

分类号

密级

中国地质大学（北京）

本科毕业论文

题目 利用背景噪声进行面波成像的研究

英文题目 A Study on Surface Wave Tomography
based on Ambient Seismic Noise

学生姓名 刘昕 院（系）地球物理与信息技术学院

专 业 地球物理学 学 号 10105225

指导教师 李红谊 职 称 讲师

二〇〇九年五月

摘 要

本文主要讨论了通过互相关运算从两个台站的背景地震噪声记录中提取经验格林函数的数据处理方法和技术,特别是从原始噪声记录中去除地震等干扰信息以保证经验格林函数中的波群均来自背景场噪声互相关计算的技术。因此,数据处理的过程分为3个阶段:(1)单个台站的数据准备;(2)互相关运算及时间域的叠加;(3)提取群速度频散曲线并作面波层析成像反演。我通过编写 Bash shell 脚本、SAC 宏和 C/C++源程序来完成各个环节的工作,并且通过 Matlab2008 和 Access 小型数据库交互更有效的组织和处理了 7000 余条路径的互相关数据,得到了去除冗余的数据。在第3阶段的工作中,我采用 FTAN(频率-时间分析)图中交互式选择提取频散速度点的方法从互相关函数中提取了 3300 余条路径的群速度频散曲线。层析成像反演的网格剖分为 70×70 个,网格大小为 0.2° 。此外,我做了互相关函数随季节变化的分析,从中得出冬、春两季的 SNR(信噪比)明显高于夏、秋两季。信噪比的方位分析表明 NW-SE 方向的信噪比最强,而与之垂直的 NE-SW 方向上信噪比最差。最后,我结合周期为 6, 10, 15, 25s 的层析成像图对意大利及其周边区域的面波群速度特征作了解释,结论是意大利西边的第勒尼安海属于洋壳结构,为高速区域;意大利东边的亚得里亚海属于陆壳结构,表现为低速。意大利北边波河平原和阿尔卑斯山区在短周期($T < 20$ s)表现为低速。

关键词: 噪声互相关; 群速度频散曲线的提取; 面波层析成像; 季节性变化; 信噪比方向性

Abstract

In this paper, we mainly discuss the methods and techniques in data processing of cross-correlating ambient seismic noises from two stations (station pair) and especially the techniques to remove earthquake from the original noise record and reveal surface wave signals from the correlations. Hence the data processing procedure divides into three principal phases: (1) single station data preparation, (2) cross-correlation and time domain stacking and (3) measurement of group velocity dispersion curves and inversion for group velocity tomography. I wrote shell scripts, SAC Macros and C/C++ source codes to accomplish each phase of work. Moreover, I applied database interaction via Matlab 2008 to organize the mass data of 7000+ paths in a more efficient way and select unique station coordinates from the redundant data source. In the third phase of the work, I extracted group velocity dispersion curves from the empirical Green's function (cross-correlations) of more than 3300 paths by interactively picking up the velocity dispersion points from the FTAN (Frequency-Time Analysis) image. The tomography inversion is based on a 70×70 grid with a delta of 0.2° . In addition, the seasonal variation in the stacked cross-correlation function is analyzed and I conclude that the SNR (signal to noise ratio) is evidently higher in winter and spring than in summer and autumn. I also calculated the direction preference of the signal and discovered that the SNR in NW-SE is apparently higher. Finally I interpreted the tomography figures on periods of 6, 10, 15, 25s and I conclude that the Tyrrhenian Sea basin has an oceanic crust and the Adriatic Sea has a continental crust.

Keywords: noise cross-correlation; group velocity dispersion curve extraction; surface wave tomography; seasonal variation; SNR direction preference

目 录

摘要.....	- 1 -
Abstract.....	- 2 -
1 综述.....	- 5 -
1.1 研究方法简介及意义.....	- 5 -
1.2 研究区域概况.....	- 6 -
1.3 研究主要内容和结论.....	- 7 -
2 原理.....	- 8 -
2.1 互相关提取格林函数.....	- 8 -
2.2 群速度频散曲线.....	- 9 -
2.3 影响噪声成像结果的几个因素.....	- 12 -
2.3.1 信号强度的方向性优势.....	- 12 -
2.3.2 信噪比的方向性.....	- 13 -
2.3.3 台站路径的方位性.....	- 13 -
3 数据处理.....	- 14 -
3.1 方法技术.....	- 14 -
3.1.1 Bash Shell 脚本.....	- 14 -
3.1.2 SAC 宏.....	- 14 -
3.1.3 Matlab 数据库交互及其地震学应用.....	- 15 -
3.1.4 Linux 中的 C/C++.....	- 16 -
3.2 单个台站数据处理.....	- 17 -
3.2.1 前期处理.....	- 17 -
3.2.2 前期处理续：儒略历转换及 C++ 程序.....	- 17 -
3.2.3 时间域归一化处理及 C 程序.....	- 18 -
3.2.4 频率域白噪声化.....	- 21 -
3.3 台站间互相关和叠加运算.....	- 22 -
3.3.1 互相关操作.....	- 22 -
3.3.2 叠加处理（意义）及其算法.....	- 23 -
3.4 方向性检验.....	- 23 -
3.4.1 收集路径信息并计算 RMS.....	- 23 -
3.4.2 建立数据库.....	- 24 -
3.4.3 绘制方位角分布图.....	- 26 -
3.5 提取频散速度曲线.....	- 27 -
3.5.1 PROGRAM.330 简介.....	- 27 -
3.5.2 提取方法及工作脚本.....	- 27 -
3.6 层析成像反演.....	- 28 -
4 研究结果与分析.....	- 29 -
4.1 噪声分布的季节性变化.....	- 29 -
4.2 噪声源分布的方向性分析.....	- 30 -
4.2.1 台站路径的方向性.....	- 30 -
4.2.2 噪声源的方位性分布.....	- 30 -
4.2.3 信噪比的方向性.....	- 32 -
4.3 瑞利面波群速度成像结果及分析.....	- 33 -

5 结论..... - 36 -

致谢..... - 37 -

参考文献..... - 38 -

附录..... - 39 -

 脚本源代码（`Bash shell`, `SAC` 宏） - 39 -

 C/C++源代码..... - 47 -

 Matlab 源代码 - 52 -

1 综述

1.1 研究方法简介及意义

从地震背景噪声中提取面波格林函数是近 5 年来地震学面波层析成像领域的一种新型方法。它不依赖于地震的发生，只需要有连续的噪声记录，并且地震台网所在区域的地震活动不太剧烈时，即可对台站两两作互相关运算提取其连线上的格林函数。由于该方法不依赖于地震的发生，所以它可以获得更好的路径覆盖率和分辨率。

这种通过互相关运算从噪音中提取格林函数的方法最早应用在太阳地震学（helioseismology）中。在一项被称为声学日光成像（acoustic daylight imaging）的方法中，人们利用迈克尔逊多普勒成像仪（Michelson Doppler Imager, MDI）对太阳表面各点噪声进行记录[e.g., Duvall et al., 1993; Kosovichev et al., 2000; Rickett and Claerbaut, 2000]。其精度为 1024×1024 点阵，相当于百万个日震台的噪声记录。通过对太阳表面噪声进行互相关运算，得到了表达时距关系的日震图，进而得到了太阳外层的三维流速度结构。

Campillo et al. 选择了与背景噪声同样特征杂乱无章的地震尾波(coda) [1]。通过对台站记录的 101 个远震事件的相同时间段的尾波记录两两进行互相关计算，再作叠加处理，进一步提高运算结果的信噪比。经对比，该波形具有 Rayleigh 面波和 Love 面波的极化特征和群速度特征。

根据 Weaver et al. 的研究，通过互相关运算提取格林函数需要满足模式均分(modal equipartition) 的条件。对于地震背景噪声，其激发波场的源包括大气扰动、海洋微震、人类活动等，虽然在空间上分布不均匀，但随着散射程度的加大，多次散射使能量在相空间变得均一，因此满足提取格林函数对模式均分的要求。

Shapiro et al. 利用背景地震噪声提取台站间的格林函数。他们在 2004 年对两个相隔较远台站 39 天的噪声记录作互相关计算而得到了格林函数，进而获得地下结构速度信息。使用背景噪声代替地震尾波的优点很明显，它不依赖于地震的发生，而且在任何地点都可以作连续记录。

2006 年，Yingjie Yang et al. 对欧洲大部分地区作了 Rayleigh 波背景噪声层析成像，周期范围为 10s ~ 50s。他们对 125 个宽频台站 2004 全年记录的数据以天为单位分段进行互相关运算，得到的结果叠加至一年。相比传统地震方法，该方法的有效路径覆盖更为密集，且其方位角分布更为均匀。此外，得到的群速度图与背景噪声数据的匹配程度更高。

1.2 研究区域概况

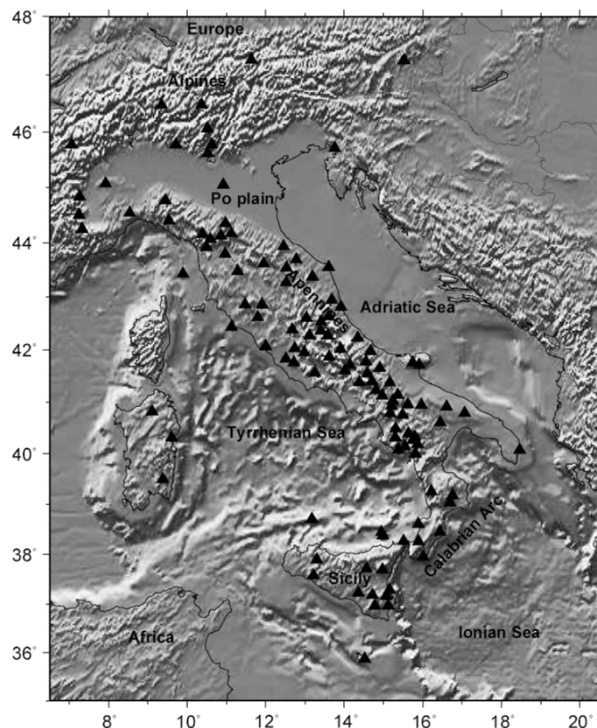


图 1.1 意大利地形及台网分布图（引自[6]）

意大利位于欧洲南部，包括亚平宁半岛（Apennines Peninsula）以及西西里岛（Sicily）、撒丁岛（Sardine）等岛屿。北以阿尔卑斯山（Alpines）为屏障，东、西、南三面临地中海的属海亚德里亚海（Adriatic Sea）、爱奥尼亚海（Ionian Sea）和第勒尼安海（Tyrrhenian Sea）。意大利中部为 NW-SE 向的亚平宁山脉（Apennines），亚平宁山脉和北部的阿尔卑斯山脉之间的地带为波河平原（Po Plain）。

我们的数据来自意大利 INGV（Istituto Nazionale di Geofisica e Vulcanologia，意大利国家火山和地球物理研究中心）台网的 114 个台站，记录时间从 2005.10~2007.03。台站分布见图 1.1。意大利亚平宁半岛的狭长地形决定了其台站间路径方位角分布的不均匀性。

意大利半岛构造复杂，导致其地震速度结构存在较大的横向不均匀性。

根据 A. Pontevivo et al. 的研究，他们使用了 200 余条路径的基阶瑞利波频散曲线（109 条来自新的地震事件的提取）作意大利及其周边地区的面波层析成像。他们提取频散曲线的方法是 FTAN（频-时关系分析法）。其层析成像反演的网格剖分为 $1^{\circ} \times 1^{\circ}$ ，横向分辨率（lateral resolving power）约为 200 km。此外他们还做了 5 个区域的一维 S 波平均速度模型。根据层析成像结果和一维 S 波平均速度模型，他们将意大利及其周边分为 7 个区域[4]。

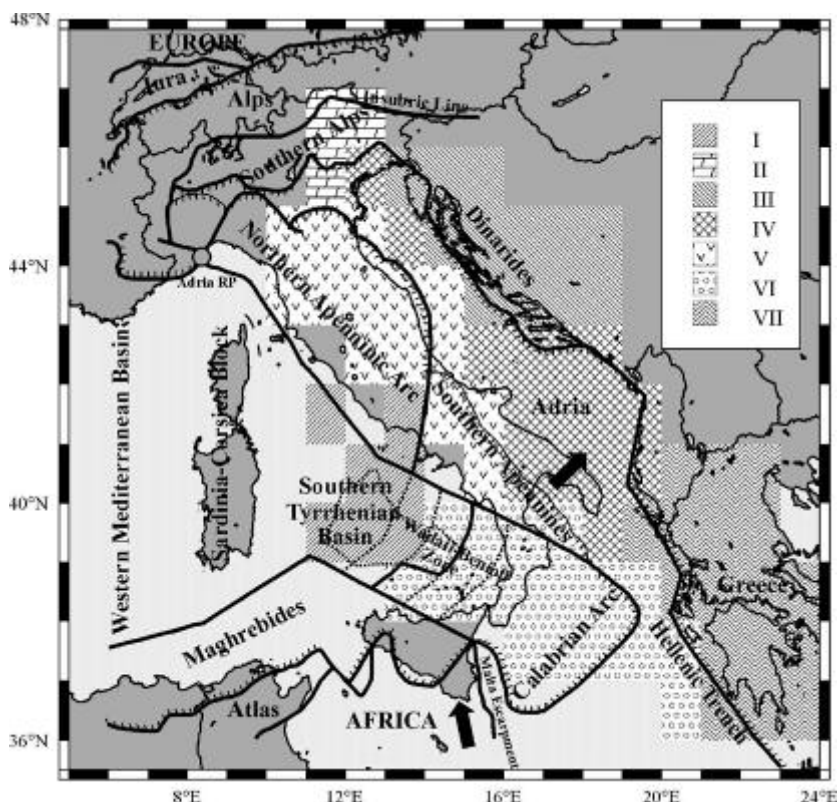


图 1.2 意大利及其周边的 7 个区域的划分（引自[4]，Fig 8）。南第勒尼安盆地（I），NE 阿尔卑斯（II），迪纳里德山脉（III），亚得里亚海（IV），北亚平宁弧和南亚平宁（V），卡拉布里亚弧（VI），希腊（VII）。

1.3 研究主要内容和结论

我们的主要研究从噪声记录的互相关运算中提取格林函数的数据处理方法，同时从 3300 余条路径的格林函数中采用人机交互的 FTAN 方法提取 Rayleigh 面波的频散速度曲线，并重点描述意大利及其周边地区周期分别为 6, 10, 15, 25s 的层析成像图。

互相关数据处理方面，我亲自编写或改写了整个噪声互相关处理流程中的 shell 脚本，C/C++ 程序，SAC 宏和 Matlab 程序及数据库，总代码量千余行，节选部分代码放在附录中。我发现时间域的归一化处理可以显著抑制噪声中的地震信号，而且滑动绝对窗法和 1 比特法是效果最好的。部分路径信号的叠加效果有明显的季节性差异；意大利地区的背景噪声主要来自 NW 方向，可能跟英吉利海峡海浪对欧洲大陆西北岸的冲击有关。关于数据处理的方法、技术和流程，在第 3 章有详细介绍。

层析成像反演方面，我们设定群速度的平均值为 2.68 km/s，划分为 70×70 个小网格，网格间距 0.2° 。最终得到了 29 张不同周期的 Rayleigh 面波群速度层析成像图（Tomography）。

我们的结论与 A. Pontevivo et al. 大体一致，不同的是第勒尼安盆地存在高速区而不是低速区，低速区南移至西西里岛。

2 原理

2.1 互相关提取格林函数

该方法源于太阳地震学中的应用。

传统地震学中，格林函数就是一点为震源，另一点接收的地震记录，两点相隔一定距离。该记录始于震源事件发生（ $t=0$ ），终止于地震尾波（Coda）的结束[1]。通常，震源与台站的距离越远，面波到时越晚，频率成分越趋向低频。

而从背景噪声中提取格林函数并不要求地震事件的发生。相反，地震事件的波形在噪声记录中反而属于干扰信息需要剔除。在该方法中，只需各台站在相同时间内连续的噪声记录即可。格林函数可以通过对两个台站的噪声记录作互相关计算得到。

从背景地震噪声中提取格林函数需满足以下基本假设：

1. 弹性体内散射波场或随机噪声场应近似均匀；
2. 经过足够长的时间，噪声源的空间位置分布随机化，由于介质的不均匀性而产生散射使噪声进一步随机平均化。

背景地震噪声是杂乱无章的。与弹道波相反，完全散射波场中，波可以有随机的相位和振幅，并可沿任意方向传播。由于完全散射波场包含了任意路径的信息（格林函数），一条射线经过一个台站后，随后又被另一个台站所接收，震相不发生变化因而是相关的。故可以对任意两个台站的噪声记录作互相关计算以提取其连线上的格林函数[Shapiro et al., 2004]。

根据 Weaver et al. 关于模式均分的假设，弹性体内的散射波场可以表示为：

$$\phi(x, t) = \sum_n a_n u_n(x) e^{i\omega_n t} \quad (2.1)$$

其中 x 为空间坐标， t 为时间， $u_n(x)$ ， ω_n 分别是地球的本征方程和本征频率。 a_n 是模激励函数。散射波场有一个重要特征，其模振幅是互不相关的随机变量：

$$\langle a_n a_m^* \rangle = \delta_{nm} F(\omega_n) \quad (2.2)$$

δ 是 kronecker 符号， $F(\omega)$ 是能谱密度， $\langle \rangle$ 表示互相关计算。由于平均来说（2.2）式的交叉项（ $n \neq m$ 项）会消失，故分别位于点 x ， y 的波场之间的互相关运算可简化为：

$$C(x, y, t) = \langle \phi(x, t) \phi(y, t) \rangle = \sum_n F(\omega_n) u_n(x) u_n(y) e^{-i\omega_n \tau} \quad (2.3)$$

与点 x ， y 之间真格林函数的表达式相比，（2.3）式仅相差一个幅度因子 F 。故而可以从背景散射场中提取两个台站之间的格林函数，方法是足够长时间的场到场的互相关计算。

设 v_A, v_B 分别为 A，B 两台站的连续噪声记录，那么 C_{AB} 可以由 v_A, v_B 的互相

关计算得到[Yao et al., 2005]:

$$C_{AB}(t) = \int_{-\infty}^{+\infty} v_A(\tau) v_B(t+\tau) d\tau \quad (2.4)$$

将 (2.4) 式离散化, 得到符合计算机处理要求的离散形式:

$$C_{AB}[n] = \sum_{m=-\infty}^{\infty} v_A[m] v_B[m+n] \quad (2.5)$$

其中 $C_{AB}[n]$ 表示互相关计算结果第 n 个点的值。实际计算中一般取有限个点,

故 $C_{AB}[n]$ 中 n 的取值为关于 $t=0$ ($n=0$) 对称的一个闭区间。

2.2 群速度频散曲线

地震学中, 面波主要分为 Love 波和 Rayleigh 波两种。Love 波质点运动方向与传播方向相垂直, 并在水平面内。而 Rayleigh 面波质点轨迹为一与传播方向平行并垂直于水平面的逆时针椭圆。

Love Wave

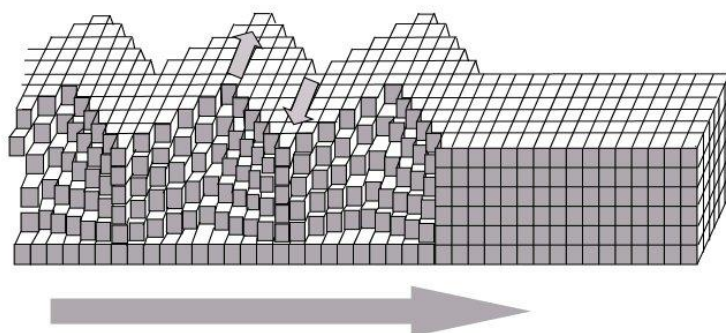


图 2.1 Love 波 (引自 Wikipedia 文献 http://en.wikipedia.org/wiki/File:Love_wave.jpg)

Rayleigh Wave

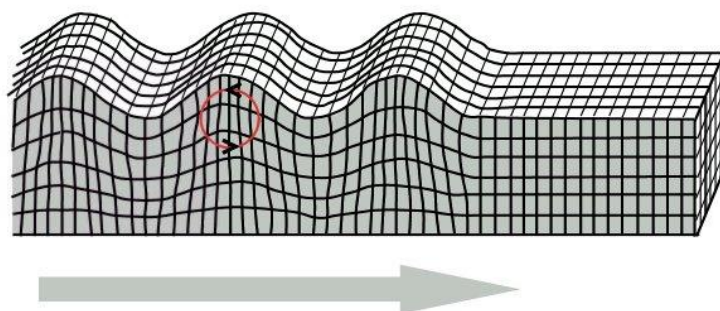


图 2.2 Rayleigh 波 (引自 Wikipedia 文献 http://en.wikipedia.org/wiki/File:Rayleigh_wave.jpg)

由于地表及地下密度分布的不均匀性, 面波在传播过程中极易产生频散。所

谓频散即不同频率的小波（Wavelets）以不同的速度传播，从而导致不同的到时和透入深度（趋肤深度）。频率相近的小波叠加在一起传播，称为一个波群。

相速度和群速度的理论最早由 Rayleigh 于 1877 年在 Theory of Sound 中提出并完善[2]。

相速度（Phase Velocity）是波形相位的传播速度：

$$v_p = \frac{\omega}{k} \quad (2.6)$$

其中 ω 为角频率， k 为波数。

我们以群速度（Group Velocity）来定义包含频率成分相近的地震面波的波形包络传播速度：

$$v_g = \frac{d\omega}{dk} \quad (2.7)$$

由（2.6）式和（2.7）式，如果 ω 与 k 成正比，则 $v_g = v_p$ 。而在非均匀各向异性介质中，相速度和群速度一般是不相等的，并且在地震面波中通常满足 $v_g < v_p$ 。由于能量与振幅的平方成正比，相速度通常被认为是能量或信息传播的速度。

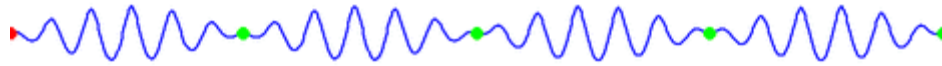


图 2.3 相速度和群速度示意图。两个相邻零振幅点之间的一段为一个波群。小波波形在传播的过程中其振幅发生改变，但其包络形状不变（不考虑衰减的情况下）。故可以用包络的传播速度来定义相速度。由于相速度一般快于群速度，所以小波波形在波群的波后出现，逐渐增长至振幅极大值，然后逐渐减小，在波前尖灭。

如图 2.4 所示，群速度频散曲线表示周期(period)的群速度(group velocity)的关系。一般来说，地下介质密度随着深度的增加而增加，高密度介质中面波波速更快。而周期越长的面波，透入深度约大。所以，长周期的波群通常具有更快的群速度。

由两个台站的噪音记录作互相关计算，得到该路径的格林函数。从格林函数中即可提取出群速度频散曲线。由于群速度表示能量的传播速度，因而群速度频散曲线必然与 FTAN (Frequency Time Analysis, 频-时关系分析) 图中的能量高值区域（图 2.4 红色部分）相吻合，以此作为提取依据（无论人工交互式提取还是软件自动提取）。Bensen et al. 的研究结果表明，为了获得可靠的群速度频散曲线，台站之间的间距至少需要三倍波长。而速度和周期的乘积等于波长，故提取群速度频散曲线应注意数据点的选取[5]。

具体来说，我们采用 Bensen et al. 介绍的方法[5]。设 $s(t)$ 为与 $S(\omega)$ 为一个傅立叶变换对，其中 $s(t)$ 为互相关折叠后格林函数包含信号的波形。在频域中定义解析信号为：

$$S_a(\omega) = S(\omega)(1 + \text{sgn}(\omega)) \quad (2.8)$$

其对应的时域表达式为：

$$s_a(t) = s(t) + iH(t) = |A(t)|\exp(i\phi(t)) \quad (2.9)$$

其中， $H(t)$ 为 $s(t)$ 对应的希尔伯特变换。为了重建时间-频率关系，让解析信号通过一系列以 ω_0 为中心频率的窄带带通高斯滤波器：

$$S_a(\omega, \omega_0) = S(\omega)(1 + \text{sgn}(\omega))G(\omega - \omega_0) \quad (2.10)$$

$$G(\omega - \omega_0) = e^{-a\left(\frac{\omega - \omega_0}{\omega_0}\right)^2} \quad (2.11)$$

对于每一段通过带通滤波器解析信号作反变换到时域，由（2.9）式即可得到平滑的二维包络函数 $|A(t, \omega_0)|$ 。以包络振幅平方的对数值得到一个矩阵，用该矩阵给定色标即可绘制 FTAN 图。

由地震台网中各条路径的群速度频散曲线，即可利用 Ditmar et al. 提出的方法进行面波层析成像反演，得到台网覆盖区域对应各个周期的群速度图[3]。

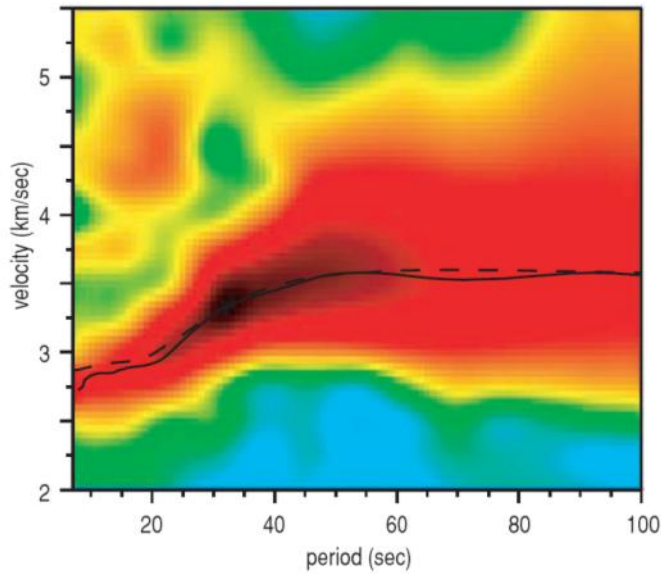


图 2.4 群速度频散曲线和 FTAN 示意图（引自 Bensen et al, 2007, Figure 13）。FTAN 图中红色指示能量高值区域，采用彩虹色标能量由高到低绘制。横坐标为周期（s），纵坐标为群速度（km/s）。红色区域与频散曲线对应，其范围越窄表明提取结果越精确。。

基于 Backus Gilbert 理论的反演算法是最为成熟的面波层析成像方法之一。该方法通过求目标函数（2.12）的最小值获得各个周期的群速度分布图：

$$(d - Gm)^T (d - Gm) + a \iint |\nabla m(x)|^2 dx = \min \quad (2.12)$$

其中

$$m(x) = (U^{-1}(x) - U_0^{-1})U_0 d_i = t_i - t_{0i} \quad (2.13)$$

$$(Gm)_i = \iint G_i(x)m(x)dx = \int_{l_{0i}} m(x) \frac{ds}{U_0} \quad (2.14)$$

$$\iint G_i(x)dx = \int_{l_{0i}} \frac{ds}{U_0} = t_{i0} \quad (2.15)$$

$x = x(0, \phi)$ 是位置矢量, U_0 是与初始模型相对应的速度, t_i 是沿第 i 条路径的观测走时, t_{0i} 是根据初始模型计算的走势, a 是正则化参数, l_{0i} 是第 i 条路径的长度, s 是参与反演的路径段。

2.3 影响噪声成像结果的几个因素

实际情况中, 由于噪声记录的时间不能无限长, 而且受地理因素的影响, 背景噪声场并非完全均匀。因而从背景噪声中提取格林函数的两个假设并不能总满足。这就导致沿路径的两个相反方向上的噪声信号强度及信噪比并不是完全对称相等。此外, 台网在研究区域的有效覆盖和各方向路径的均匀分布对于层析成像反演的精度有很大的影响。

2.3.1 信号强度的方向性优势

首先讨论信号强度的方向性优势。设两个台站 A、B, 规定正方向（传统地震学中事件到台站的方向）为从 A 到 B。互相关后的噪声信息包含两个分支, 正的分支表示信号沿正方向传播, 即从 A 到 B 的方向, 称为因果分支; 负的分支表示信号沿负方向传播, 即从 B 到 A, 称为非因果分支。理论上两个分支的信号是关于 $t=0$ 完全镜像的。然而, 受陆壳结构、海岸线分布（海浪冲击海岸）、人为活动等多种因素影响, 噪声源的分布不均匀, 因而两个分支的信号强弱是有差别的, 某些情况下（一定周期范围内某个方向的路径）差别非常显著。

我们使用 RMS（Root Mean Square, 均方根）方法分别计算正分支（positive leg）和负分支（negative leg）的 RMS 值, 定义 ratio 为正分支和负分支 RMS 值之比:

$$ratio = \frac{RMS(positive\ leg)}{RMS(negative\ leg)} \quad (2.16)$$

$$RMS(\{a_n\}) = \sqrt{\frac{1}{n} \sum_{i=1}^n a_i^2} \quad (2.17)$$

$\{a_n\}$ 为包含 n 项的数列。

若 $ratio > 1$ ，则噪声信号在正方向上（A 到 B）占有优势；若 $0 < ratio < 1$ ，则噪声信号在负方向上占有优势；若 $ratio = 1$ ，则两个方向噪声信号对称。

2.3.2 信噪比的方向性

提高信噪比（SNR, Signal-to-Noise Ratio）可以增加提取频散曲线和层析成像反演的可靠性。所以我们要研究信噪比的方位角分布以筛选高信噪比的路径并分析反演结果的精度。

我们计算信噪比的方法是：选择信号窗口和噪声窗口，分别计算 RMS 值。以正分支为例。信号窗口（signal）定义为 $(Dist/V_{\max}, Dist/V_{\min})$ ，其中 Dist 为两站距离， V_{\max} 为最大群速度， V_{\min} 为最小群速度。噪声窗口（noise）定义为单个分支的最后 50 秒噪声记录。

定义正分支的信噪比为：

$$SNR = \frac{RMS(signal)}{RMS(noise)} \quad (2.18)$$

由于负分支和正分支的时间轴对称，所以负分支的时间窗也与正分支的对称。同理，负分支的信噪比也由（2.18）式定义。

计算出信噪比后，即可绘制所有路径正、负分支分别的信噪比方位角分布图。一条路径总信噪比的信号窗、噪声窗分别是正负分支信号窗、噪声窗的并集。

2.3.3 台站路径的方位性

每条路径都规定一个正方向，因而都有一个方位角（Azimuth）。与方位角相差 180 度的方向称为后方位角（Back Azimuth）。绘制路径的方位角分布图时，路径的方位角和后方位角都要纳入统计，因而路径的方位角分布图是关于原点对称的。理论上背景散射场是均匀分布的，因而台网有效路径的方位角分布越均匀，反演效果越好。如果路径的方位角分布极为不均，就可能导致反演时通过某些网格的射线方向过于单一或条数过少，都会显著降低反演可靠性和精度。

3 数据处理

3.1 方法技术

3.1.1 Bash Shell 脚本

GNU Bash (Bourne Again Shell) 是当前 Linux 上的默认 shell。所谓 shell（外壳），与 core（内核）相对，即用户与 Linux 内核交互的接口。shell 是一个命令语言解释器（command-language interpreter）。拥有自己内建的 shell 命令集。此外，shell 也能被系统中其他有效的 Linux 实用程序和应用程序（utilities and application programs）所调用。

Bash shell 脚本类似于 Matlab 的 m 文件，但不同点是更为灵活，且具有模块化编程功能。我们可以使用任意一种 Linux 的文字编辑器来编写 shell 脚本。

一个典型的 Bash shell 脚本如下所示：

<code>#!/bin/bash</code>
<code>set -x</code>
<code>ads=`echo `this is a linux shell script by Liu Xin``</code>
<code>echo \$ads gawk `{print \$3,\$4,\$5}`</code>
<code>#end of script</code>

表 3.1 Bash shell 脚本示例，myfirstbash.sh

第一行 `#!/bin/bash` 告诉我们这是一个 bash shell 脚本。

当编辑好脚本时，如果要执行该脚本，还必须使其可执行：

```
$chmod +x myfirstbash.sh
```

运行脚本：

```
$/myfirstbash.sh
```

输出结果为

```
a linux shell
```

在本项目中，格林函数的自动化提取、目录文件操作、SAC 宏调用、频散曲线交互式提取、数据导出和后期处理等都要用到 Bash shell。Bash shell 不仅提供了 grep、gawk、sed 这样功能强大的字符、流和正则表达式的处理程序，还可以把 SAC 的模块、我们自己编写的程序以及其他脚本有机结合起来，极大地提高了数据处理分析、图件绘制的效率。

所有我编写的 Bash shell 脚本见附录。

3.1.2 SAC 宏

SAC (Seismic Analysis Code) 是一款基于 Unix 的免费地震数据处理软件，由美国 Lawrence Livermore 国家实验室开发。SAC 主要用于时序信号数据的处理。其主要功能包括通用算术运算，傅立叶变换，IIR 和 FIR 滤波器，谱分析，地震相提取，信号叠加，重采样和互相关等。

r file1.sac
* read the data
bp p 2 n 4 c 1 10
* use two way, 4th order bandpass butterworth filter
plot

表 3.2 SAC 宏示例, mytest.mac

SAC 宏是 SAC 软件包内置的一种编程语言，其语法类似于 fortran77，地位类似于 Microsoft Word 中的 Visual Basic 宏。

SAC 宏可内嵌任何 SAC 命令，具有基本的程序语言结构（顺序，循环，条件转折）但模块化支持不好，可执行外部程序，实现高精度（double）计算，并且支持环境变量(keys variable)和黑板报变量（blackboard variable）。

SAC 宏可以从 SAC 环境下的命令行调用并传递参数，因此 SAC 宏的调用通常在 Bash shell 脚本中实现，可以看所 Bash 脚本的一个子程序。

3.1.3 Matlab 数据库交互及其地震学应用

Matlab 是我们用到的唯一一个 Windows 下的程序。

使用 Matlab 的目的是从 7000 多条路径的文件中提取剔除冗余数据的台站的经纬度信息并作图。但更重要的一点，是数据库的应用可以把 7000 多个文件的信息保存在一个数据库文件中，而 Matlab 简化了与数据库的操作。

相对于传统 Linux 下的单个 SAC 文件的存储和管理模式，数据库的优势非常明显。

- 第一，数据库是完全的二进制存储，读写更新操作（Insert, delete, update）和查询比文本文件快很多，而且插入和删除任意多条数据记录后，数据表仍可保证有序。
- 第二，数据库可以包含多个关系表（table）和关系查询（query）。数据分散在几个关系表中可以保持冗余度最小，节省磁盘空间且方便更新（避免二义性冲突）。需要时可以通过连接查询（join query）快速从 2 个甚至多个表恢复需要的信息和记录。
- 第三，数据关系表的每一个字段都可以定义数据类型。SAC 文件的头信息和数据都可以放在数据库中对应的字段，相当于把原来的数千文件压缩成一个文件，查询速度还更快。

Matlab 可以通过 ODBC（Open Database Connectivity，开放数据库连接）来访问多种数据库，比如 SQL Server，Oracle，Java，MySQL 等。我们主要使用 Access 2003/97 格式的小型数据库，以便增强程序和数据库的可移植性。

Matlab 环境下数据库的访问虽然得到简化，但由于 Matlab 本质仍然是解释执行的一种高级语言，其运行速度远低于 C++ 等经过编译的目标代码。另外，如果希望数据库技术广泛应用在地震数据处理中，最佳选择是 GNU C++ 和跨平台的 MySQL。由于时间有限，我采用 Matlab 加 ODBC 的方案，为数据库技术在地震学中的应用提供借鉴。

3.1.4 Linux 中的 C/C++

虽然我们都学过 C/C++, 但 Linux 下的编程和 Dos/Windows 下的习惯完全不同。在 Linux 下, 所有代码都要自己写, 几乎没有可以用软件生成的框架。即使使用 KDevelop 这样的半自动化开发环境, 也不能如 VC++ 那样可视化的生成图形界面, 而必须用 GTK+ 或其他 XWindow 图形包首先生成图形界面部分。因此, 在 Linux 下开发 C/C++ 程序时, 一定要明确各个模块之间的相互调用关系并掌握编译预处理命令以检测头文件 include 是否有交叉引用存在。必要时绘制出各文件的树型包含关系图和模块结构图 (C 语言) / 类对象关系图 (C++)。

Linux 下的软件与 Windows 软件最大的不同是其开源性和免费性。一个 Linux 软件包通常就是它的源码包, 要想在单台机器上安装, 首先要在本地编译成可执行代码。

各种语言的编译器统称为 GCC (GNU Compiler Collection), 具有统一的调用风格和编译开关, C/C++, Ada, Fortran 等多种语言的编译器均包含在其中。使用 GCC 编译时, 首先要将程序包中每个源代码文件编译成.o 目标文件:

```
gcc -c -g myprog1.c
```

```
gcc -c -g myprog2.c
```

再将每个.o 文件连接起来得到一个可执行文件:

```
gcc -o -g myprog myprog1.o myprog2.o
```

也可以将上面 3 行编译命令写在一个 shell 脚本中。但这种方法有一个明显的缺陷: 假设源代码文件很多, 如果我们更改了一个文件, 就要重新编译所有文件。所以, 另一种自动编译方法 makefile 应运而生。

makefile 方法具有更新逻辑, 它会自动检查.o 目标文件和.c 源码文件的时间戳是否一致, 并且只会重新编译时间戳不一样的目标文件, 最后再将这些目标文件连接成单个程序文件。编译时, 只需键入命令 \$make 即可。

一个典型的 makefile 如下表所示:

main:	myprog1.o myprog2.o
	gcc -o myprog myprog1.o myprog2.o
myprog1.o:	myprog1.c myprog1.h
	gcc -c myprog1.c
myprog2.o:	myprog2.c myprog2.h
	gcc -c myprog2.c

表 3.3 makefile 示例

我用 GDB 调试编译后的 C/C++ 程序。GDB 可以自动链接源代码文件, 前提是 gcc 编译时加上 -g 开关。我们比较习惯了图形化的调试工具像 VC、BCB 等 IDE 的调试, 但有时候, 命令行的调试工具 GDB 却有着图形化工具所不能完成的功能。GDB 的功能概括如下:

- 1、启动程序, 可以自定义程序的运行参数;
- 2、可让被调试的程序在指定的断点处停住。(断点可以是条件表达式);
- 3、当程序被停住时, 可以检查此时程序中变量的当前值和堆栈调用情况, 并可以单步运行调试 (step, next);

- 4、动态的改变程序的执行环境。

最后, C/C++ 可以实现 Linux 系统级调用, 比如文件 I/O, 网络管理等。通

过 `system(char *command)` 函数亦可调用 Unix 命令。

本项目中,我有两个小程序分别采用 C/C++ 编写,总代码约为 200 行左右。

3.2 单个台站数据处理

单个台站的数据处理是整个处理流程中原始数据量最大的环节。它的目的是从单个台站的波形数据中剔除地震信号和仪器异常,只留下噪声。由于某些小地震信号非常微弱,仅仅从幅度很难将其剔除,只能通过滤波 (`filter`) 和归一化 (`normalization`) 的方法尽量消除其影响[5]。

单台数据处理包括以下环节:

原始数据消除仪器响应、均值、趋势,带通滤波,切割为一小时长度,旋转到大圆路径,时间域归一化和频率域归一化(谱白噪声化)。

3.2.1 前期处理

前期处理主要将单个台站连续波形记录分割为以小时为单位的文件并去除均值、趋势和仪器响应。如果原记录文件不是 `sac` 格式,需要先将其转换为 `sac` 格式。

去均值的命令为 `Rmean`,即每一个数据点减去该数据文件所有数据点的平均值,相当于平移运算。

去趋势命令为 `Rtrend`,即除去线性变化的趋势。其原理是用最小二乘法作某一文件中所有数据点的拟合直线。若拟合直线斜率非零,则所有数据点减去对应拟合直线函数值,令新的拟合直线斜率为零。

去除仪器响应的命令为 `Transfer`。由于地震仪记录的信号为感应电动势值 (`V`),所以给定仪器型号,通过逆卷积运算 (`deconvolution`),即可将其转换为质点加速度、速度或地面实际位移,三者是对时间差商关系[8]。

带通滤波器的命令为 `bandpass`。它相当于一个 `IIR` 带通滤波器,类型可以是 `Butterworth`, `Chebyshev`, `Bessel`。一般面波通带选择 2~100s。

经过前期处理后,由于已去掉均值和趋势,单个文件(1 小时)各数据点的平均值为零,波形数据也没有了线性变化的趋势,为下一步的噪声归一化处理做准备。

3.2.2 前期处理续: 儒略历转换及 C++ 程序

为保证各台站记录的同步性,地震台站记录的时间戳以格林尼治时间 (`GMT`) 为准从 `GPS` 获得,所以,前期处理时需要将噪声记录文件的时间戳由格林尼治时间转换为当地时间。以北京时间为例,从格林尼治时间转换为北京时间 (`GMT+8`),需要给每个文件的时间加 8 个小时,这就可能涉及到日期的变化。原先的年月日记录格式难以支持这样的转换,所以要将年月日的日期转化为儒略日,即某一天在这一年中是第几天(1 月 1 日起算,考虑闰年/平年)。

我编写了 C++ 程序 `dateconv` 来完成以上转换。

该程序包含 3 个 `.cpp` 源码文件,2 个 `.h` 头文件。从结构上看,该程序有一个

主函数 `int main(int argc, char *argv[])`，和两个类 `class juliaday` 和 `class monthday` 构成。Class `juliaday` 对象完成年月日到儒略日的转换，即输入 YYYYMMDD 格式的日期，转换为 YYYYDAY 的儒略日。Class `monthday` 对象完成儒略日到年月日的转换，即输入 YYYYDAY 格式的儒略日，转换为 YYYYMMDD 格式的日期。

运行该程序时，`-2j` 开关表示从年月日转换为儒略日；`-2d` 开关表示从儒略日转换为年月日。如表 3.4 所示。

输入1	<code>./dateconv -2j 20080120</code>
输出	<code>2008020</code>
输入2	<code>./dateconv -2d 2008120</code>
输出	<code>20080429</code>

表 3.4 dateconv C++程序调用范例

dateconv 的源代码见附录。

3.2.3 时间域归一化处理及 C 程序

时间域归一化处理 (Time-domain Normalization) 是单个台站数据准备中最重要的一环。因为它可以直接减弱地震信号、仪器异常即台站附近的非静态噪声 (non-stationary noise sources) 的影响。

由于时间域归一化方法和下面将要介绍的频率域归一化方法都会对波形作非线性的更改，所以二者的顺序非常重要，不可轻易颠倒。

根据 Bensen et al. 的介绍，被广泛应用的时间域归一化处理的方法包括 1 比特归一化 (Onebit normalization) 和滑动窗绝对平均归一化 (Running Absolute Mean normalization) [5]。

1 比特归一化的基本思想是：对于每一个数据点，如果为正就赋值 1，为负则赋值 -1。由于只有 1, -1 两种状态，因而得名 1 比特归一化。这种方法可以将地震和噪声的振幅都归一化为 1，虽然不能完全消除地震记录，但可以显著减小其在互相关计算中的干扰。如图 3.1 所示，由于数据点密集 ($\Delta t = 0.02$)，曲线重叠在一起，白线出现的地方对应图 3.2 b) 中的地震。因此，虽然地震信号和噪声信号的通带范围有部分重叠，但二者是可以区分的。

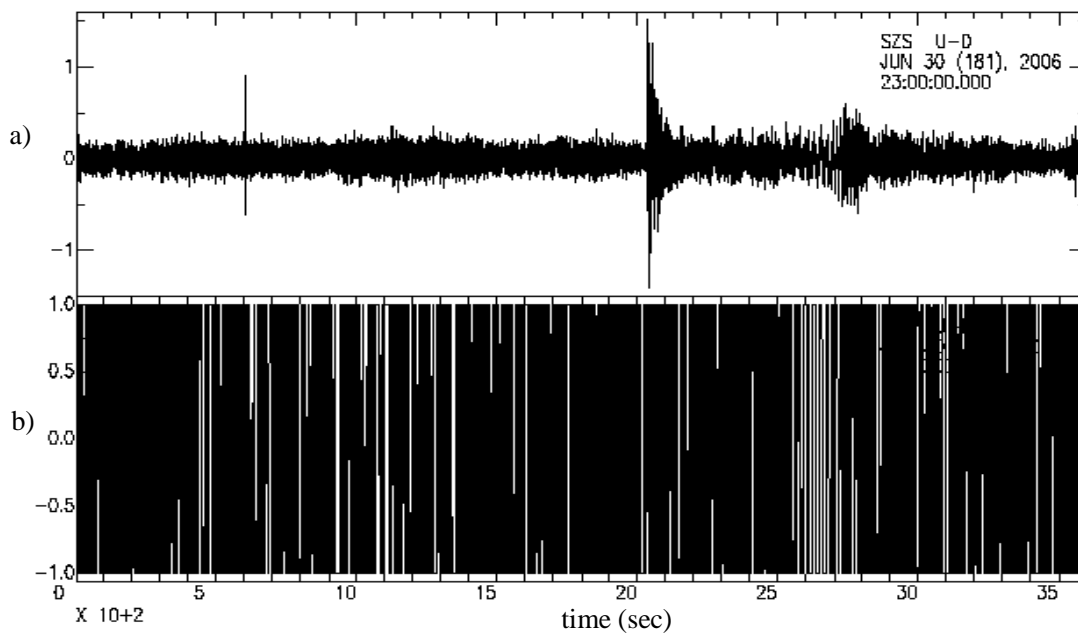


图 3.1 Onebit 归一化，数据源文件为 2006 年 6 月 30 日 23 点 SZS 台站 z 分量 1 个小时的记录。a) 图为原始记录；b) 图为 Onebit 归一化后的噪声记录。

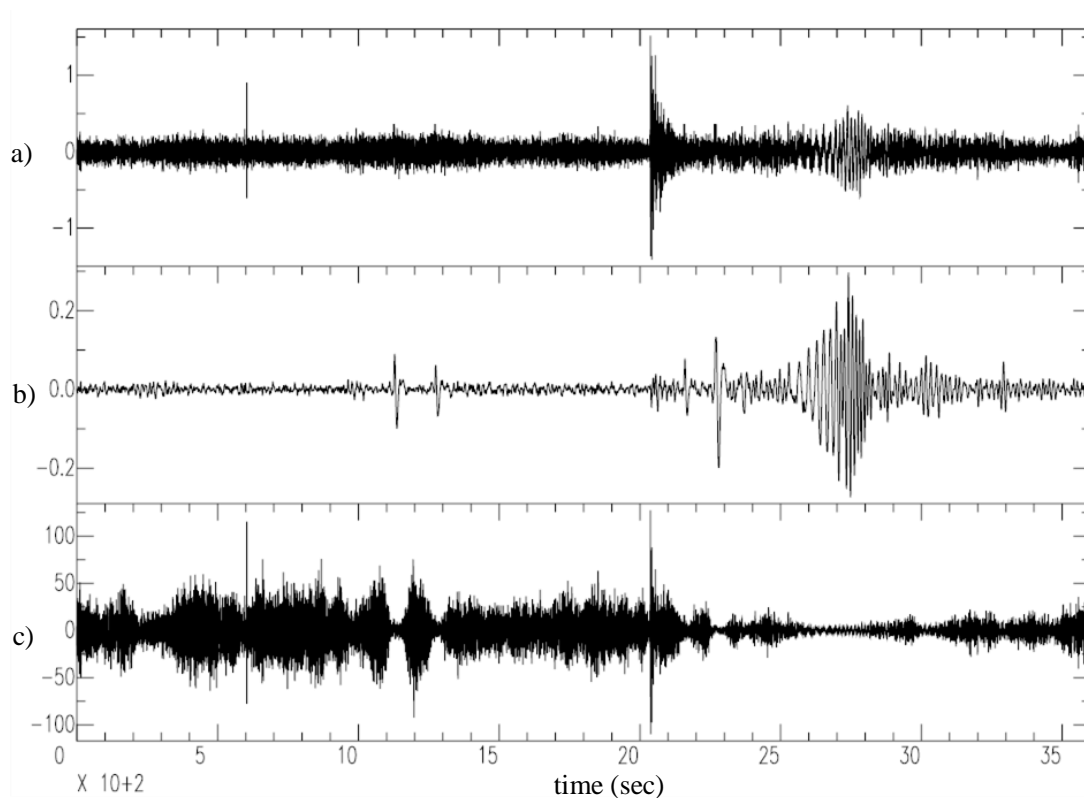


图 3.2 Running Absolute Mean 归一化。数据源文件为 2006 年 6 月 30 日 23 点 SZS 台站 z 分量 1 个小时的记录。a) 图为原始噪声记录，噪声和地震难以分辨；b) 图为带通滤波后揭示的地震记录，通带为 15~50s；c) 图为归一化后的噪声记录，地震信号已显著减弱。

滑动窗绝对平均归一化需要两个二进制输入数据文件，文件 1（file1）为原始数据文件，文件 2（file2）为文件 1 经过带通滤波后（通带一般选取 15~50 s）突出的地震信号。而 $2N+1$ 为滑动窗长度（点数）。

归一化权重的定义基于文件 2，公式为：

$$\hat{\omega}_n = \frac{1}{2N+1} \sum_{j=n-N}^{n+N} |\hat{d}_j| \quad (3.1)$$

其中 \hat{d}_j 为文件 2 中第 j 个数据点的值。得到权重后，按照公式 (3.2) 对文件 1 进行归一化：

$$\tilde{d}_n = \frac{d_n}{\hat{\omega}_n} \quad (3.2)$$

其中 d_n 为文件 1 中第 n 个点的原值， \tilde{d}_n 为文件 1 中第 n 个点的归一化值。

根据公式 (3.1)、公式 (3.2)，我设计了滑动窗绝对平均归一化的算法并用 C 语言实现（runabs.c 源代码见附录）。

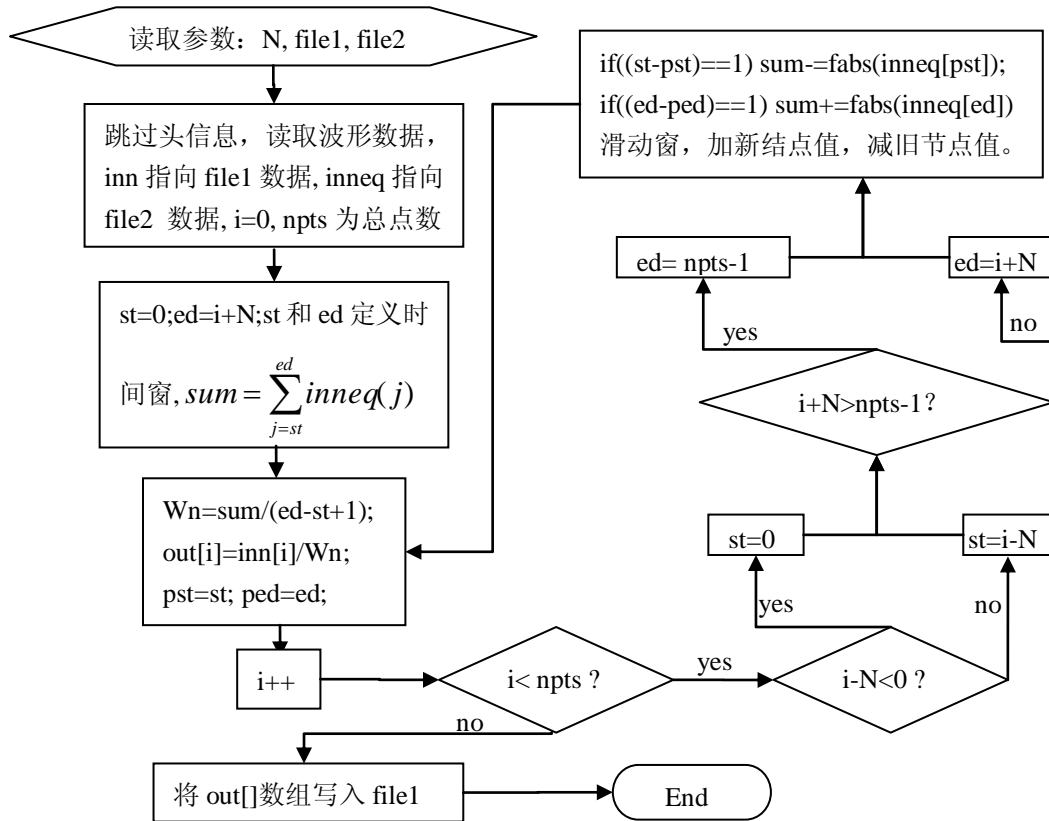


图 3.3 runabs.c 流程图。该程序实现了滑动窗绝对平均归一化的算法。

3.2.4 频率域白噪声化

经过时间域归一化的波形数据在频率域并不平坦，而是包含有峰值和起伏。这些峰值是一些微震信号（microseisms），可能是由于一些持续的单一频率噪声源（monochromatic noise source）的存在所致。而频域白噪声化（Spectral Whitening），即频率域的归一化，可以拓宽背景噪声在互相关计算中的频带并可以减小微震信号对计算的结果的干扰[5]。

根据 Bensen et al. 的研究，这些持续的单一频率噪声源通常随季节变化。频率域白噪声化可以很好的减弱来自这些持续的噪声源的信号。另外，频率域白噪声化也可以让频域背景噪声信号更为平坦。

频域白噪声化的命令为 **Whiten**，默认为 6 阶，即 6 个极点。阶数越高曲线就越平坦，但阶数过高可能导致重要信息丢失。

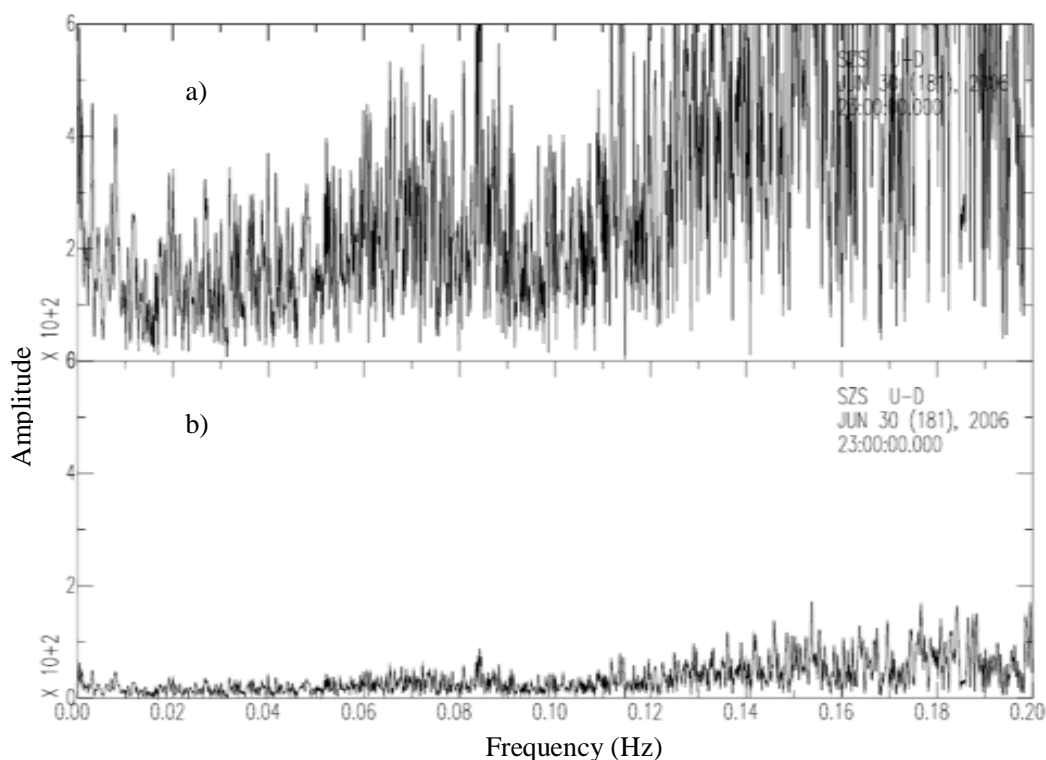


图 3.4 频域白噪声归一化。数据文件为 SZS 台站在 20060630 晚 23 点 1 小时长记录。
a) 未经过频域白噪声化处理的频谱，与图 3.2 c) 中的时序信号对应；b) 经过频域白噪声化处理的频谱，更平坦。

图 3.4 所示的频谱对比由我在 SAC 环境下对原始文件记录和频域白噪声化后的文件作 FFT（快速傅立叶变换）得到。可以看出 **Whiten** 后的频谱更为平坦。

3.3 台站间互相关和叠加运算

3.3.1 互相关操作

互相关计算的原理在前边已有介绍。这里重点讲互相关计算的实现。

SAC 软件包中，互相关计算的命令是 **Correlate**。只需读入两个文件（记作 file1, file2），执行 **Correlate**，则内存中自动生成两个新文件，一个是 file1 的自相关文件，其波形与单位脉冲响应函数（IMPULSE）一致。另一个便是 file1 与 file2 的互相关文件。所以保存时，我们只需要第二个文件，而第一个文件直接写入到一个固定的临时文件中，数据处理完后删除。

互相关计算的运算量和数据量都很大。如果有 n 个台站，则理论上的路径条数是 $\frac{n(n-1)}{2}$ 。对于意大利的数据，共有 114 个台站，每个台站有 12 个月的三分量数据，按每条路径每小时每个分量一个文件算，折合 172, 265, 400 个数据文件。由于计算机机时有限，我只选择了 7 个台站（28 条路径）6 个月的数据进行计算，共计约 362,880 个互相关数据文件，计算约耗时 22 小时（Xeon 4 核处理器）。实际中，由于台站间距小于一定数值的路径频散速度信息不可靠，而且某些路径的互相关计算结果很差，或者由于波形记录中包含较强的地震波形而放弃该路径，真正得到的未对称折叠的有效路径只有 7476 条（Italy 数据）。

我编写的互相关计算脚本思路如下：

1. 给出台站列表，存入变量 **stalist** 中；
2. 二重循环，**staA**, **staB** 分别选择 **stalist** 中的一个台站名并且 **staA** \neq **staB**，以输入流的方式在每次循环中将 **staA** 和 **staB** 以及时间传递给 **sac2008**，
3. 在传递给 **sac2008** 的输入流中执行 **Correlate**，作互相关计算。

以下为互相关运算的部分关键代码：

```
stalist=AMUR DOI SACS NOIP CINO
for staA in $stalist
do
  for staB in $stalist
  do
    if [[ $staA -eq $staB ]]; then
      continue
    fi
  done
done
sac2008<<EOF>sac.log
read $staA $staB
correlate
write tmp.sac Corr_{$staA}-{$staB}.sac
quit
EOF
```

表 3.5 互相关处理关键代码段

3.3.2 叠加处理（意义）及其算法

叠加处理的目的是增加互相关得到的格林函数的信噪比（SNR）以突出信号波形。叠加处理源于这样一个思想：虽然由于噪声源的时间空间分布的随机性导致背景散射场在较短时间内并非完全均匀，但通过长时间的互相关运算的叠加，可以使随机干扰相互抵消，而有用信号相加得到加强，从而提高了面波到时信号的信噪比。

叠加处理使用 SAC 软件包的 SSS（Signal Stacking Subprocedure, 信号叠加子过程）来实现，方法是编写一个具有编程接口的 SAC 宏，以便在 Bash shell 脚本中调用该宏。

叠加 SAC 宏 stacks.mac 的编写思想如下：

1. 给定台站对名 `stapair`，数据文件目录 `datadir`，和小时互相关文件的列表 `filelist`，即可将其作为参数传递给 `stacks.mac` 宏；
2. 对于每一个互相关文件，将其顺序、逆序加入到堆栈中（`ADDSTACK`）；
3. 堆栈文件添加完后，对堆栈求和 `SUMSTACK`，结果写入叠加文件中。

<code># stack.mac {stapair} {datadir} {filelist}</code>
<code>...</code>
<code>SSS # signal stacking subprocedure</code>
<code>DO file LIST \$filelist</code>
<code>ADDSTACK \$file</code>
<code>ADDSTACK \$file REVERSED</code>
<code>#one file is stacked two times, normal & reversed, 顺序和逆序叠加</code>
<code>LOOP</code>
<code>SUMSTACK</code>
<code>...</code>

表 3.6 stack.mac 叠加处理 SAC 宏关键代码段。

将 `stack.mac` 稍作改动，即可作各路径分季度的叠加互相关波形图（见附录 `stacks_season.mac`）。叠加效果的季节性分析将在 4.1 章节中给出。

3.4 方向性检验

3.4.1 收集路径信息并计算 RMS

本节的主要目的是计算互相关文件正、负分支的噪声 RMS 值和各分支信号窗口、噪声窗口的 RMS 值。

SAC 格式的文件包含一个长度固定（ 158×4 byte）的头信息（header），和长度可变的数据记录部分[8]。其数据记录部分长度等于 $NPTS \times 4$ ，格式为 float 型（4 byte），NPTS（Number of Points）为数据点数，可以从头信息中读取。

SAC 头信息不仅包含了诸如台站名称，台站经纬度，地震事件经纬度等信息，还可以自定义用户变量（格式为 `USER?`）。所以，可以把 RMS 命令的计算

结果存入用户变量中，再统一输出到文本文件 RMS.txt 中。

RMS 命令用于计算均方根值，但是调用前要首先给定计算窗口。例如：

MTW 0 50
* 设定 RMS 计算窗口为 0~50 秒
RMS to USER0
* 计算窗口内的 RMS 值并存入 USER0 头变量

表 3.7 RMS 值计算实例

噪声窗口和信号窗口的定义参考 2.3.2 节。

得到的 RMS.txt 文件包含有台站对名，台站 1 的经、纬度，台站 2 的经、纬度，正分支的 RMS 值，负分支的 RMS 值，正分支的 SNR 值，负分支的 SNR 值，路径的方位角。

3.4.2 建立数据库

本节的主要内容是建立数据库并将 RMS.txt 的内容导入数据库中。数据库可以极大地提高数据处理的速度，并能更有效的管理海量数据。

3.4.2.1 目的和方法

我们的目的是编写 Matlab 程序通过 ODBC（开放数据库互连）调用 Access2003 小型数据库并完成数据导入、分类和查询的操作。

首先使用 Access 建立两个数据库文件，文件内容见下表

File Name	Function	Contents
Italy1.mdb	辅助	临时数据，1 table
Italy2.mdb	主要	2 tables , 1 query

表 3.8 数据库文件

数据库文件 italy2.mdb 结构的详细介绍参见下一节。

然后，配置 Windows 的 ODBC 数据源信息，添加两个 Access2003/97 用户数据源 italy、italy2，分别指向 italy1.mdb 和 italy2.mdb 这两个文件。

在 Matlab 中，调用 importfile.m 导入原始数据，再通过 insert2db.m 将 7436 条数据的台站名、经纬度插入 italy1.mdb 的表 station 中。

其次，调用 getunique.m，从 italy1.mdb 的表 station 中查询数据并按台站名升序排。由于文本文件在保存数据时舍入误差的存在，所以有些台站即使名称相同，其经纬度也会有很小的差异。对于这部分冗余数据，我采用 Matlab 程序去除，重名台站的经纬度取升序排列时第一条记录的经纬度，把唯一的台站信息存入 italy2.mdb 的 stations 表。

通过调用 mkpathsdb.m，从路径关系数据中删除各台站的经纬度信息，仅

仅保留路径的起、止台站的名称，存入 italy2.mdb 的关系表 paths。

最后调用 paths_pos_retri.m，利用连接查询，重建绘制噪声方向性图所需的数据。

File Name	Explanation
importfile.m	从 RMS.txt 导入原始数据，存储于元胞数组（Cell array）
insert2db.m	插入 7436 个数据到 Italy1.mdb
getunique.m	选择保证唯一性的台站信息，包括台站名称，经度，纬度
mkpathsdb.m	删除冗余经纬度信息，建立新关系表 paths
paths_pos_retri.m	连接查询，重建绘图作序的数据表
plotmap.m	主函数，绘图

表 3.9 Matlab 数据处理 m 文件

涉及数据库操作时，所有 SQL 语句均可通过 Matlab 的 Visual Querybuilder（可视化查询语句生成器）来获得，极大地方便了数据库程序的编写。

3.4.2.2 数据库结构和连接查询

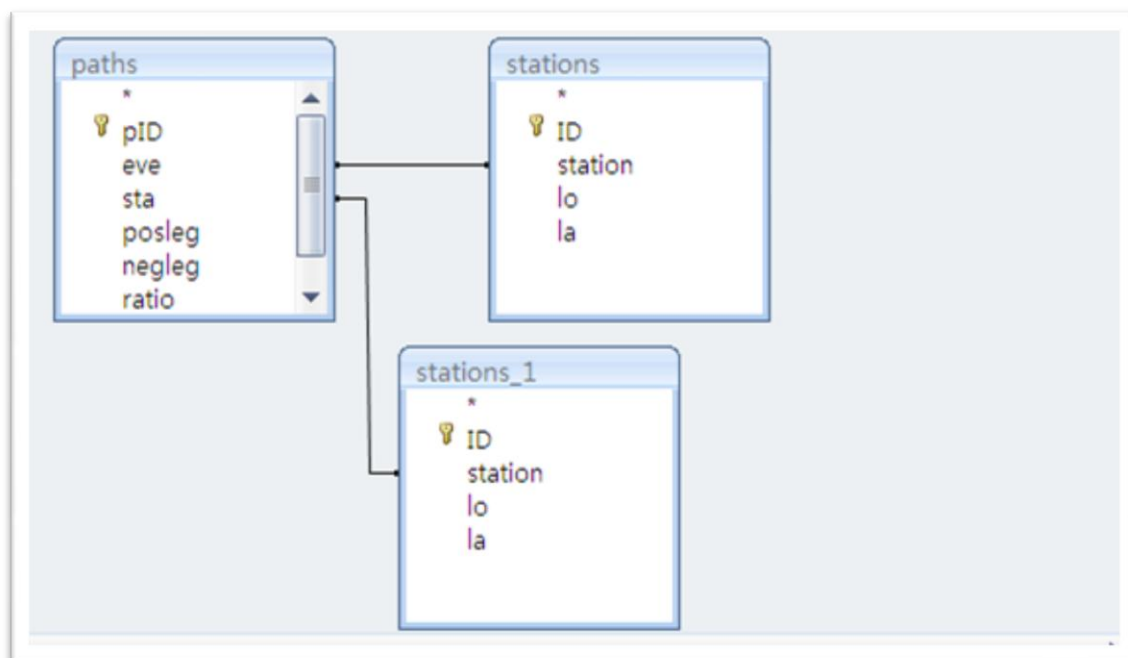
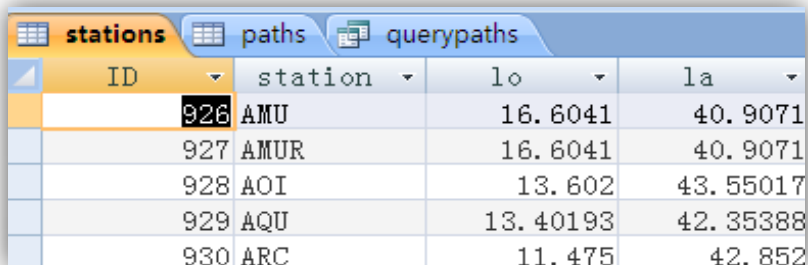


图 3.5 数据库逻辑结构图。paths 表的 eve, sta 字段与 stations 表的 station 字段关联

如图 3.5 所示，数据库 italy2.mdb 的逻辑结构包含 2 个关系表 paths, stations，和 1 个查询 querypaths。path 表主要存储路径信息及 RMS 值，其主键为 pID（路

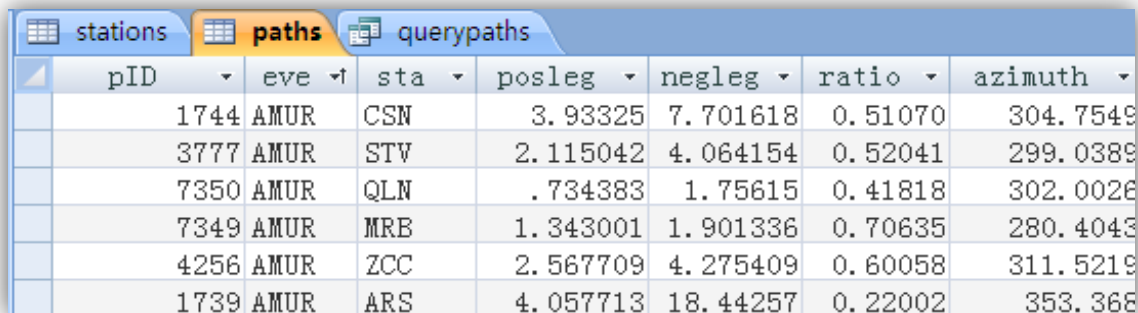
径编号)。station 表主要存储台站经纬度信息，主键为 ID，在 station 字段建立索引并增加唯一性逻辑。

querypaths，为数据库内置的连接查询，它将 stations 表映射为 stations_1，然后 paths 表通过 eve 和 sta 两个字段与 stations 和 stations_1 作连接运算，得到绘制噪声强度方向性分布图的数据。



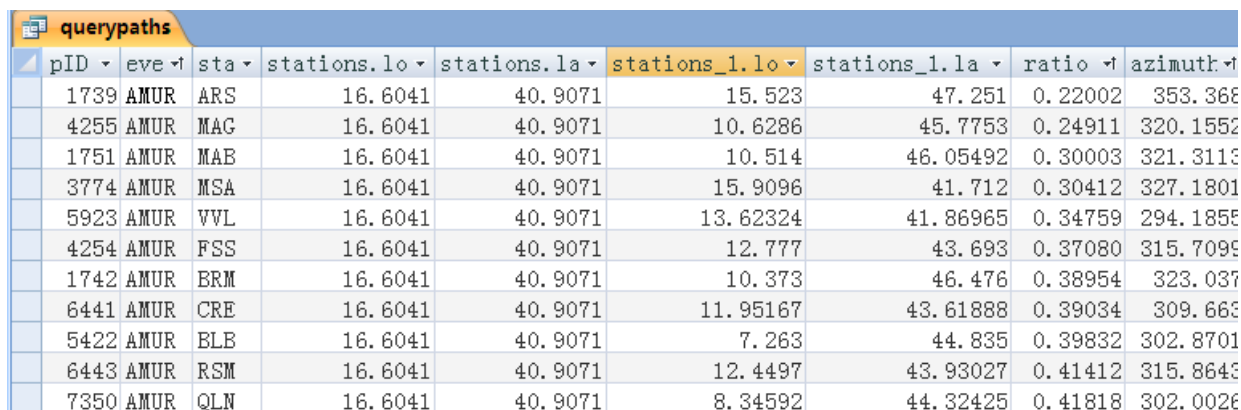
ID	station	lo	la
926	AMU	16.6041	40.9071
927	AMUR	16.6041	40.9071
928	AOI	13.602	43.55017
929	AQU	13.40193	42.35388
930	ARC	11.475	42.852

图 3.6 stations 关系表



pID	eve	sta	posleg	negleg	ratio	azimuth
1744	AMUR	CSN	3.93325	7.701618	0.51070	304.7549
3777	AMUR	STV	2.115042	4.064154	0.52041	299.0389
7350	AMUR	QLN	.734383	1.75615	0.41818	302.0026
7349	AMUR	MRB	1.343001	1.901336	0.70635	280.4043
4256	AMUR	ZCC	2.567709	4.275409	0.60058	311.5219
1739	AMUR	ARS	4.057713	18.44257	0.22002	353.368

图 3.7 paths 关系表



pID	eve	sta	stations.lo	stations.la	stations_1.lo	stations_1.la	ratio	azimuth
1739	AMUR	ARS	16.6041	40.9071	15.523	47.251	0.22002	353.368
4255	AMUR	MAG	16.6041	40.9071	10.6286	45.7753	0.24911	320.1552
1751	AMUR	MAB	16.6041	40.9071	10.514	46.05492	0.30003	321.3113
3774	AMUR	MSA	16.6041	40.9071	15.9096	41.712	0.30412	327.1801
5923	AMUR	VVL	16.6041	40.9071	13.62324	41.86965	0.34759	294.1859
4254	AMUR	FSS	16.6041	40.9071	12.777	43.693	0.37080	315.7099
1742	AMUR	BRM	16.6041	40.9071	10.373	46.476	0.38954	323.037
6441	AMUR	CRE	16.6041	40.9071	11.95167	43.61888	0.39034	309.663
5422	AMUR	BLB	16.6041	40.9071	7.263	44.835	0.39832	302.8701
6443	AMUR	RSM	16.6041	40.9071	12.4497	43.93027	0.41412	315.8643
7350	AMUR	QLN	16.6041	40.9071	8.34592	44.32425	0.41818	302.0026

图 3.8 连接查询 querypaths，由 paths 和 stations 表作连接运算得到

3.4.3 绘制方位角分布图

利用上一节连接查询导出的数据，可以方便的作出路径方位角分布图和噪声强度方位性优势图。最初我用 Matlab 作方向性优势图。但由于 Matlab 无法很好的支持底图上图形的绘制，所以最终我采用 Linux 下的 GMT4.0 绘制图件。

绘图脚本 makemap.sh 及 plt_SNR_rose.sh 见附件

3.5 提取频散速度曲线

3.5.1 PROGRAM.330 简介

PROGRAM.330 为圣路易斯大学地震实验室开发的人机交互式提取面波频散速度曲线的程序。该软件包的主程序 do_mft 采用 C 语言编写。而 FTAN（频率-时间分析图）的绘制和傅立叶变换、希尔伯特变换及矩阵运算均采用 Fortran77 编写。

3.5.2 提取方法及工作脚本

面波频散速度曲线是本项目中比较耗时的内容之一。我从 italy 3300 余条路径的折叠互相关叠加数据文件中提取频散速度曲线。

提取步骤如下：执行脚本 myjobs.sh, do_mft 程序自行启动，从 allpaths.lst 中选择一个未被标记的路径。然后根据台站间距离 DIST 设定各种参数，若 DIST 大于 250km，则 ALPHA 值设为 25；若 DIST 小于 250km，ALPHA 设为 12.5。这里我提取的是 R 分量，所以类型选择 Rayleigh 波。进入到所示的界面时，需要用鼠标在 FTAN 图上选取频散速度点。被选择的点高亮显示。最后保存.dsp 文件即可。

do_mft 程序结束后，脚本会询问路径的好坏。有 4 个选项可供选择：y(good), n(bad), e(error, 计算出错), q(quit, 退出脚本)。若选择为 y，则路径会被保留下来，并且会将 dsp 文件名改成相应台站对的名称。除 q 选项退出脚本执行外，其他选择都会记录在 allpaths.lst 文件中。

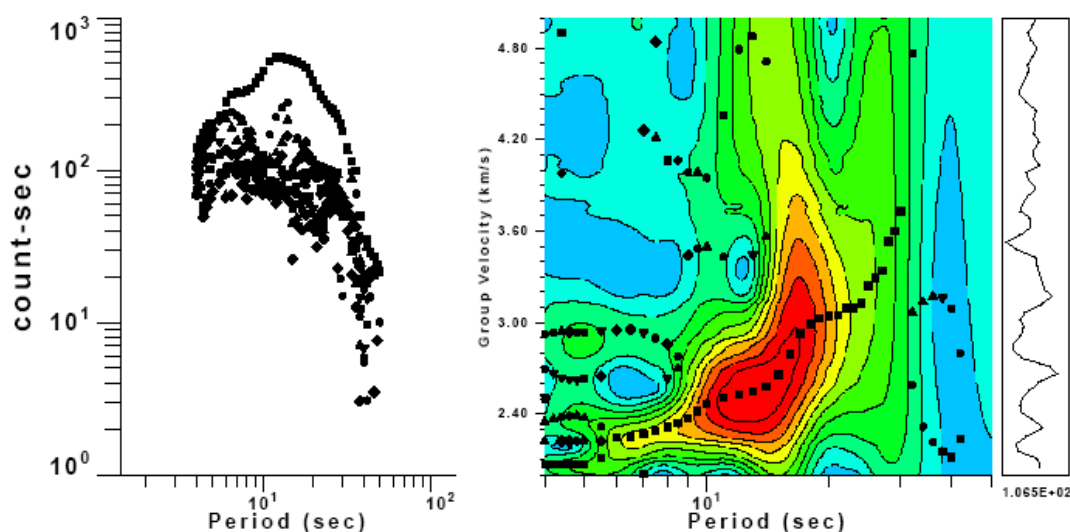


图 3.9 FTAN 图及面波频散速度提取。提取波形为 Rayleigh 波，红色区域对应右侧群速度包络振幅模的最大值。

myjobs.sh 源代码见附录。

3.6 层析成像反演

面波频散速度曲线提取完成后，每一个有效路径会得到 4 个文件：.dsp, .eps, .PLT 和.d 文件。其中.dsp 为频散速度信息（dispersion），最为重要。

要进行层析成像反演，首先要得到各个周期在不同路径上的频散速度信息。我编写了一个脚本 get_eachT.sh，用于自动从.dsp 文件中提取各周期的频散速度信息并将其写入该周期的.dat 文件中。最后得到 29 个以周期命名的.dat 文件，作为反演的输入文件。

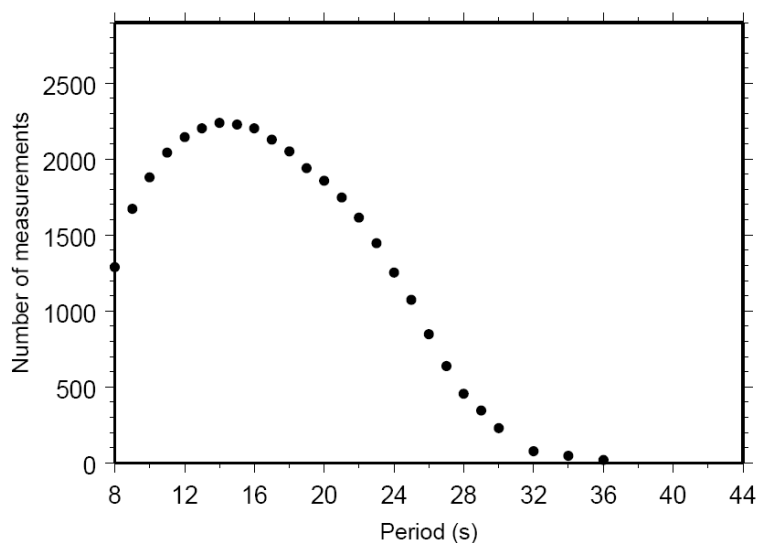


图 3.10 各周期有效测量个数

然后在 model.ini 中设定模型参数。平均速度设定为 2.68 km/s，网格划分为 70×70 个，以 0.2° 为间隔。

反演由脚本 doinv.sh（李老师提供）自动完成。反演完成后，得到各周期对应的实际速度模型，每个周期输出为 t*.dat 和 t*.out 文件。

执行 meanvelocity.sh（自己编写）脚本，从每个周期的 t*.dat 文件中获得速度的最大值、最小值和平均值，并将其作为参数传递给 plt_tomo.sh 脚本，即可做出层析成像图。

把 t*.out 文件传递给 plt_pthd.sh，即可得到该周期的路径覆盖密度图(paths density graph)。

4 研究结果与分析

4.1 噪声分布的季节性变化

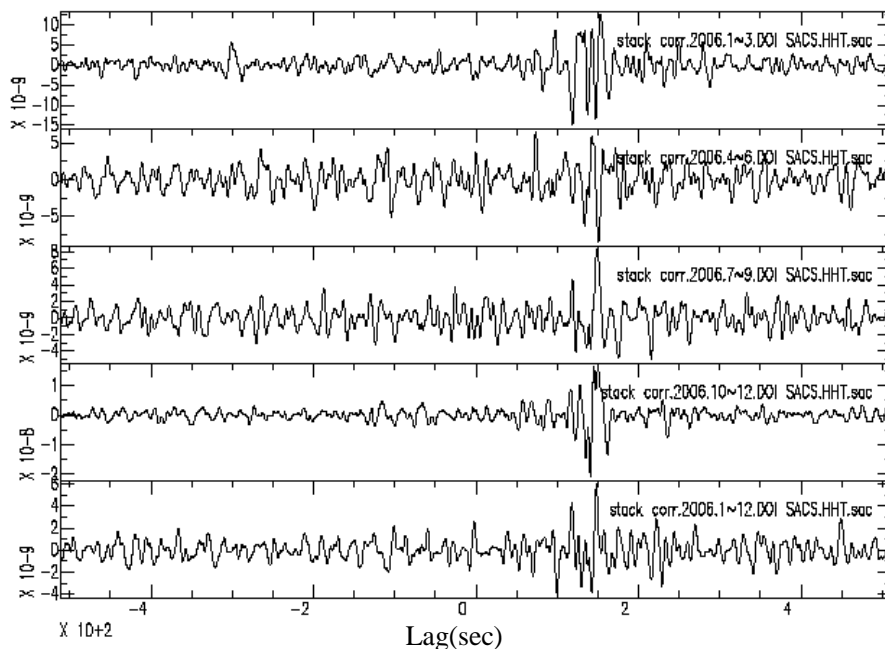


图 4.1 DOI-SACS 路径 T 分量 2006 年分季度叠加对比图。

如图 4.1, DOI-SACS 路径 T 分量的面波信号 (Love 波) 在冬季 10~12 月和春季 1~3 月比较明显, 信噪比高于夏秋季节 4~9 月的噪声互相关记录。而全年的互相关波形叠加 (1~12 月) 虽然没有冬、春季的信噪比高, 但是比夏秋季的信噪比要高。

从图 4.1 可以读出面波的到时在 150 s 左右。

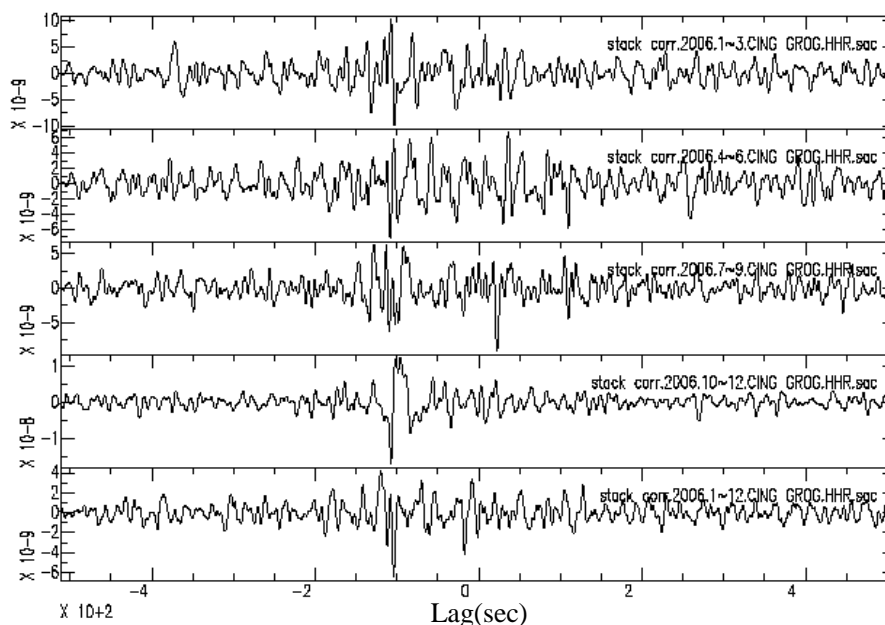


图 4.2 CING_GROG 路径 R 分量 2006 年分季度叠加对比图。

如图 4.2 所示, CING-GROG 路径 R 分量的面波信号 (Rayleigh 波) 在冬季 10~12 月和春季 1~3 月比较明显, 信噪比高于夏秋季节 4~9 月的噪声互相关记录。但是全年的互相关波形叠加 (1~12 月) 的信噪比比夏秋季要高, 与春季的信噪比接近。

综上, 背景噪声场在秋冬季节 (10~3 月) 较为均匀, 互相关叠加信噪比较高; 而在夏秋季节 (4~9 月), 由于人类活动较为频繁, 单频噪声源干扰加剧, 信噪比较差。长时间的叠加可以很好的消除互相关叠加波形中的随机干扰。

4.2 噪声源分布的方向性分析

4.2.1 台站路径的方向性

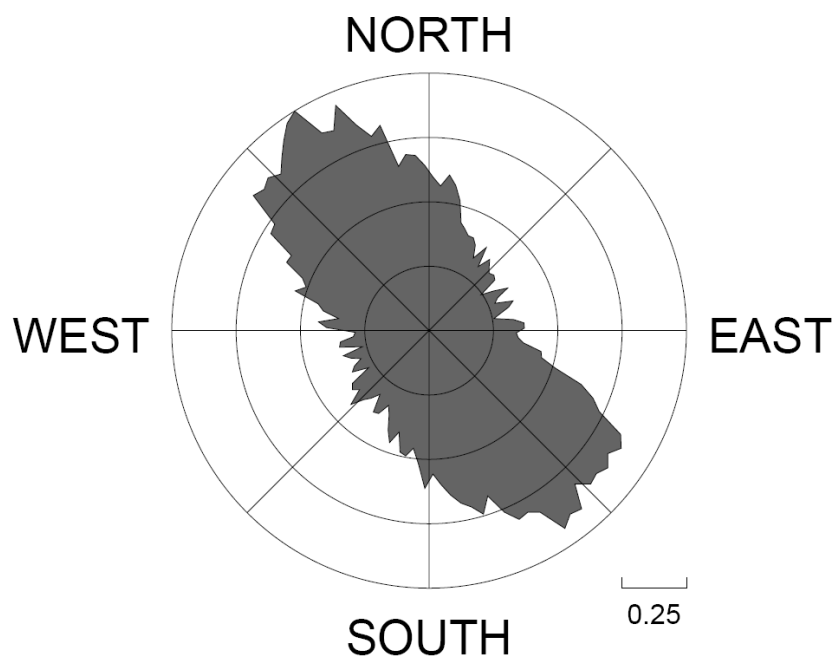


图 4.3 路径方向性。值越大表明该方向的路径数越多。按各方向路径数最大值归一化

台网中有效路径的方位角主要分布在 NW-SE 方向上。这是由意大利狭长的地形决定的。

4.2.2 噪声源的方位性分布

由公式 (2.17), 定义 ratio 为正分支和负分支 RMS 值之比, 规定由西向东传播的信号为正。如果某条路径的方位角 (Azimuth) 大于 180 度, 则将其旋转 180 度, 并将路径端点两个台站的顺序颠倒。

分别绘制 ratio 在不同取值域时各路径噪声的方位性分布图。

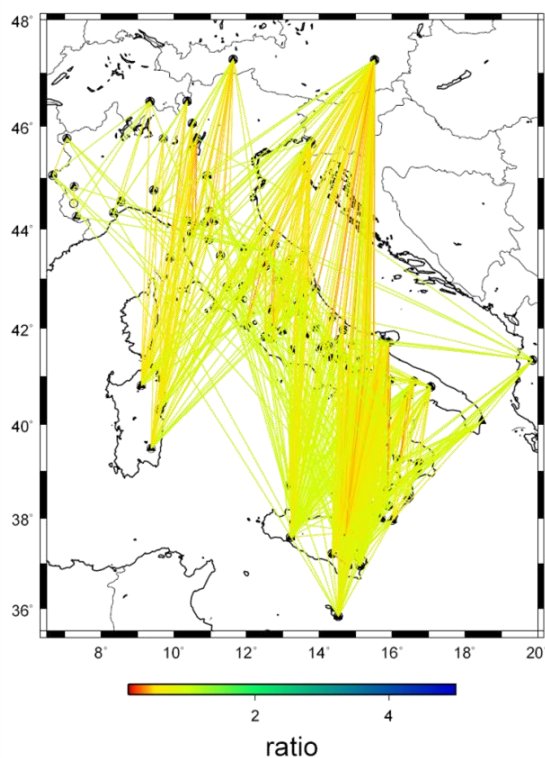


图 4.4 ratio < 1 时的路径噪声方位性图

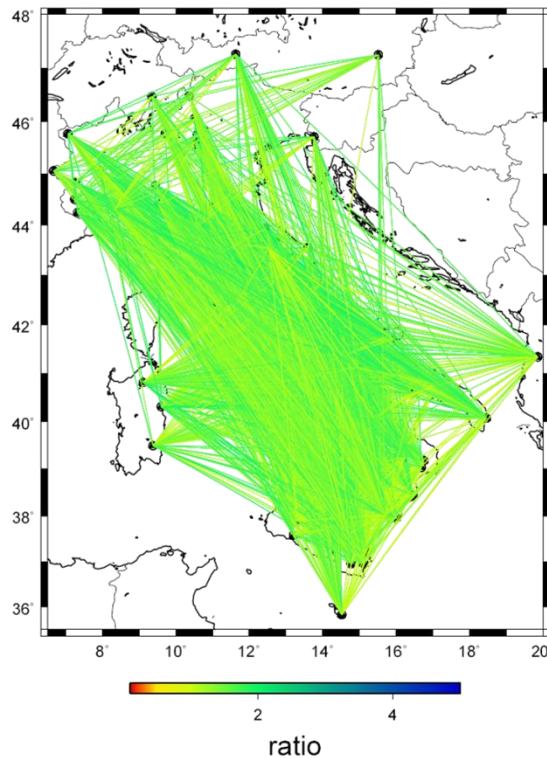


图 4.5 1 < ratio < 2 的路径噪声方位性图

当 ratio 小于 1，即信号沿着负方向（从东到西）传播占优势时，相关路径多集中在方位角 10 度范围内，且较为稀疏（1064 条路径），推测其主要来自北海。

当 ratio 大于 1 小于 2 时，即信号沿着正方向（从西向东）传播占优势时，相关路径多集中在方位角 130 度到 155 度范围内，且极为密集（3371 条路径），推测其主要来自大西洋海浪对英吉利海峡两岸的冲击。

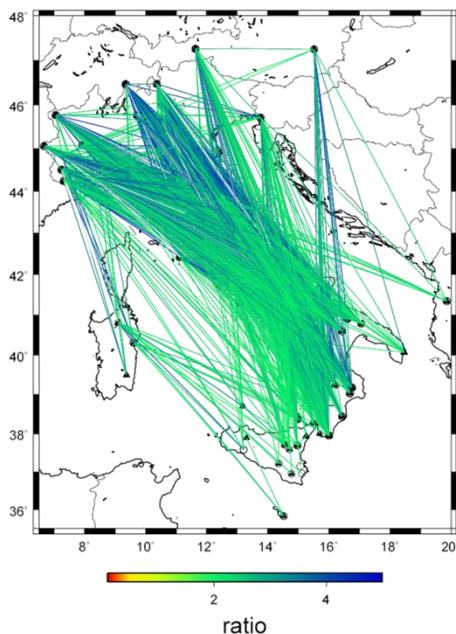


图 4.6 ratio > 2 的路径噪声方位性图

当 ratio 大于 2 时，即信号沿着正方向（从西向东）传播占较大优势时，相关路径多集中在方位角 125 度到 170 度范围内，且较为密集（3041 条路径），推测其主要来自大西洋海浪对英吉利海峡沿岸的冲击。

综上所述，噪声信号更多的来自大西洋海浪对西北欧海岸的冲击。

4.2.3 信噪比的方向性

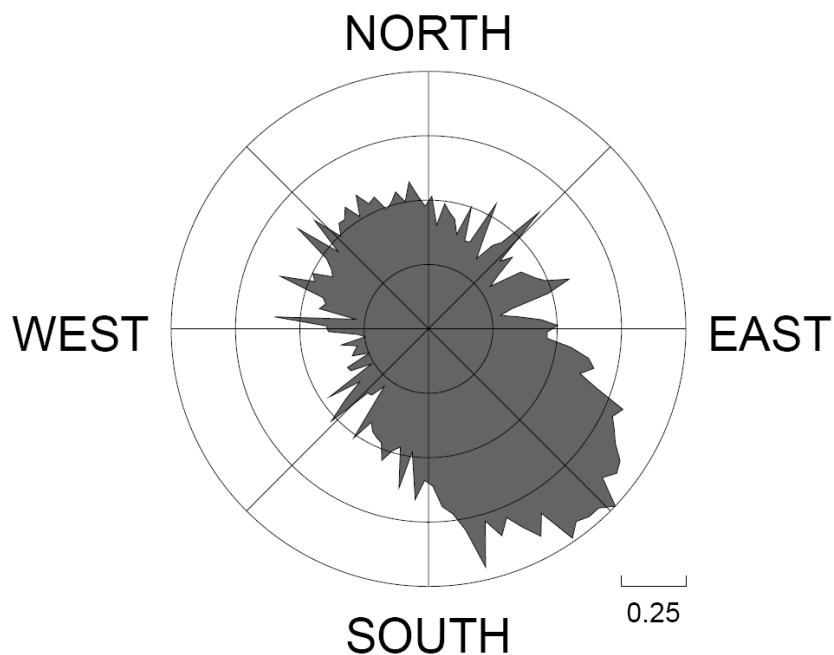


图 4.7 正分支 SNR 的方向性。值越大表明该方向 SNR 越好。最大值归一化为 1。

由图 4.7，向东南方向传播的噪声的互相关叠加函数的信噪比（SNR）最高，西北方向的信噪比次之，而东北-西南方向的信噪比最低。

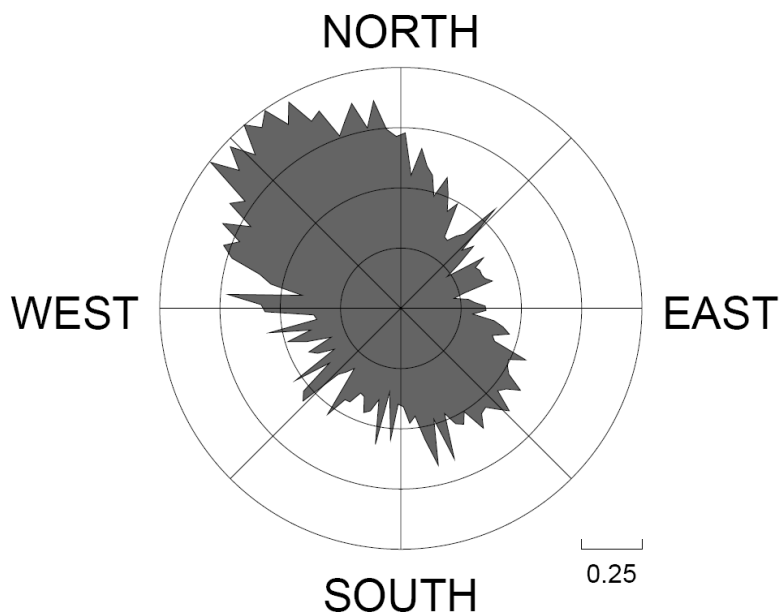


图 4.8 负分支 SNR 的方向性。值越大表明该方向 SNR 越好。最大值归一化为 1

由图 4.8，由于负分支为非因果信号，其传播方向与路径方位（Azimuth）相反，故本例的结果与图 4.7 一致，向东南方向传播的噪声的互相关叠加函数的信噪比（SNR）最高，西北方向的信噪比次之，而东北-西南方向的信噪比最低。

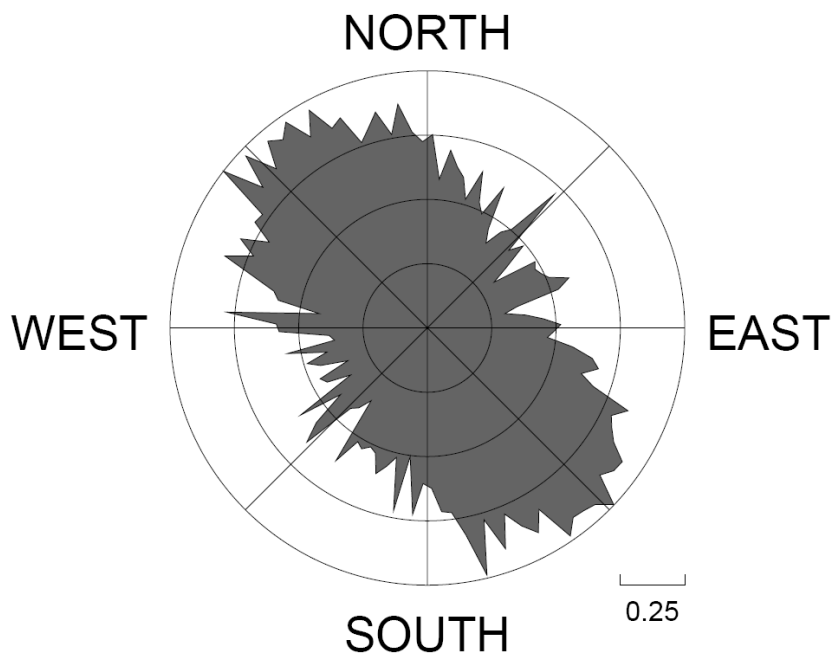


图 4.9 平均 SNR 方向性。值越大表明该方向 SNR 越好。最大值归一化为 1

图 4.9 是前两张图的综合，它表明 NW-SE 方向路径的信噪比较高，而与之垂直的方向 NE-SW 上信噪比较低。

4.3 瑞利面波群速度成像结果及分析

根据提取出的 2400 余条频散速度曲线，我做出了周期 5~30 s 的 26 张图和 32s, 34s, 36 s 的层析成像图，共计 29 张图。这里选取 4 张图作解释。

对应周期 $T=6$ s 的 Rayleigh 面波，在意大利北部的阿尔卑斯山区和波河平原存在低速区，意大利中部亚平宁山脉腹地存在高速区。

对应周期 $T=10$ s 的 Rayleigh 面波，意大利北部阿尔卑斯山区和波河平原的低速区有向南扩大的趋势，意大利中部亚平宁山脉以东速度较低，以西速度较高，西部的第勒尼安海（Tyrrhenian Sea）存在高速区，推测为洋壳结构。东部的亚得里亚海存在低速区，推测为陆壳结构。此时的路径覆盖较为密集。

对应周期 $T=15$ s 的 Rayleigh 面波，意大利北部阿尔卑斯山区及波河平原的低速区与亚平宁山脉以东的低速区连成一片，向东延伸至亚得里亚海。西部的第勒尼安海高速区没有扩大但仍然很明显。此时的路径覆盖最为密集。

对应周期 $T=25$ s 的 Rayleigh 面波，亚平宁山脉以东速度较低，以西速度较高。此时路径覆盖密度已显著下降，但是在意大利中南部和西西里岛仍有不错的覆盖。

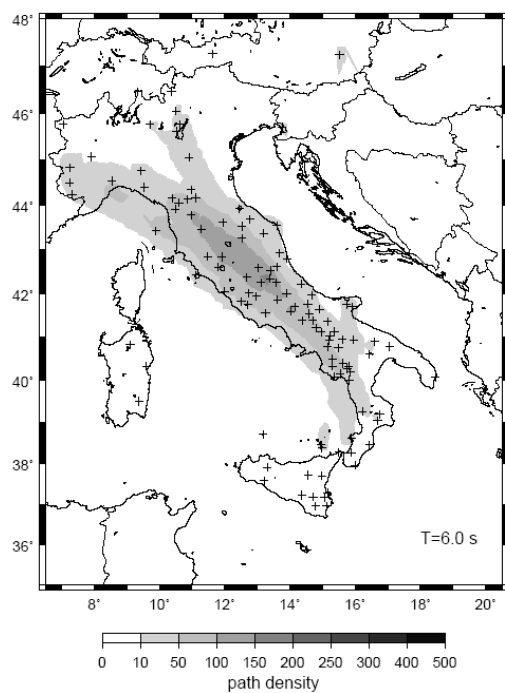


图 4.10 $T=6.0$ s 路径覆盖密度图

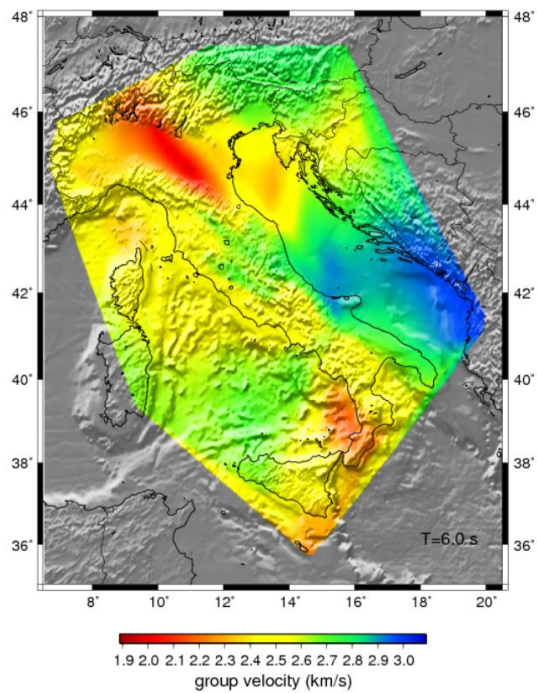


图 4.11 $T=6.0$ s 群速度层析成像图

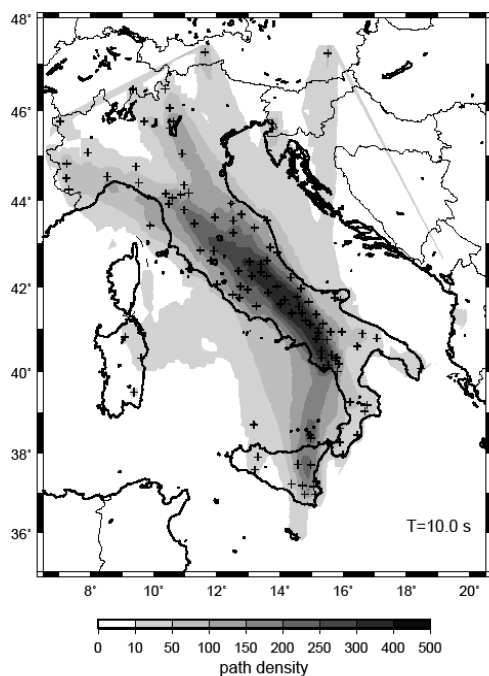


图 4.12 $T=10.0$ s 路径覆盖密度图

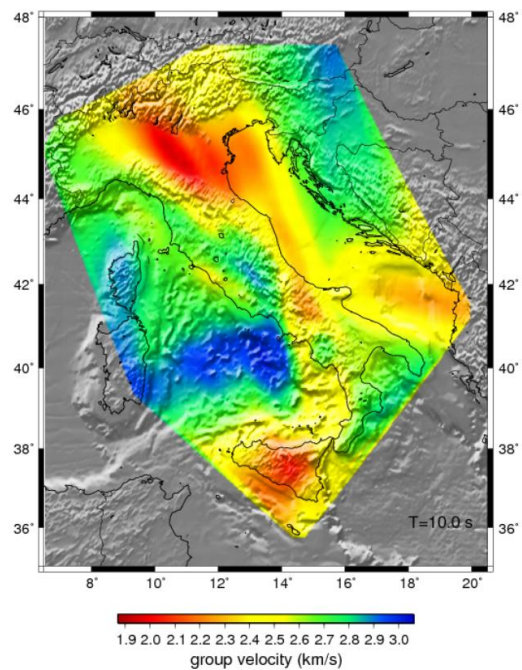


图 4.13 $T=10.0$ s 群速度层析成像图

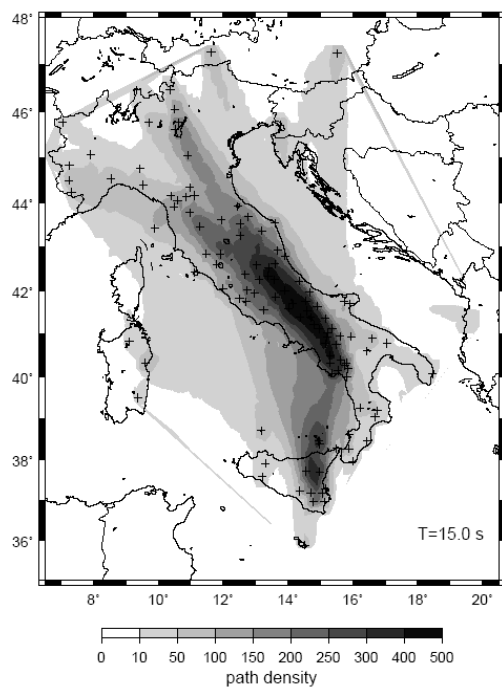


图 4.14 T=15.0 s 路径覆盖密度图

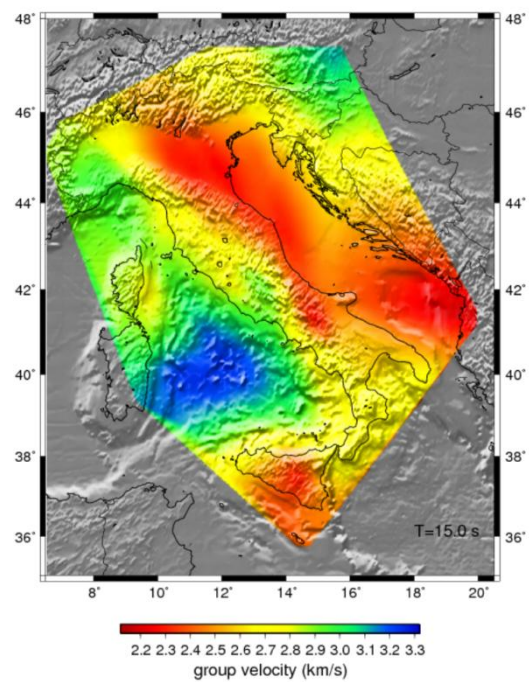


图 4.15 T=15.0 s 群速度层析成像图

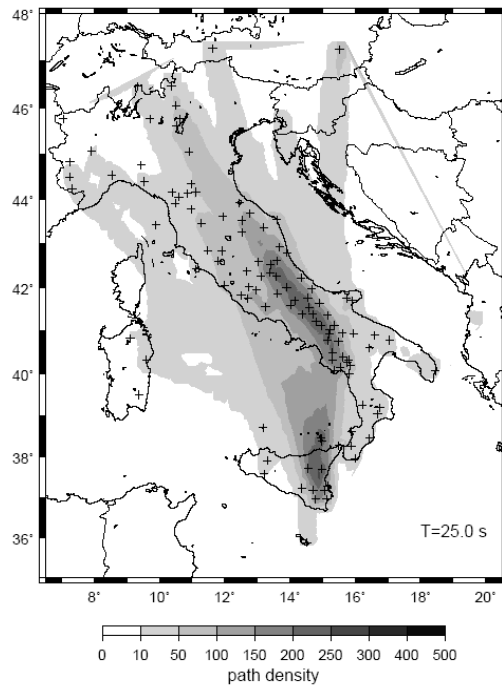


图 4.16 T=25.0 s 路径覆盖密度图

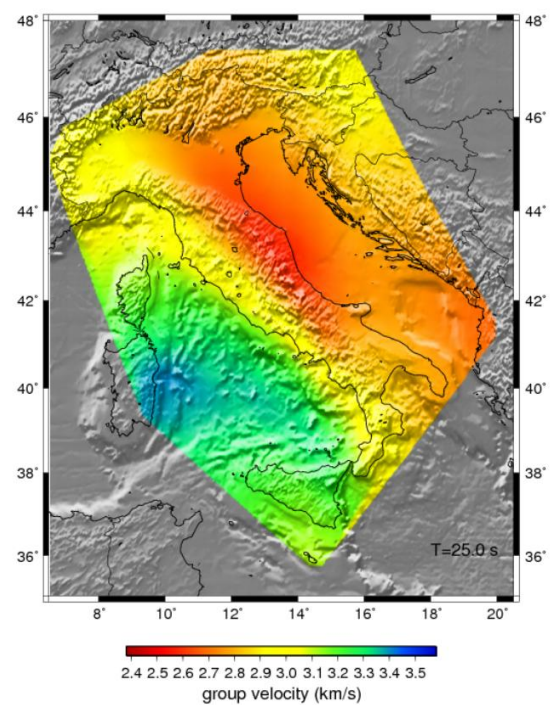


图 4.17 T=25.0 s 群速度层析成像图

5 结论

互相关数据处理方面,我发现时间域的归一化处理可以显著抑制噪声中的地震信号,而且滑动绝对平均法(Running absolute mean)和1比特法(Onebit)是效果最好的。

部分路径信号的叠加效果有明显的季节性差异,这可能是季节性的人工单频噪声源所致。意大利地区的背景噪声主要来自NW方向,可能跟英吉利海峡海浪对欧洲大陆西北岸的冲击有关。相比之下,来自波罗的海方向的噪声源要弱一些。

同时,NW方向的信号信噪比更高,这可能由于该方向上存在持续稳定的噪声源以及台站路径更多的分布在NW-SE方向。

层析成像反演方面,我们设定群速度的平均值为2.68 km/s,划分为 70×70 个小网格,网格大小为 0.2° 。相比以往的面波层析成像研究,我们的层析成像反演图达到了更高的精度和路径覆盖密度。

我们的结论与A. Pontevivo et al.大体一致,不同的是第勒尼安盆地(海)存在高速区(洋壳结构)而不是低速区,低速区南移至西西里岛($T=10, 15, 25s$),对应陆壳结构,这说明西西里岛和亚平宁半岛曾经是连在一起的。周期 $T=15s$ 的层析成像图非常清晰的显示了第勒尼安盆地高速区域的分布范围和特征并与盆地地形吻合较好。而在亚平宁半岛东侧的亚得里亚海为低速区,对应洋壳结构,与C. Chiarabba et al.的研究相符[12]。另一方面,我也不同意他们七个区域的划分。从我们在4.3节的讨论可以看出,意大利及其周边对应不同周期Rayleigh面波的低速区和高速区范围在不断变化中,所以7个区域的固定划分无法解释这种变化。

从 $T=6, 10, 15, 25s$ 的群速度层析成像图上可以看出,随着周期增加,Rayleigh面波透入深度增加,面波的群速度有增加的趋势。所以 $T=6s$ 的层析成像图与 $T=15s$ 的层析成像图相比,其速度的平均值显著减小。

总体上,地下介质群速度的划分与地表地质界限对应较好。亚得里亚海为陆壳结构并且沉积层较厚,所以在短周期周期($T < 20s$)为低速区。第勒尼安盆地属洋壳结构,地壳较薄,周期 $T \geq 15s$ 的Rayleigh面波可以探测到部分上地幔,故表现为高速。波河平原接近阿尔卑斯山,且沉积较厚,故在短周期和长周期层析成像图上均为低速区。周期 $T=15s$ 层析成像图的低速/高速区域分界线相对于北亚平宁弧有少量偏移,这由于探测深度范围主要是下地壳与上地幔,与地表情况并非完全一致,需进一步研究。

致 谢

感谢李老师的悉心指导和独到的实验安排。我从本科毕业设计中学到了很多东西，除了地震背景噪声的相关知识和处理流程，还包括熟悉使用 **Linux** 环境，**Bash shell** 编程，**SAC** 软件交互和宏编写，以及 **GNU C/C++** 的编写和调试，更重要的是我学到了科学的思维方法——如何从文章中提取有价值信息，如何借鉴他人的成果或程序而设计自己的风格，如何展开研究一个问题。

感谢大三时叶高峰老师的指导，自那时起我对 **Matlab** 和地球物理数据处理产生了浓厚的兴趣。

感谢学院为我们本科生作毕设提供的设备和良好的学习环境。同时感谢学校在我大三时提供的本科生创新实验项目机会，那次实验中我积累了很多科研经验。

参考文献

- [1] Campillo, M. & Paul, A., 2003. Long-range correlations in the diffuse seismic coda, *Science*, 299, 547–549.
- [2] Brillouin, Léon., Wave Propagation and Group Velocity. *Academic Press Inc.*, New York (1960).
- [3] Dltmar et al., Generalization of Backus Gilbert Method for estimation of lateral variations of surface wave vilocities. *Phys. Solid Earth, Izvestia Acad. Sci. U. S. S.R.* 1987, 470~477
- [4] A. Pontevivo et al., Group velocity tomography and regionalization in Italy and bordering areas, *Physics of the Earth and Planetary Interiors* 134 (2002) 1–15
- [5] Bensen el al., Processing seismic ambient noise data to obtain reliable broad-band surface wave dispersion measurements, *Geophys. J. Int.* (2007) 169, 1239–1260
- [6] Hongyi Li et al., Surface Wave Dispersion Measurements from Ambient Seismic Noise Analysis in Italy, *Geophys. J. Int.* (2000) 000, 000–000
- [7] Ellie Quigley, Unix Shells by Example fourth edition, *Pearson Education*, 2007
- [8] Peter Goldstein et al ,SAC Manual, <http://www.iris.edu/manuals/sac/manual.html>, the Regents of the University of California, 2009
- [9] Ellie Quigley, Unix Shells by Example, *Pearson Education Asia Ltd.*, 2007
- [10] Gerald Pfeifer et al., GCC Manual, <http://gcc.gnu.org/onlinedocs/gcc-4.4.0/gcc/>, *Free Software Foundation, Inc.*, 2009
- [11] P. Wessel et al., The Generic Mapping Tools, <http://gmt.soest.hawaii.edu/>, *University of Hawaii*, 2009
- [12] C. Chiarabba et al, A new view of Italian seismicity using 20 years of instrumental recordings, *Tectonophysics* 395 (2005) 251–268, 2005

附 录

脚本源代码（Bash shell, SAC 宏）

crosscorr.mac 作互相关运算的宏，参数为台站 A、B 的名称和下标范围

```
*** my sac macro begins here:
```

```
***
```

```
*      this is a simple script to test
```

```
***
```

```
$KEYS staA staB compon mydir lower upper
```

```
$default staA aaaa
```

```
$default staB bbbb
```

```
$default mydir /home/Bob/projects/test_SAZ0_d400/
```

```
$default compon z
```

```
setbb OUTPUTF sum_corr.$staA$.$staB$.files
```

```
setbb indice $lower
```

```
WHILE %indice LE $upper
```

```
setbb ff1 $staA$_%indice%.$compon
```

```
setbb ff2 $staB$_%indice%.$compon
```

```
read %ff1 %ff2
```

```
TRANSFER FROM ELMAG FREEP 15. MAG 750. TO NONE
```

```
rmean
```

```
rtrend
```

```
bp c 0.02 0.1
```

```
interpolate delta 0.1
```

```
whiten
```

```
correlate
```

```
write TMP.SAC CORR_$staA$.$staB$.%indice%.$compon$.sac
```

```
setbb indice (INTEGER (ADD %indice 1 0.5))
```

```
ENDDO
```

stacks.mac 作叠加运算，将两台站所有的小时互相关文件叠加成一个文件

```
$KEYS compon mydir lower upper
```

```
$default compon z
```

```
$default mydir /home/xinliu/projects/test_SAZ0_d400
```

```
$default lower 10
```

```
$default upper 350
```



```
setbb TAG CORR_aaaa.bbbb
setbb OUTPUTF sum_corr_aaaabbbb.$compon
sc rm -f *.sgf
* remove all the previous sgf files

SSS
*ENTER SIGNAL STACK SUBPROCEDURE~~~
**Purpose: DO FILE WILD DIR $mydir %TAG%*.$compon$*
setbb indice $lower
WHILE %indice LE $upper
setbb ff %TAG%.%indice%.$compon$.sac
*compon is the global variable defined in the first line
ADDSTACK %ff
ADDSTACK %ff reversed
setbb indice (INTEGER (ADD %indice 1 0.5))
ENDDO

TIMEWINDOW -200 200
SUMSTACK
writestack %OUTPUTF%.sac
QUITSUB
*QUIT THIS SUBPROCEDURE
r %OUTPUTF%.sac
*write %OUTPUTF%.sac
bd sgf
* start the SAC Graphics File device driver
pl
* plot it again to f001.sgf

sgftops f001.sgf %OUTPUTF%.ps
* convert it to postscript, done
sc rm -f *.sgf
* remove all the previous sgf files
sc gs %OUTPUTF%.ps
* view the outout eps file
```

myjobs.sh, 用于实现群速度信息的交互式半自动提取

```
#!/bin/bash
###
#   Automatically process group-velocity extraction
###
set -x
for file in `cat zz|egrep -v 'done|bad|error' |head -800`
do
```

```

do_mft $file # here just "do_mft"
echo "Yes or No? (Type y or n): "
read answer # read from the screen "y" or "n"
if [[ $answer == [yY] ]]; then
    echo Good path
    ./chnm.sh $file # change the file names of our results
    #gawk '{if ($1==ff) {print $1," done"} else {print $1}}' ff=$file $file
    gawk '{sub(aa,aa " done",$1); print $0}' aa=$file zz > tmp1
elif [[ $answer == [nN] ]]; then
    echo Bad path
    #gawk '{if ($1==ff) {print $1," bad"} else {print $1}}' ff=$file $file
    gawk '{sub(aa,aa " bad",$1); print $0}' aa=$file zz > tmp1
elif [[ $answer == [eE] ]]; then
    echo ERROR path
    gawk '{sub(aa,aa " error",$1); print $0}' aa=$file zz > tmp1
elif [[ $answer == q ]]; then
    echo quit
    break
fi
mv tmp1 zz
cp zz zz.bk
done

```

get_eachT.sh, 用于把每个周期对应的多条路径的群速度信息从**.dsp** 文件中收集起来, 放在以周期值命名的**.dat** 文件中

```

#!/bin/bash
#####
# This shell converts .dsp file to period (.dat) file @ Italy
## Input syntax: ./desp2period_file.sh 5 25
##                                     ,in which { 5 25 } defines the period window
#####
period=$1
maxperiod=$2
while (( $period <= $maxperiod ))
do
    for aa in `ls *.dsp`
    do
        gawk '{if ($5==periododd) {printf "%8s%9.3f%9.3f%9.3f%9.3f%
6.3f%6.3f%8.1f%6.1f%6.1f\n",substr($19,1,7),$11,$12,$13,$14,$6,$7,$8,$9,($9+180)%360} }'
periododd=$period $aa>> $period.dat
        echo doit!!
    done
    let period+=1
done

```

meanvelocity.sh, 用于求群速度最大值与最小值, 并调用 **./plt_tomo.sh** 绘图

```
#!/bin/bash
getmean () {
    echo getmean $1 $2
    input=$1
    output=$2
    lst=`gawk '{print $3}' $input`
    local max=0
    local min=10000
    for ff in $lst
    do
        max=`echo $ff $max|gawk '{if($1>$2) print $1; else print $2}'`
        min=`echo $ff $min|gawk '{if($1<$2) print $1; else print $2}'`
    done
    mean=`echo $max $min|gawk '{printf "%.5f",($1+$2)/2}'`
    delta=`echo $max $min|gawk '{printf "%.5f",($1-$2)/2+0.001}'`
    echo $input $mean $delta >> $output
}
flist=`ls t??mu1.dat|head -30`
outfile=meanvelocity.txt
mv $outfile ${outfile}.old
for fun in $flist
do
    getmean $fun $outfile
    echo $fun $mean
    ./plt_tomo.sh $fun $mean $delta
done
```

prog_2006.sh, 用于文件重命名并调用 **stacks_season.mac** 实现按季节叠加

```
#!/bin/bash
datadir=~/.projects/italy-data/CING_AMUR
workdir=`pwd`
cd ..
# here we select desired datafiles by using wildcard {*,?,...}
flist=`ls 2006*.stk` stapair=`echo $flist|head -1|gawk -F. '{print $2}'`
for aa in `echo $flist|grep $stapair`
do
    # -F indicates the Field Separator type: '.'
    # use PIPE, in -which 'sh' indicates the input of executable script
    # this line simply copy the data files to the workdir with eveid,etc. the file name.
    bb=$aa
    mon=`echo ${bb/2006/}|gawk -F. '{print $1}'`
```

```

mon=${mon/_0}
mon=${mon/_}
echo $aa|gawk -F. '{print "cp ",$0, dir "/"month"."$3"."$4".sac"}' dir=$workdir
month=$mon |sh
done
cd 2006
for component in `echo HHZ HHT HHR`
do
sac2008<<EOF>sac.log
echo on
qdp off
macro stacks_season.mac compon $component month1 1 month2 3 stapair $stapair
macro stacks_season.mac compon $component month1 4 month2 6 stapair $stapair
macro stacks_season.mac compon $component month1 7 month2 9 stapair $stapair
macro stacks_season.mac compon $component month1 10 month2 12 stapair $stapair
macro stacks_season.mac compon $component month1 1 month2 12 stapair $stapair
r stack_corr.2006*.${stapair}.${component}.sac
sgf NUMBER 1 OVERWRITE ON
p1
sgftops f001.sgf stack_corr_overlap.${stapair}.${component}.ps
sc rm -f *.sgf
quit
EOF
done
### end of my bash script~

```

stacks_season.mac SAC 宏，用于实现按季节叠加并绘图

```

$KEYS compon mydir lower upper month1 month2 stapair
$default compon HHZ
$default month1 1
$default month2 3
$default stapair DOI_SACS

setbb TAG stk
setbb OUTPUTF stack_corr.2006.$month1$~$month2$.$stapair$.$compon
sc rm -f *.sgf
* remove all the previous sgf files
SSS
*ENTER SIGNAL STACK SUBPROCEDURE~~~
setbb indice $month1

WHILE %indice LE $month2
setbb ff %indice%.$compon$.%TAG%.sac
*compon is the global variable defined in the first line

```

```
ADDSTACK %ff
ADDSTACK %ff reversed
setbb indice (INTEGER (ADD %indice 1 0.5))
* indice=indice+1
ENDDO
```

```
TIMEWINDOW -600 600
SUMSTACK
writestack %OUTPUTF%.sac
QUITSUB
r %OUTPUTF%.sac
ch allt -600
bp c 0.02 0.1
write %OUTPUTF%.sac
bd sgf
* start the SAC Graphics File device driver
sgf NUMBER 1 OVERWRITE ON
p1
* plot it again to f001.sgf
sgftops f001.sgf %OUTPUTF%.ps
* convert it to postscript, done
```

Oper_RMS_SNR.mac SAC 宏, 用于实现计算 7000 余条数据的 RMS 值和 SNR, 并输出至 RMS.txt 文件

```
$KEYS compon mydir lower upper maxt filename
$default compon HHZ
$default mydir /home/Bob/projects/AMUR_TERO/2006
$default lower 1
$default upper 20
$default maxt 500
$default filename RMS.txt
$default errfilename RMS_anomaly.txt
setbb minv 1.5
setbb maxv 4.2
setbb indice $lower
```

```
WHILE %indice LE $upper
setbb ff %indice%.sac
r %ff
*rmean
*rtr
setbb tw1 ( DIV &1,dist %maxv )
setbb tw2 ( DIV &1,dist %minv )
setbb tw3 ( &1,e - 50 )
```

```

IF %tw2 GT %tw3
setbb tw2 %tw3
ENDIF
*#to control if the signal window and the noise window overlaps
setbb tw4 &1,e
MTW %tw1 %tw2
rms to user0
*signal on positive lag
MTW %tw3 %tw4
rms to user1
*to environment variable user1
*noise on positive lag

setbb tw3 ( &1,b + 50 )
setbb tw4 &1,b
*cut o -#tw2 -#tw1
MTW -%tw2 -%tw1
rms to user2
*signal on neg lag
MTW %tw4 %tw3
rms to user3
*to environment variable user3
*noise on neg lag

w %ff
IF &1,user1 NE nan

setbb rat1 ( &1,user0 / &1,user1 )
setbb rat2 ( &1,user2 / &1,user3 )
TRANSCRIPT OPEN FILE $filename CONTENTS OUTPUT
MESSAGE " %indice &1,KSTNM &1,user0 &1,user1 %rat1  &1,user2 &1,user3 %rat2
&1,evla &1,evlo &1,stla &1,stlo &1,AZ "
TRANSCRIPT CLOSE
ELSE
TRANSCRIPT OPEN FILE $filename CONTENTS OUTPUT
MESSAGE " %indice &1,KSTNM &1,user0 &1,user1 nan &1,user2 &1,user3 nan &1,evla
&1,evlo &1,stla &1,stlo &1,AZ "
TRANSCRIPT CLOSE
ENDIF

setbb indice (INTEGER (ADD %indice 1 0.5))
ENDDO

```

plt_capshift.sh SAC 宏，用于绘制噪声源的方位性分布图

```
#!/bin/bash

set -x
#####
# ./plt_capshift.sh rms.dat {minvalue} {maxvalue}
# To plot the direction preference with colorscale

#####

md=$1
minval=$2
maxval=$3
# min 0.1;
# max 7.8.
AREA=-R6.5/20.0/35.5/48.0
AREA=-R6.5/20.0/35.5/48.0
makecpt -Cseis -Qo -T${minval}/${maxval}/2 -Z > shift.cpt
pscoast -JM6 ${AREA} -Ba2f1/a2f1WSne -Di -W5 -N1/2 -K -X1.2 -Y2 -P > ${md}/dat/ps}
psscale -Cshift.cpt -B2:"ratio": -S -D3.0i/-0.65i/4.0i/0.12ih -K -O>> ${md}/dat/ps}
gawk '{print $4,$5}' ${md} |psxy -JM ${AREA} -Sc0.1 -O -K -:>> ${md}/dat/ps}
gawk '{print $6,$7}' ${md} |psxy -JM ${AREA} -St0.12 -G0 -O -K -:>> ${md}/dat/ps}
gawk '{print "# -Z", $9"\n"$4,$5,"\n", $6,$7}' ${md} |psxy -JM ${AREA} -Cshift.cpt -O -K
-M# -:>> ${md}/dat/ps}
gawk '{print "# -Z", $9"\n"$4,$5,"\n", $6,$7}' ${md} |psxy -JM ${AREA} -Cshift.cpt -O -K
-M# -:>> ${md}/dat/ps}
gawk '{print "# -Z", $9"\n"$4,$5,"\n", $6,$7}' ${md} |psxy -JM ${AREA} -Cshift.cpt -O -M#
-:>> ${md}/dat/ps}
```

plt_SNR_rose.sh, bash shell 脚本，用于绘制信噪比 SNR 的方位角分布图

```
#!/bin/sh
#####
# usage: ./plt_SNR_rose.sh RMS_normal.txt
#####
rmsdata=$1
gawk '{print $3/($4),$13}' $rmsdata|psrose -A3r -R0/1/0/360 -S1.8n -G100 -W2
-Bg0.25/30g45nswe -X3 -Y4 -N> prms.ps
gawk '{print $6/($7),$13}' $rmsdata |psrose -A3r -R0/1/0/360 -S1.8n -G100 -W2
-Bg0.25/30g45 -X3 -Y4 -N> nrms.ps
gawk '{print ($3+$6)/($4+$7),$13}' $rmsdata|psrose -A3r -R0/1/0/360 -S1.8n -G100 -W2
-Bg0.25/30g45 -X3 -Y4 -N> avgrms.ps
```

C/C++源代码

runabs.c, 用于做滑动时间窗归一化（Running Absolute Mean）

```

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>
#include <unistd.h>

/* usage: runabsmean N input_rawdata input_earthquake extension_name_of_newf */
float *readfile(char *infile, int *npts);
int main (int argc, char *argv[]) {
    /******
    /* Variables */
    float Wn, sum=0;
    int N,nr,nrb,i; //N: the width of the window
    int *npts = NULL;
    int *npts2 = NULL;
    char *infile = NULL;
    char *infile2 = NULL;
    char *extnm = NULL;
    float *dato = NULL;
    FILE *f = NULL;
    char mycm1[2048]; /* change to 2048, then no error message */
    char mycm2[2048];
    char mycm3[2048];
    char tmpf[1024];
    char outfile[1024];
    float *inn=NULL; // input raw data data
    float *inneq=NULL; // input earthquake data
    float *out=NULL; // ouput data which are normailized in time domain

    /******
    /* Read arguments: N sac_raw_data sac_earthquake_file extension_name_of_new_file */
    if (argc != 5) {
        fprintf(stderr, "Usage:runabsmean N sac_raw_data sac_earthquake_file
extension_name_of_new_file\n");
        return(0);
    }

    //allocate memory for arguments variables
    infile = (char *) malloc((strlen(argv[2])+1)*sizeof(char));
    infile2 = (char *) malloc((strlen(argv[3])+1)*sizeof(char));
    extnm = (char *) malloc((strlen(argv[4])+1)*sizeof(char));
    dato = (float *) malloc(sizeof(float));

```



```

npts  = (int *)   malloc(sizeof(int));
npts2  = (int *)   malloc(sizeof(int));
/*****/
/* Read values of arguments                                     */
N=atoi(argv[1]); // read the width N of the window ; atoi(char *) converts string to number
strcpy(infile, argv[2]);      // read sac bin file
strcpy(outfile, argv[2]);
strcpy(tmpf, argv[2]);
strcpy(infile2, argv[3]);     // read sac earthquake file
strcpy(extnm, argv[4]);      // read the extension name
// copy the original data file to output file (not modified)
strcat(strcat(outfile, "_"), extnm);
strcat(mycml, "cp ");
strcat(mycml, infile);
strcat(strcat(mycml, " "), outfile);
fprintf(stderr, "Copy infile to temporary output file: %s\n", mycml);
system(mycml);
/// read data files and store the data in temporary arrays.
    inn=readfile(infile, npts);
    inneq=readfile(infile2, npts2);
    if((*npts) != (*npts2)) {
        fprintf(stderr, "npts (number of points) not equal in 2 input
files!!\n %d,%d,%s", *npts, *npts2, infile);
        return(0);
    }
    out=(float *)calloc(*npts, sizeof(float));
/* Compute Running Abs Mean temporal normalizat                                     */
int st=0, ed=0, pst=0, ped=0; // start and end pointer of the window
int j;
i=0;
st=0; ed=i+N; sum=0;
for(j=st; j<=ed; j++) {
    sum+=fabs(inneq[j]);
}
Wn=sum/(ed-st+1); // according to the definition of Wn=sum(abs(dj))/(2N+1)
out[i]=inn[i]/Wn; // according to the definition of normalized  $\tilde{d}n = dn/Wn$ 
pst=st; ped=ed; // store the previous window location
for(i=1; i<*npts; i++) {
    if(i-N<0)
        st=0;
    else
        st=i-N;
    if(i+N>*npts-1)
        ed=*npts-1; // *npts-1 point to the last element in array inn[]

```

```

else
    ed=i+N;
    if((st-pst)==1) sum-=fabs(inneq[pst]); //subtract the deleted element from sum
    if((ed-ped)==1) sum+=fabs(inneq[ed]); // add the income element to sum
    Wn=sum/(ed-st+1); // according to the definition of  $W_n = \text{sum}(\text{abs}(d_j)) / (2N+1)$ 
    out[i]=inn[i]/Wn; //according to the definition of normalized  $dn^{\sim} = dn/W_n$ 
    pst=st; ped=ed;
}
/* Write normalized values over sac */
f=fopen(outfile, "r+b");
dato=out; // locate the pointer dato to the same address in memory as out
for(nr=158; nr<*npts+158; nr++) {
    nrb=4*nr;
    fseek(f, nrb, SEEK_SET);
    fwrite(dato, 4, 1, f);
    dato++;
}
fclose(f);
free(infile);
free(extnm);
free(npts);
return(0);
}

float *readfile(char *infile, int *npts) { // read sac binary data and return the value
/* Open infile, read npts and data */
int nr, nrb;
FILE *f=NULL;
f=fopen(infile, "rb"); //f=fopen(infile, "r+b");
nr=79;
nrb=4*nr;
fseek(f, nrb, SEEK_SET);
fread(npts, 4, 1, f);
float *inn=NULL;
float *dato = NULL;
inn=(float *)calloc(*npts, sizeof(float));
dato=inn;
for(nr=158; nr<*npts+158; nr++) {
    nrb=4*nr;
    fseek(f, nrb, SEEK_SET);
    fread(dato, 4, 1, f);
    dato++;
}
/* Close file and free malloc */
fclose(f);

```

```

    return inn;
}

```

dateconv, C++程序, 用于年月日与儒略日的转换, 包括 5 个源文件(版面原因, 只列出 3 个)

1. main.c

```

#include "mytool1.h"
#include "mytool2.h"
#include <iostream>
#include <string>
using namespace std;
int main(int argc, char *argv[]) {
    if(argc==1) {
        cout<<"Input Argument Error: please refer to README or HELP"<<
        endl<<"usage: dateconv -A {date}"<<
        endl<<"\t-A could be:"<<
        endl<<"\t\t-2j\tconvert from month day to julia day YYYYDDD,"<<
        endl<<"\t\t\tinput date should be formulated in YYYYMMDD"<<
        endl<<"\t\t-2d\tconvert from julia day to month day YYYYMMDD,"<<
        endl<<"\t\t\tinput date should be formulated in YYYYDDD"<<
        endl<<"Example 1: ./dateconv -2j 20080120"<<
        endl<<"output: 2008020"<<
        endl<<"Example 2: ./dateconv -2d 2008120"<<
        endl<<"output: 20080429"<<endl;
        return 0;
    }
    if(strlen(argv[1])<=6) { // the command include a delimiter
        if(!strcmp(argv[1], "-2j")) { //convert to julia day
            //cout<<"print juliaday!"<<endl;
            juliaday ju(argv[2]);
            ju.printjuliaday();
        }
        else
            if(!strcmp(argv[1], "-2d")) { //convert to month day~aargv[1]=="-2j"
                //cout<<"print monthday!"<<endl;
                monthday mu(argv[2]);
                mu.printmonthday();
            }
    }
    else { // the command doesn't have a delimiter
        //cout<<"default conversion: "<<endl;
    }
}

```

```
juliaday ju(argv[1]);  
ju.printjuliaday();  
}  
return 0;  
}
```

2. mytool1.cpp

```
#include "mytool1.h"  
#include <iostream>  
#include <string>  
#include <cstdlib> //we use std::atoi to convert  
  
using namespace std;  
  
int juliaday::printjuliaday()  
{  
    juliadays=0;  
    int twmonth[]={31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};  
  
    if((year%4==0 && year%100!=0) || (year%100==0 && year%400==0))  
        twmonth[2-1]+=1;  
    for(int i=0;i<month-1;i++){  
        juliadays+=twmonth[i];  
    }  
    juliadays+=day;  
    char str_1[8];  
    sprintf(str_1, "%4d%03d", year, juliadays);  
    //cout<<"%4d%3d"<<year<<juliadays<<endl;  
    cout<<str_1<<endl;  
    //cout<<year<<month<<day;  
  
    return 0;  
}
```

3. mytool1.h

```
#ifndef _MYTOOL_1_H  
#define _MYTOOL_1_H  
#include <iostream>  
#include <string>  
#include <cstdlib>  
  
using namespace std;  
class juliaday{
```

```

int year, month, day;
int juliadays;
public:
juliaday() {
    year=1000;
    month=10;
    day=1; juliadays=0;
}
juliaday(string stddate) {
    year=std::atoi(stddate.substr(0,4).c_str());
    month=std::atoi(stddate.substr(4,2).c_str());
    day=std::atoi(stddate.substr(6,2).c_str()); juliadays=0;
}
int printjuliaday();//the definition of this function is in mytool.cpp
};
#endif

```

4. mytool2.cpp
因篇幅原因，略。
5. mytool2.h
因篇幅原因，略。

Matlab 源代码

1. plotmap.m 主函数，绘图

```

importfile('RMS_normal.txt')
cel_eve=[dbseis.eve num2cell(dbseis.evelo) num2cell(dbseis.evela)];
cel_sta=[dbseis.sta num2cell(dbseis.stalo) num2cell(dbseis.stala)];
%% this two command applied the database functions
insert2db
getunique
%% make paths database
cel_paths=[num2cell(dbseis.ID),dbseis.eve,dbseis.sta,num2cell(dbseis.
posleg),num2cell(dbseis.negleg),num2cell(dbseis.posleg./dbseis.negleg
),num2cell(dbseis.azimuth)];
mkpathsdb

%% Ready to plot
clear
paths_pos_retri %retrieve paths and store in e struct
paths2draw=e.Data;
pathsnum=cell2mat(paths2draw(:,4:9));
clear s conn e %paths2draw

```

```

figure(2)
hold on
sizedata=size(pathsnum);
logratio=log(pathsnum(:,5));
for j=1:sizedata(1)
    if(pathsnum(j,6)>180) % if the azimuth ranges from 180 to 360
        logratio(j)=-logratio(j);
    end
end
minrms=min(logratio);
maxrms=max(logratio);
level=256; %colorlevel
cmp=colormap(jet(level));
delta=(maxrms-minrms)/level;
for i=1:sizedata(1)
    indcol=mod(fix((logratio(i)-minrms)/delta),256);
    if(indcol==0) indcol=1; end
plot([pathsnum(i,1);pathsnum(i,3)],[pathsnum(i,2);pathsnum(i,4)],'color',cmp(indcol,:))
end
ylim([35 48])
xlim([6.5 20.5])

```

2. importfile.m 从 RMS.txt 导入原始数据,存储于元胞数组(Cell array) 因篇幅原因,略。

3. insert2db.m 插入 7436 个数据到 Italy1.mdb

```

% Make connection to database. Note that the password has been omitted.
% Using ODBC driver.
conn = database('italia','','password');
e = exec(conn,'DELETE ALL * FROM stations WHERE ID > 0');
% INSERT 7000+ data records to database.
insert(conn,'stations',{'station','lo','la'},cel_eve)
insert(conn,'stations',{'station','lo','la'},cel_sta)
close(e)
% Close database connection.
close(conn)

```

4. getunique.m 选择保证唯一性的台站信息,包括台站名称,经度,纬度

```

conn = database('italia','','password');
% Read data from database.

```

```

e = exec(conn, 'SELECT DISTINCT station, lo, la FROM stations');
e = fetch(e);
close(e)
% Close database connection.
close(conn)
raw_sta_loc=e.Data;
sizeraw=size(raw_sta_loc);
conn1 = database('italia2', '', 'password');
e1 = exec(conn1, 'DELETE ALL * FROM stations WHERE ID > 0');
pt=sizeraw(1);
for i=1:sizeraw(1)
    if(~strcmp(raw_sta_loc{pt,1},raw_sta_loc{i,1}))
        insert(conn1, 'stations', {'station', 'lo', 'la'}, raw_sta_lo
oc(i,:))
    end
    pt=i;
end
close(conn1)
clear conn conn1 e1 e sizeraw

```

5. mkpathsdb.m 删除冗余经纬度信息，建立新关系表 paths

```

% Using ODBC driver.
conn = database('italia2', '', 'password');
e1 = exec(conn, 'DELETE ALL * FROM paths WHERE pID > 0');
% Write data to database.
insert(conn, 'paths', {'pID', 'eve', 'sta', 'posleg', 'negleg', 'ratio',
'azimuth'}, cel_paths)
% Close database connection.
close(conn)
clear e1 conn

```

6. paths_pos_retri.m 连接查询，重建绘图作序的数据表

```

% Make connection to database. Note that the password has been omitted.
% Using ODBC driver.
conn = database('italia2', '', 'password');
% Read data from database via Join query
e = exec(conn, 'SELECT ALL pID, eve, sta, stations.lo, stations.la, ...
stations_1.lo, stations_1.la, ratio, azimuth FROM querypaths');
e = fetch(e);
close(e)
% Close database connection.
close(conn)

```