retreat: 在 init/job_loaddata.m 中发挥作用。设置为 0 时表示不移动 zero-time
- 5) dre: 决定是否重采样。仅当值为-1 时不重采样。如果设置为其他值，则会执行重采样程序。根据代码来看，例如如果将 dre 设置为 10，则会重采样为 10Hz

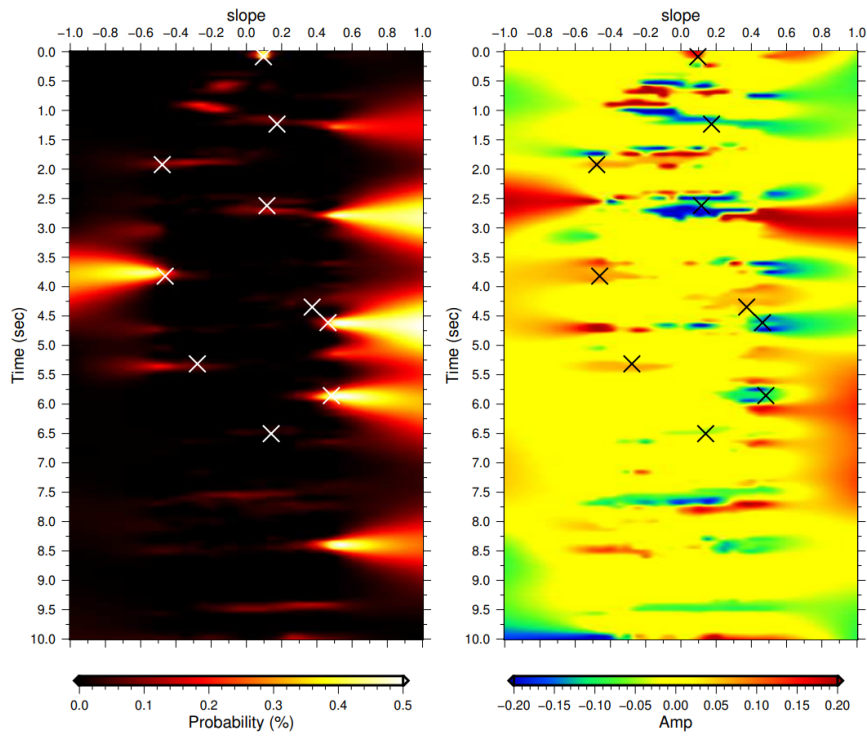
(3) rjMCMCinv.f90: 在

Change the parameters if necessary

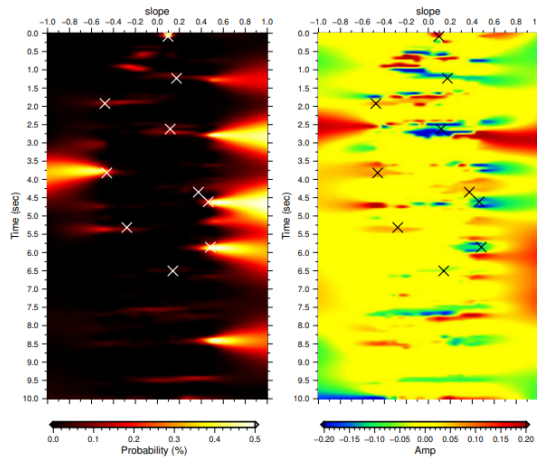


这里参数的影响可以通过前面的绘图脚本来体现，例如：

ps_min 和 ps_max 为-1.0 和 1.0 时，绘图结果和默认的绘图脚本可以呈现出一致的形式。例如如果这二者分别为-0.5 和 0.5 时，用默认的绘图脚本 plotMCMC.csh 绘图，结果可能为：



另外，如果例如 `job_params.in` 中的 `len_rf` 设置为 10.0，但是此处的 `rf_dura` 和 `pt_max` 都设为 15.0 的话，可能会出现这样的情况：

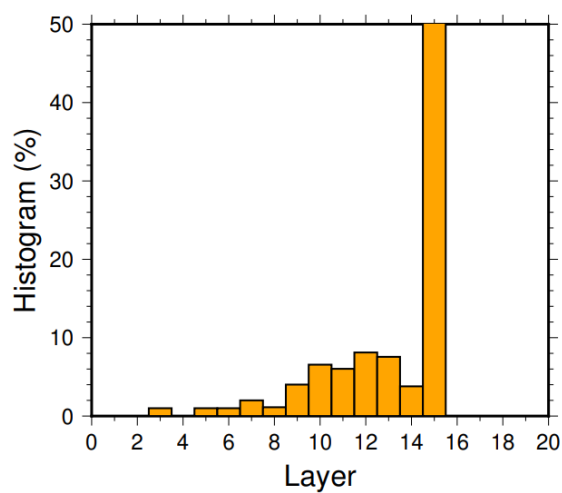


→ × 15s

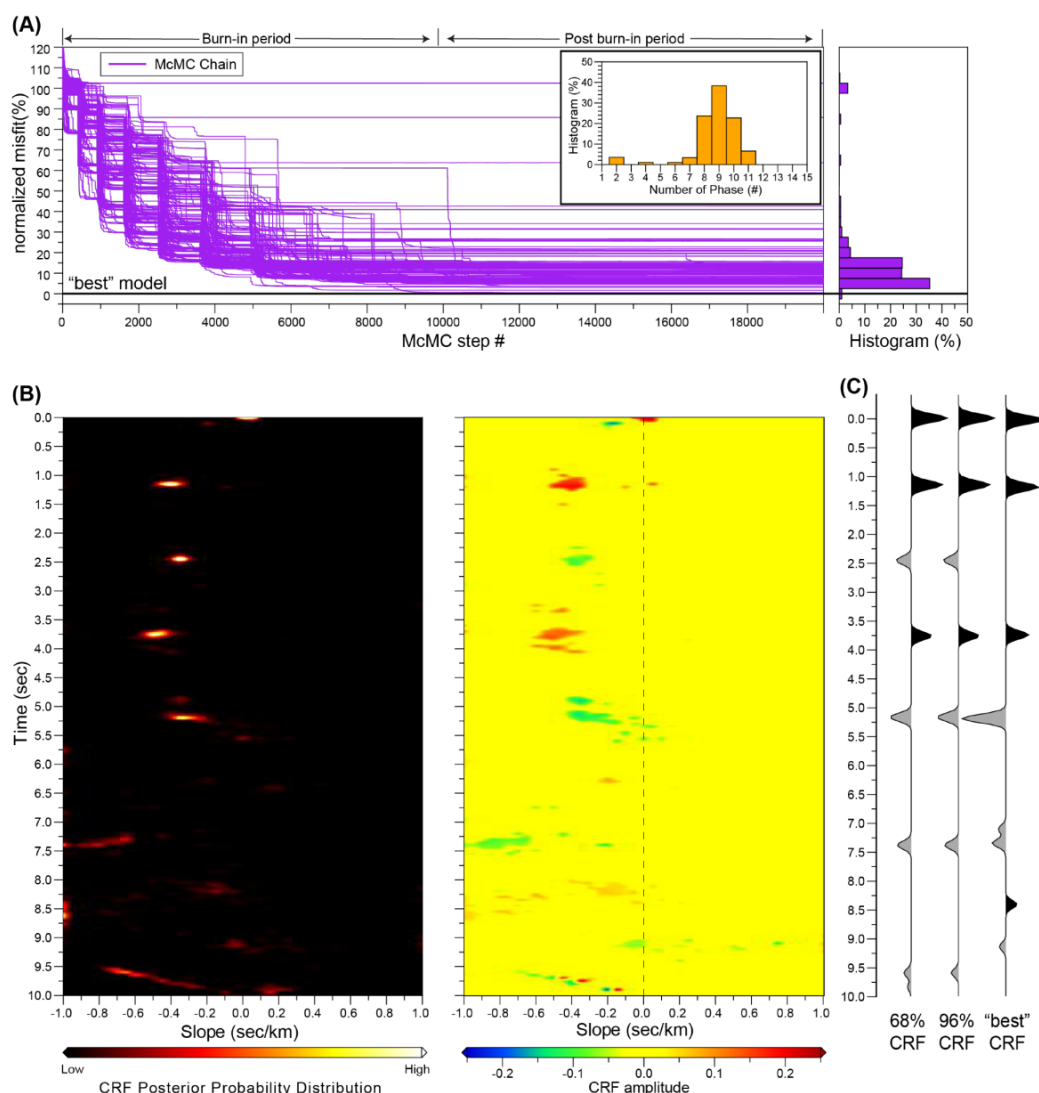
上面的情况可能是由于，`job_params.in` 中的 `len_rf` 会称为子台阵中 `params.in` 中的倒数第三行的内容。`params.in` 会被 `rdparams.f90` 读取，赋值给 Fortran 程序的内部变量 `len_rf`。而在 `forward_modelling.f90` 中，`len_rf` 会用来生成 `t_bound`。一旦不再 `t_bound` 允许的范围内，`misfitval` 会被设置成一个很大的

罚值，进而影响后续的 `likelihood` 的更新（`rjMCMCinv.f90`，第 117 行及 253 行附近）。因此，可能在运行程序前，我们需要考虑到 `len_rf` 和 `rf_dura` 和 `pt_max` 是否匹配。

`np_min` 以及 `np_max` 会影响 `CRFLayer.pdf` 绘图的结果。例如下图是设置 `np_min` 为 2，`np_max` 为 15 时的结果。



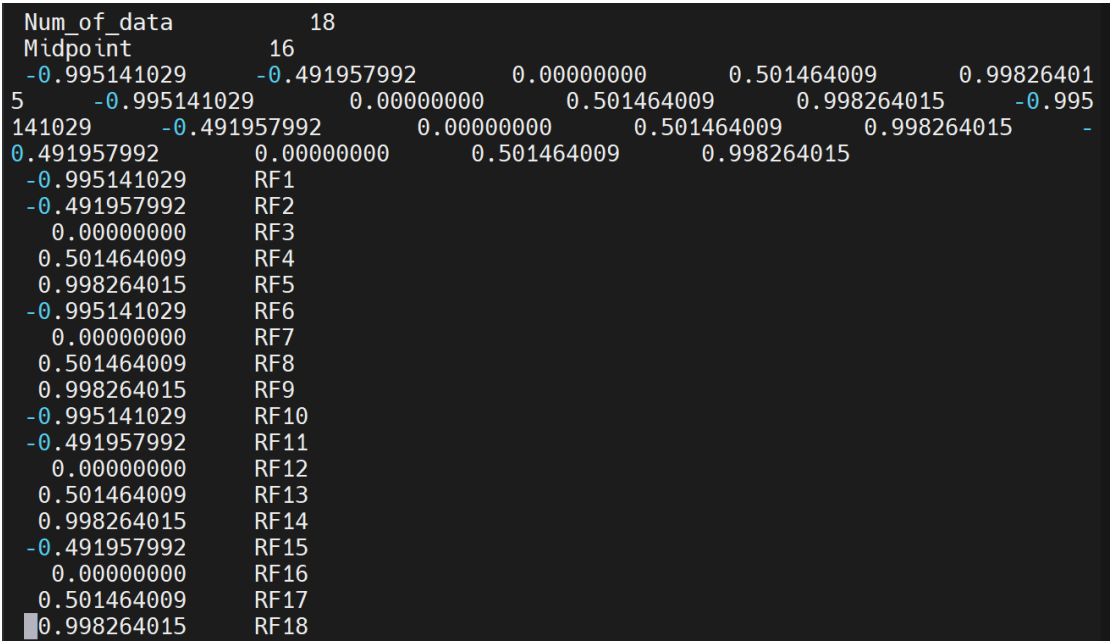
CRF 结果的含义



CRF 考虑沿密集测线子台阵的接收函数应该是缓变的，这种缓变能通过线性变化的趋势刻画。因此，在 CRF 中，会输出两个很重要的结果，其中一个为中心台站的接收函数，还有一个中心台站出，接收函数各震相变化的斜率。注意到接收函数到时为秒（sec），因此震相到时随位置变化的斜率的单位为 sec/km。

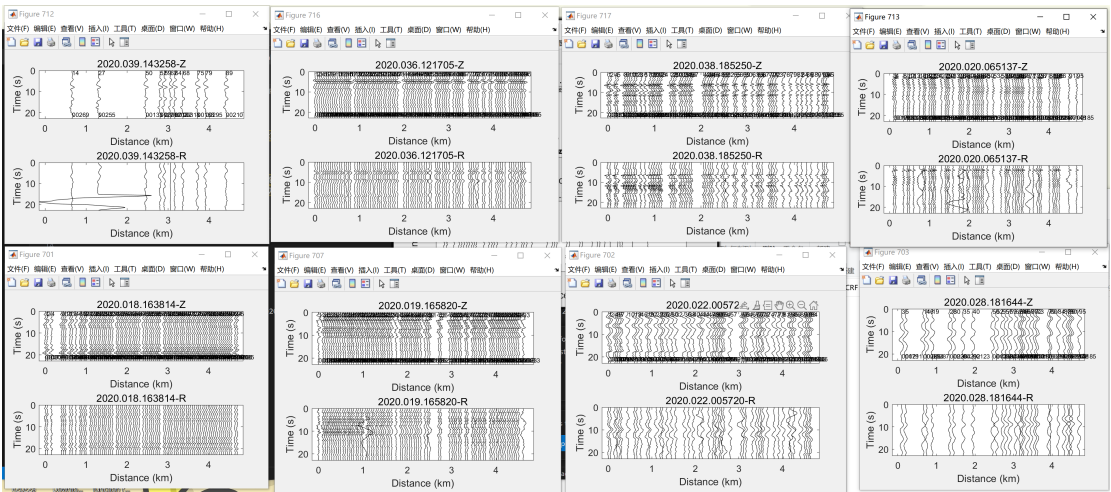
上图中，图 a 表示的是马尔科夫链蒙特卡洛的 misfit 收敛情况。图 b 左图是引入“斜率”域后的接收函数的概率分布图。图中热度颜色表示震相出现的概率。概率对应的图中纵坐标，则为中心台站处的接收函数到时；概率对应的横坐标，则对应该震相的空间线性变化趋势（即，斜率）。图 b 右图是在震相“到时-斜率”域中对应的震相振幅。图 c 则是中心台站接收函数的直观展现（由于只考虑中心台站，因此去除了斜率域的信息）。

在文件“rf_plot.info”中，可以找到中心台站处接收函数的信息，如下图所示。
0.0 对应的接收函数即为中心台站接收函数。这里之所以出现多个 0.0，可能是因为有多个地震。



准备原始数据时需要注意的事情

请务必人工仔细检查原始数据之后，再运行程序。而不要像利用布设时段很长的台站计算传统接收函数时的思路一样，对原始数据仅做一些较为简单的自动化的质量控制，而主要对计算出的接收函数进行较多的人工质量控制。



例如，上图是在 matlab 程序的第二步执行时，输出的地震事件波形，我们可以看到：

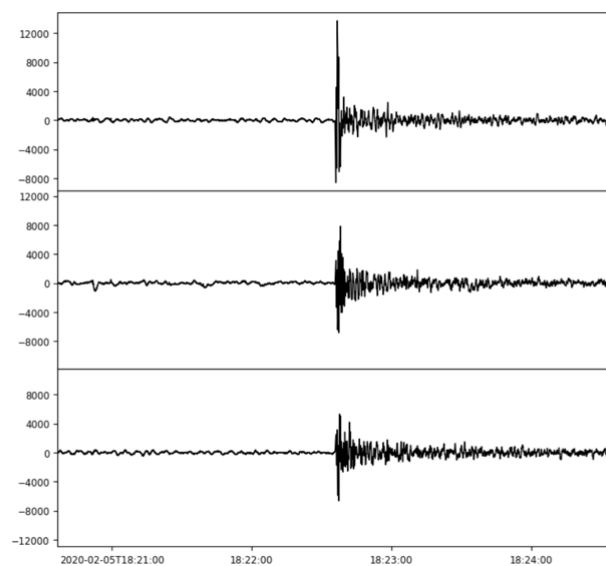
Figure 712、707、713 代表的地震段，存在异常振幅。

Figure 702、703 代表的地震段，频率较低。

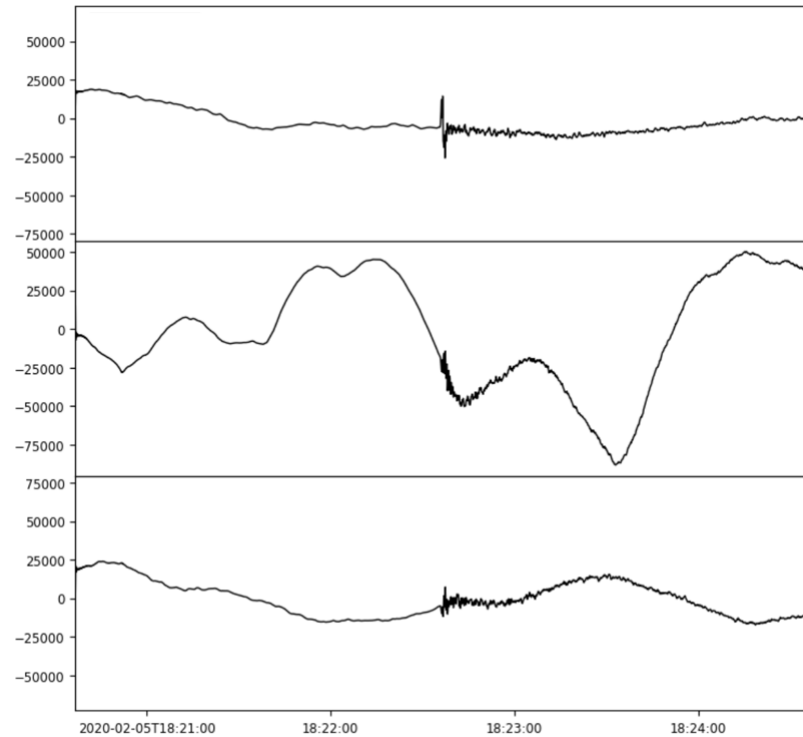
我们可以用 Obspy 绘制原始波形。通过对比 **CRF_job** 输出的图像以及一些原始波形，或者结合一些其他外部的质量控制代码来删除掉不好的地震记录。由于 CRF 程序运算量较大，请不要缺少该步骤。在检查异常数据时，建议创建一个 word 文档（参考地震波形检查.docx）来记录检查的依据和检查结果。为了方便检查，可以在 matlab 中输入 `global stnames` 来获取台站名对应的台站编号。

可能需要检查的项目：某个地震的异常道、异常的地震、R 分量与其他多数台站呈现相反特征的台站。

除了检查原始数据外，我们每做一步对数据的转换，都应该检查一下转换的结果是否正确。例如，原始数据可能为：



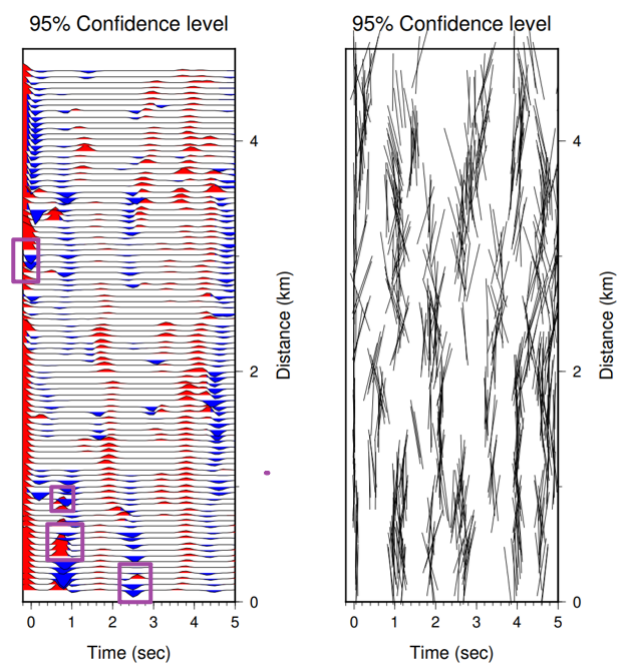
但是转换为 SAC 格式后的数据可能为：



以上问题可能是由于错误地去除了仪器响应造成的。

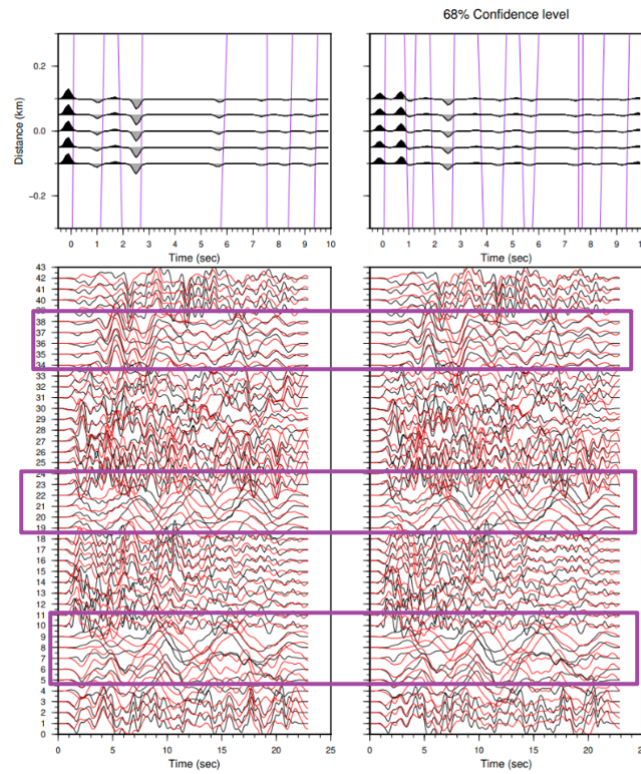
质量控制对 CRF 结果的影响

以下结果：利用所有地震事件，程序自动筛选；数据存在错误地去除仪器响应的情况

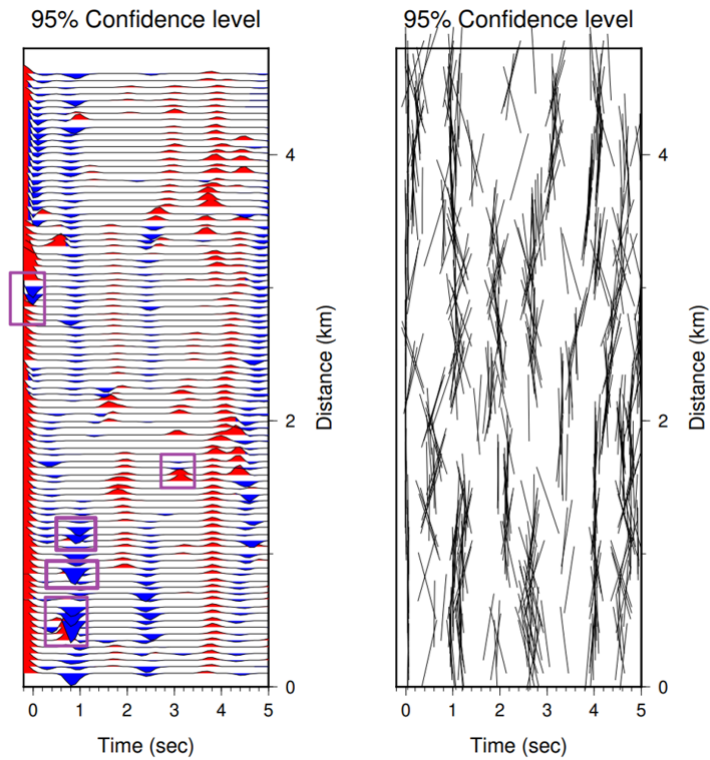


当台间距很小的时候，subarray 中子台站的数目可能需要增加。因为 CRF 中出现了少数不稳定的临近台站正负震相交替的情况。另外，CRF 的斜率似乎也不太稳定

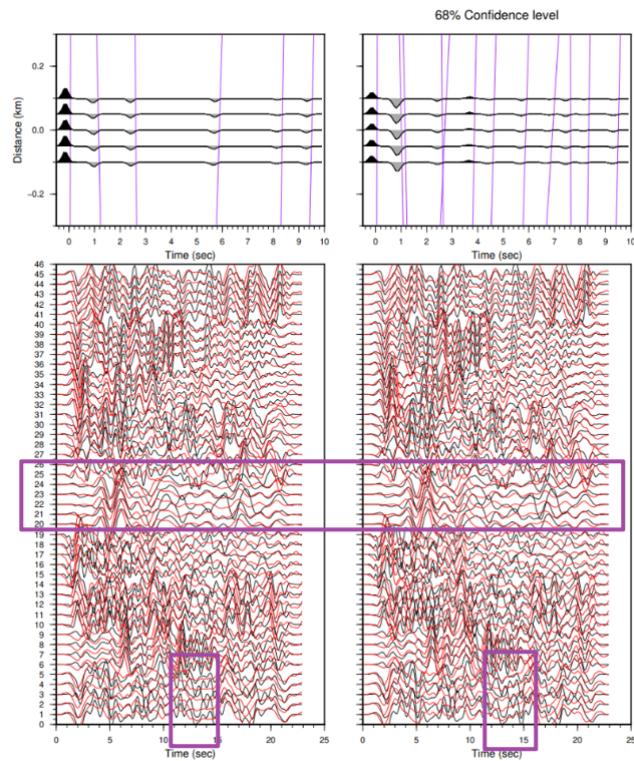
该方案中 sub-1 的波形拟合情况：



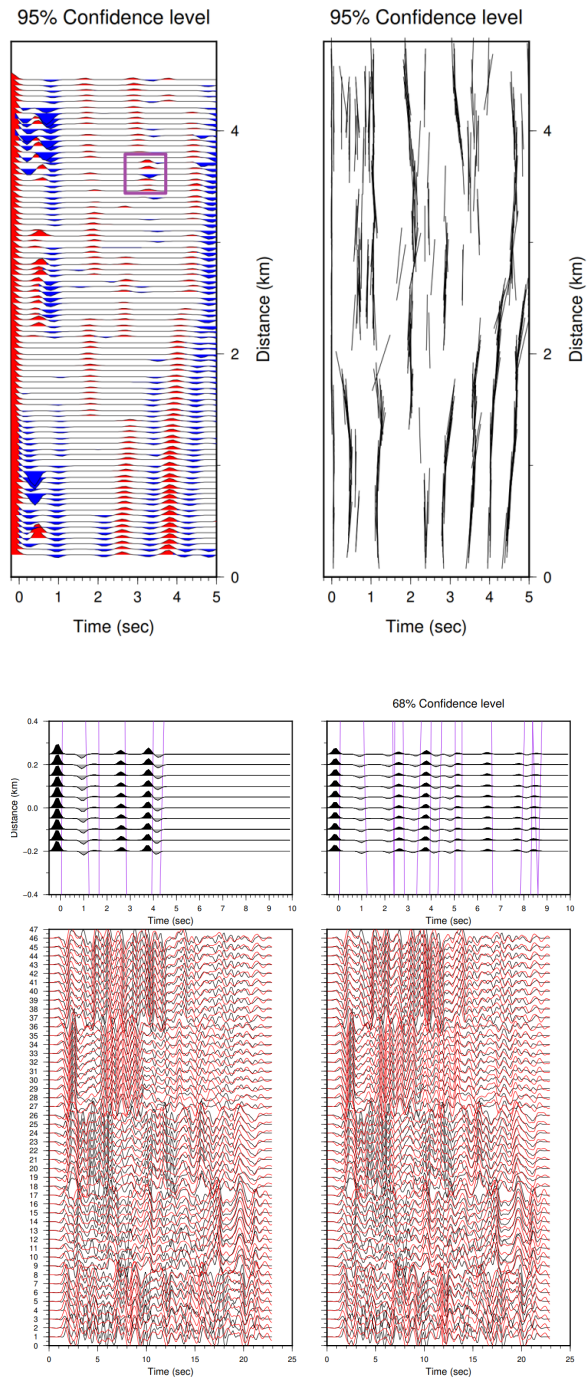
以下结果：没有错误地去除仪器响应；按照“地震波形检查.docx”中的记录，去除了异常振幅的记录、比较奇怪的地震，并校正了一个 R 分量反向的台站



以下为该方案中 sub-1 的拟合情况：



以下结果：没有错误地去除仪器响应；将地震数量减少到 5 个；将子台阵数量增加到 10 个（可能由于这个原因导致程序执行时间大幅上涨？？）



部分子台阵运行失败的情况

通过判断文件夹中是否存在结果文件 `average.out` 来确定是否需要重新执行

```
#!/bin/bash

recordname=Resubmit_`date +%Y%m%d`.txt

for n in `ls | grep sub-`
```

```

do

#echo $n
if [ -f ${n}/average.out ]; then
echo "${n} has done."
continue
else
echo "${n} re-submit."
echo "${n} re-submit." >> ${recordname}
fi

echo ""#!/bin/bash
#SBATCH --account=xxxx
#SBATCH --partition=pub
#SBATCH --ntasks=100

module load openmpi/4.0.5_OPA_gcc9.3
"" > submit_one_job.sbatch

echo 'cd $SLURM_SUBMIT_DIR' >> submit_one_job.sbatch

echo ""cp *.bin *.info ./${n}/data/

cd ./${n}/
mpirun -np 100 rjMCMCinv.o"" >> submit_one_job.sbatch

sbatch submit_one_job.sbatch

done

echo >> ${recordname}
echo `date` >> ${recordname}
echo >> ${recordname}

```

文件输出各列可能的含义

- (1) average.out: 时间, 振幅, 斜率, 未归一化的 Num
- (2) posterigor.out: 时间, 斜率, 振幅, 未归一化的概率
- (3) bestmodel.out: 第一行为震相数, 及 Likelihood; 其余行为时间, 斜率, 振幅