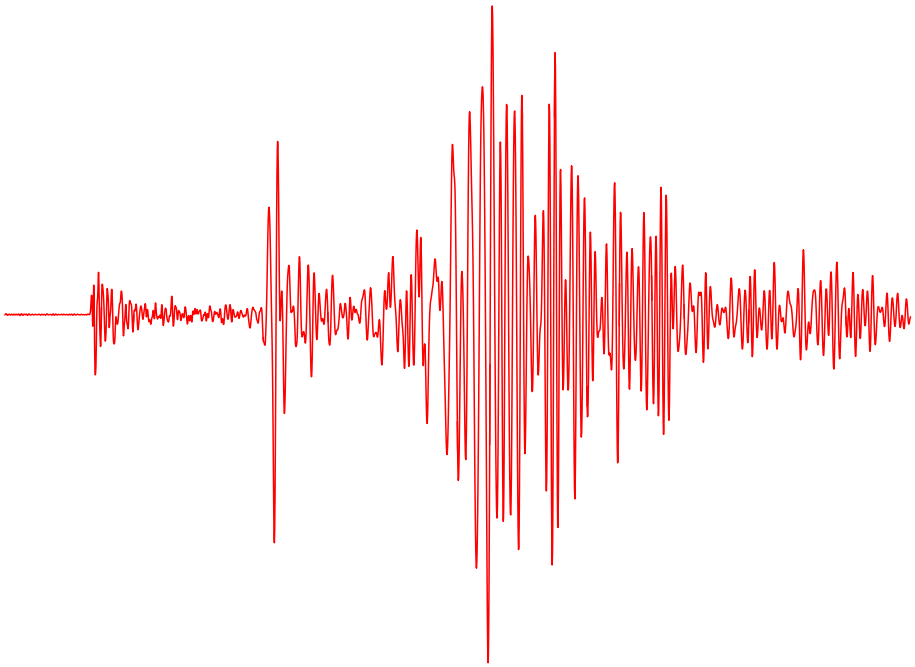

SAC 参考手册

基于 SAC v101.6a

作者： SeisMan

版本： 3.5

日期： 2016-01-09

写在第三版前的一些废话

> “工欲善其事，必先利其器。
>
> —《论语 · 卫灵公》”

2010 年 10 月，大三，开始接触并学习 SAC；2011 年的暑假开始着手 SAC 文档的翻译工作；2012 年 01 月，文档的 v1.0 版本发布；2013 年 03 月，文档的 v2.0 版发布。3 年多的时间过去了，文档更新到了 v3.0 版。

v2.0 大体上算是官方英文文档的译本，整体结构上完全遵循了官方文档的风格。整个文档的条理不够清晰，教程部分稍显单薄，命令部分也不够完善。

2013 年 11 月，George Helffrich 著的 “*The Seismic Analysic Code: A Primer and User's Guide*” 一书出版了。该书基于 MacSAC，与本文档所关注的 SAC 有一些区别，但核心部分是一致的。v3.0 版借鉴了该书的整体结构和部分内容，重新设计了文档结构并重写了教程的大部分内容，希望能够有一个结构更清晰、内容更丰富的版本。

整个文档分为教程部分和命令部分。教程部分又分为如下几章：

SAC 简介 SAC 软件的相关信息

SAC 基础 继续阅读手册所需的基础知识

SAC 文件格式 SAC 文件格式及 SAC 头段变量

SAC 数据处理 利用 SAC 命令进行地震数据处理和分析

SAC 图像 用 SAC 绘制地震波形，并控制绘图的细节

SAC 编程 利用 SAC 的宏编程功能实现数据批处理

SAC 与脚本语言 脚本 (Bash、Perl 和 Python) 中调用 SAC

SAC 函数库 C、Fortran 程序中调用 SAC 提供的库函数

SAC I/O C、Fortran、Matlab 和 Python 中实现 SAC I/O

SAC 相关工具 SAC 相关工具

SAC 技巧与陷阱 使用 SAC 时的若干技巧与陷阱

本文档的源码开源托管在 GitHub 上，欢迎更多的 SAC 用户参与到该项目，共同维护并完善本文档。

本文档仅供个人学习使用，希望不涉及版权问题。

个人博客：http://seisman.info
项目主页：https://github.com/seisman/SAC_Docs_zh
联系方式：seisman.info@gmail.com
文档发布及更新：http://seisman.info/sac-manual.html
捐赠页面：http://seisman.info/donations.html

作者：SeisMan
2014 年 04 月 14 日

第 3 章　SAC 文件格式

3.1　SAC 格式简介

一个地震波形数据包含了时间上连续的一系列数据点，数据点可以是等间隔或不等间隔采样。SAC 的数据格式要求一个文件中只包含一个地震波形数据，这样的定义更适合单个地震波形的处理。

每个 SAC 文件包含两个部分：一个头段区和一个数据区。

头段区位于每个文件的起始处，其大小是固定的，用于描述数据的相关信息，比如数据点数、采样周期等等。

数据区紧跟在头段区之后，数据区又包含了一个或多个子数据区：

- 如果数据是时间序列，且是等间隔采样的，则只有一个子数据区，包含因变量（Y，也就是数据）的值，因变量（X）的信息可以直接从头段区中获得；
- 如果数据是时间序列，但是不等间隔采样的，则有两个子数据区，分别包含因变量（Y）和自变量（X）的值；
- 如果数据是谱数据而非时间序列，则有两个子数据区，分别包含振幅和相位或者实部和虚部；
- 如果数据是三维数据（XYZ），则包含 `nxsize*nysize` 个子数据区。

最经常使用的数据是等间隔采样的数据，即文件中只有一个头段区和一个数据区。

3.2　两种数据形式

SAC 文件格式有两种形式：二进制型和字符型[1]。字符型与二进制型是完全等价的，只是字符型是给人看的，二进制型是给机器读写的。从 C 程序的角度来看，两者的区别在于，写文件时前者使用 `fprintf` 后者使用 `fwrite`。

二进制型的 SAC 数据，占用更小的磁盘空间，读写速度更快，因而是最常用的 SAC 格式形式。当文件出现问题时，字符型数据便于查看文件内容。

通常，字符型的 SAC 文件以 `ASC`[2]结尾，二进制型的 SAC 文件以后缀 `SAC` 结尾。但是，SAC 在读取文件时是不判断文件后缀的，所以文件后缀是什么并不重要，在某些情况下，会以 `BHE`、`BHN`、`BHZ` 这样的后缀结尾。在下面的示例中，很多 SAC 文件甚至没有后缀。

3.2.1　两种形式的互相转换

你是否想要一个字符型的 SAC 文件，用编辑器打开好好看看 SAC 数据究竟长什么样？SAC 自带的命令可以实现两种形式的转换[3]。

[1] 原为 alphanumeric，译为文数字。

[2] ASCII 的简写。

[3] 也可以使用 SAC 的 `convert` 命令进行转换，不过此命令即将被淘汰，因而这里不会介绍。

```
SAC> fg seis
SAC> w seis          // 先生成一个二进制型SAC数据，以做测试
```

将二进制型转换成字符型：

```
SAC> r seis          // 读二进制型文件
SAC> w alpha seis.a  // 以字符型写入
```

将字符型转换成二进制型：

```
SAC> r alpha seis.a  // 读字符型文件
SAC> w sac seis.b    // 以二进制型写入，可以省略sac，写成w seis.b
```

试试用你最喜欢的文本编辑器打开字符型的 seis.a 吧，其内容如下：

```
     0.01000000      -1.569280       1.520640      -12345.00      -12345.00
       9.459999       19.45000      -41.43000       10.46400      -12345.00
      -12345.00      -12345.00      -12345.00      -12345.00      -12345.00
      -12345.00      -12345.00      -12345.00      -12345.00      -12345.00
      -12345.00      -12345.00      -12345.00      -12345.00      -12345.00
      -12345.00      -12345.00      -12345.00      -12345.00      -12345.00
      -12345.00       48.00000      -120.0000      -12345.00      -12345.00
       48.00000      -125.0000      -12345.00       15.00000      -12345.00
      -12345.00      -12345.00      -12345.00      -12345.00      -12345.00
      -12345.00      -12345.00      -12345.00      -12345.00      -12345.00
       373.0627       88.14721       271.8528       3.357465      -12345.00
      -12345.00    -0.09854718       0.000000       0.000000      -12345.00
      -12345.00      -12345.00      -12345.00      -12345.00      -12345.00
      -12345.00      -12345.00      -12345.00      -12345.00      -12345.00
       1981          88             10             38             14
          0           6              0              0           1000
    -12345        -12345         -12345         -12345         -12345
          1          50              9         -12345         -12345
    -12345        -12345             42         -12345         -12345
    -12345        -12345         -12345         -12345         -12345
    -12345        -12345         -12345         -12345         -12345
          1           1              1              1              0
CDV       K8108838
-12345  -12345  -12345
-12345  -12345  -12345
-12345  -12345  -12345
-12345  -12345  -12345
-12345  -12345  -12345
-12345  -12345  -12345
-12345  -12345  -12345
    -0.09728001    -0.09728001    -0.09856002    -0.09856002    -0.09728001
    -0.09600000    -0.09472002    -0.09344001    -0.09344001    -0.09344001
```

```
33      -0.09344001     -0.09344001     -0.09472002     -0.09472002     -0.09344001
34      ......
```

第 1–30 行是头段区，31 之后的行是数据区。目前你可能还看不懂头段区的这些数字或者字符代表什么。没关系，在下一节会详细介绍 SAC 头段区，记得一定要一边看下一节的内容，一边对照着这个例子，好好琢磨每一个头段的含义。

3.3　SAC 头段结构

SAC 头段区包含了一系列头段变量，从这些头段变量中可以了解到波形记录的很多信息，比如台站经纬度、发震时刻、震相到时等等。表 3.1 列出了 SAC 头段区的全部头段变量。

表 3.1: SAC 头段变量列表

Byte	Type	Names				
0	F	delta	depmin	depmax	scale	odelta
20	F	b	e	o	a	internal
40	F	t0	t1	t2	t3	t4
60	F	t5	t6	t7	t8	t9
80	F	f	resp0	resp1	resp2	resp3
100	F	resp4	resp5	resp6	resp7	resp8
120	F	resp9	stla	stlo	stel	stdp
140	F	evla	evlo	evel	evdp	mag
160	F	user0	user1	user2	user3	user4
180	F	user5	user6	user7	user8	user9
200	F	dist	az	baz	gcarc	internal
220	F	internal	depmen	cmpaz	cmpinc	xminimum
240	F	xmaximum	yminimum	ymaximum	unused	unused
260	F	unused	unused	unused	unused	unused
280	N	nzyear	nzjday	nzhour	nzmin	nzsec
300	N	nzmsec	nvhdr	norid	nevid	npts
320	N	internal	nwfid	nxsize	nysize	unused
340	I	iftype	idep	iztype	unused	iinst
360	I	istreg	ievreg	ievtyp	iqual	isynth
380	I	imagtyp	imagsrc	unused	unused	unused
400	I	unused	unused	unused	unused	unused
420	L	leven	lpspol	lovrok	lcalda	unused
440	K	kstnm	kevnm*			
464	K	khole	ko	ka		
488	K	kt0	kt1	kt2		
512	K	kt3	kt4	kt5		
536	K	kt6	kt7	kt8		
560	K	kt9	kf	kuser0		
584	K	kuser1	kuser2	kcmpnm		
608	K	knetwk	kdatrd	kinst		

整个头段区，共有头段变量 133 个，占 632 个字节。头段区的前四个字节是第一个头段变量 delta，第 5–8 个字节是第二个头段变量 depmin，第 21–24 个字节是第 6 个头段变量 b，以此类推。

表的第一列给出了当前行的第一个头段变量在文件中的起始字节，第二列给出了当前行的头段变量的变量类型。

表 3.2 列出了 SAC 头段中的头段变量类型及其相关信息。第一列为头段变量类型代码，第二类给出了其代表的头段变量类型，第三列指出 C 源码中该变量的是用什么类型定义的，第四列给出了每个变量所占据的字节数，第五列给出了写字符型 SAC 文件时的输出格式，最后一列则给出该类型的未定义值。

表 3.2: 变量类型说明

Code	Type	C Type	sizeof	printf	未定义值
F	浮点型	float	4	%15.7f	-12345.0
N	整型	int	4	%10d	-12345
I	枚举型	int	4	%10d	-12345
L	逻辑型	int	4	%10d	FALSE
K	字符型	char*	8	%-8.8s	"-12345␣␣"
A	辅助型				

说明：

- 所有头段变量的变量名均以变量类型码开头，比如 nvhdr 是 N 型变量，leven 是 L 型变量，但 F 型变量除外；
- 枚举型变量本质上是 int 型，只能在固定的几个值中取值
- 逻辑型变量可以取值 TRUE 和 FALSE，本质上分别是整型的 1 和 0
- 字符型变量长度为 8[1]，只有 kevnm 很特殊，其长度为 16；
- 变量名为 internal 表示该变量是 SAC 内部使用的头段变量，用户不可对其进行操作；
- 变量名为 unused 表示该变量尚未使用，为以后可能出现的新头段变量占位；
- 当某个头段变量未定义时，其包含未定义值；不同类型的头段变量有不同的未定义值；若一个整型头段变量的值为 -12345，则认为该变量未定义；实际使用时，可以直接用 undef 表示所有类型的头段变量的未定义值，SAC 会根据头段变量的类型自动将其转换成相应类型的未定义值。
- 辅助型变量并不在 SAC 头段区中，而是从其它头段变量推导得到的；

3.4 SAC 头段变量

3.4.1 基本变量

nvhdr*

SAC 头段版本号。nvhdr[2]是 SAC 中很重要但是不太常用的头段变量。目前版本号为 6，旧版本的 SAC 文件（nvhdr<6）在读入时头段区会自动更新。

[1] C 语言中用 “\0” 作为字符串的结束标识符，因而源码中变量的实际长度为 9。

[2] 星号表示该头段变量在 SAC 中必须有定义值，下同。

nzyear, nzjday, nzhour, nzmin, nzsec, nzmsec

分别表示 “年”、“一年的第几天”[3][4]、“时”、“分”、“秒”、“毫秒”[5]。这六个头段变量构成了 SAC 中唯一的绝对时刻，SAC 中的其它时刻都被转换为相对于该时刻的相对时间（单位为秒）。关于 SAC 中的绝对时间和相对时间的概念，参考 “SAC 中的时间概念” 一节。

根据这六个头段变量还可以推导出其它一些辅助型头段变量：

- `kzdate`：字符数字格式的参考日期，由 `nzyear` 和 `nzjday` 导出
- `kztime`：字符数字格式的参考时间，由 `nzhour`、`nzmin`、`nzsec`、`nzmsec` 导出

如下例所示：

```
SAC> fg seis
SAC> lh nzyear nzjday nzhour nzmin nzsec nzmsec

    nzyear = 1981
    nzjday = 88
    nzhour = 10
     nzmin = 38
     nzsec = 14
    nzmsec = 0
SAC> lh kzdate kztime

    kzdate = MAR 29 (088), 1981
    kztime = 10:38:14.000
```

iztype

等效参考时刻。SAC 的参考时刻是可以任意指定的，但一般选取某个特定的时刻（比如文件起始时刻、发震时刻等等）作为参考时刻。其可以取如下枚举值 [1]：

- `IUNKN`：未知
- `IB`：以文件开始时刻为参考时间
- `IDAY`：以参考日期当天的午夜作为参考时间
- `IO`：以事件发生时间为参考时间
- `IA`：以初动到时为参考时间
- `ITn`：以用户自定义的时间 `Tn` 为参考时间（n 可取 0–9）

若 `iztype=IO`，则表示数据以发震时刻作为参考时刻，此时头段变量 `o` 的值应为 0。

iftype*

SAC 文件类型，其决定了头段区之后有几个子数据区。可以取如下枚举值：

- `ITIME`：时间序列文件（即 Y 数据，一般的地震波形数据）
- `IRLIM`：频谱文件（实部-虚部格式）
- `IAMPH`：频谱文件（振幅-相位格式）
- `IXY`：一般的 X-Y 数据

[3] 使用 jday 而不是 “month+day” 可以少用一个头段变量。

[4] 1 月 1 日对应的 `nzjday` 是 1 而不是 0。

[5] 1 s = 1000 ms

[1] 枚举型在 C 源码中使用 `#define` 宏来定义的，比如 `#define IO 11`，所有可取的枚举值都以字母 I 开头。

- IXYZ: 一般的 XYZ (3D) 文件

idep

因变量 (Y) 类型, 该头段变量可以不定义, 其可以取如下枚举值:

- IUNKN: 未知类型
- IDISP: 位移量, 单位为 nm
- IVEL: 速度量, 单位为 nm/s
- IVOLTS: 速度量, 单位为 V[1]
- IACC: 加速度量: 单位为 nm/s^2

3.4.2 数据相关变量

npts*

数据点数, 其值决定了在数据区有多少个数据点。

delta*

等间隔数据的数据点采样周期 (标称值)。

odelta

采样周期的实际值, 若实际值与标称值不同则有值, 一般来说都是未定义的。

b*, e*

文件的起始时间和结束时间 (相对于参考时刻的秒数)。

leven*

若数据为等间隔则为 TRUE, 否则为 FALSE。

depmin, depmax, depmen

因变量 (Y) 的最小值、最大值和均值。

在读入 SAC 文件以及对数据进行处理时, 这三个头段变量的值会被自动计算并更新。示例如下:

```
$ sac
SAC> fg seis
SAC> lh depmax
    depmax = 1.520640e+00  // 最大值
SAC> ch depmax 1000        // 强行修改数据最大值
                           // 这是错误的示范, 不要这样做
SAC> lh depmax 1000        // 查看depmax, 修改成功
    depmax = 1.000000e+03
SAC> w seis.SAC            // 写到磁盘中
SAC> q
$ saclst depmax f seis.SAC // 调用saclst查看磁盘文件中的depmax
seis.SAC       1000        // 可以看到磁盘中的文件depmax=1000
$ sac
SAC> r ./seis.SAC          // 读入SAC
SAC> lh depmax
    depmax = 1.520640e+00  // 此时depmax被自动计算并更新
```

[1] 不解

scale

因变量比例因子，即真实物理场被乘以该比例因子而得到现有数据。

假设真实物理场的 Y 值大概在 10^{-20} 量级，由于数据量级太小处理起来可能不太方便。此时可以将数据乘以 10^{20} 变成合适的量级，并修改 `scale=1.0e20`，这样就可以知道自己对数据人为放大了多少倍。

101.5 之前的版本中，在使用 `transfer` 命令去仪器响应时，若 `scale` 的值有定义，则输出的数据会根据该值进行放大并修改 `scale`。在 101.5 及其之后的版本中，`scale` 被忽略。

xminimum, xmaximum, yminimum, ymaximum

仅用于 3D (XYZ) 文件中，记录 X 和 Y 的最小/大值。

nxsize, nysize

仅用于 3D (XYZ) 文件中，表示 X 和 Y 方向的数据点数。

iqual†

iqual[1]标识数据质量，可取如下值：

- `IGOOD`：高质量数据
- `IGLCH`：数据中有毛刺 (glitches)
- `IDROP`：数据有丢失 (dropouts)
- `ILOWSN`：低信噪比数据
- `IOTHER`：其它

isynth†

合成数地震图标识。

- `IRLDTA`：真实数据

3.4.3　事件相关变量

kevnm

事件名，长度为 16 个字节。

evla, evlo, evel, evdp

分别代表事件的纬度 (-90 到 90)、经度 (-180 到 180)、高程 (单位为 m，未使用) 和深度 (单位为 km，以前为 m)。

ievreg†

事件地理区域[2]。

ievtyp

事件类型，这里仅列出部分常见的枚举值：

- `IUNKN`：未知事件
- `INUCL`：核事件
- `IEQ`：地震
- `IOTHER`：其它

[1] † 标识仅表示 SAC 程序内部未使用该头段变量，即变量有值或者无值、有何值，对于程序的运行不会产生任何影响，但用户可以在自己的程序中自由使用这些头段变量。下同。

[2] Flinn-Engdahl Regions:http://en.wikipedia.org/wiki/Flinn-Engdahl_regions

mag

事件震级。

imagsrc

震级信息来源, 可以取如下枚举值:

- INEIC: http://earthquake.usgs.gov/earthquakes/search/
- IPDE: http://earthquake.usgs.gov/data/pde.php
- IISC: http://www.isc.ac.uk/iscbulletin/search/catalogue/
- IREB: 人工检查过的事件目录
- IUSGS: USGS
- IBRK: UC Berkeley
- ICALTECH: California Institute of Technology
- ILLNL: Lawrence Livermore National Laboratory
- IEVLOC: Event Location
- IJSOP: Joint Seismic Observation Program
- IUSER: The individual using SAC2000
- IUNKNOWN: 未知

imagtyp

震级类型, 取如下枚举值:

- IMB: 体波震级
- IMS: 面波震级
- IML: 区域震级
- IMW: 矩震级
- IMD: 持续时间震级
- IMX: 用户自定义震级

gcarc, dist, az, baz

- gcarc: 全称 Great Circle Arc, 即震中到台站的大圆弧的长度, 单位为度;
- dist: 震中到台站的距离, 单位为 km;
- az: 方位角, 震中到台站的连线与地理北向的夹角;
- baz: 反方位角, 台站到震中的连线与地理北向的夹角。

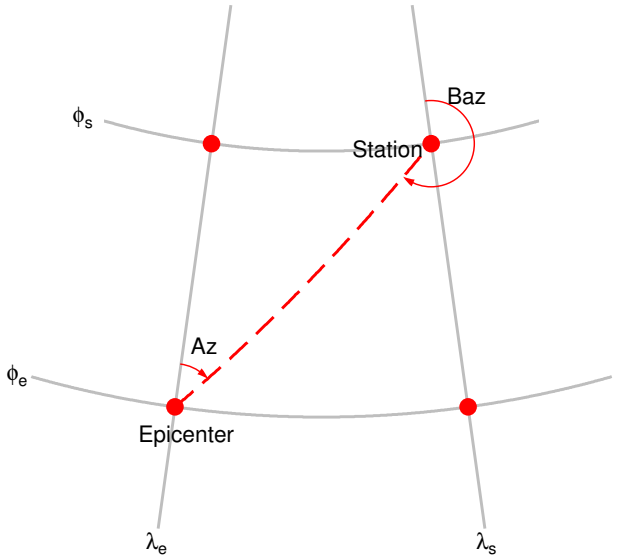

图 3.1: 震中距、方位角、反方位角示意图。

震中距、方位角和反方位角的计算涉及到球面三角的知识，具体公式及其推导可以参考相关代码及书籍。此处列出部分仅供参考：

- http://www.eas.slu.edu/People/RBHerrmann/Courses/EASA462/
- http://www.seis.sc.edu/software/distaz/
- SAC 源码 `src/ucf/distaz.c`
- CPS330 源码 `VOLI/src/udelaz.c`

`o, ko`

`o` 为事件的发生时刻相对于参考时刻的秒数。`ko` 是绘图时时间变量 `o` 的标识符。

`khole`

若为核爆事件，则其为孔眼标识；若为其它事件，则为位置标识。

`nevid, norid, nwfid`

三者分别标识事件 ID、起始时间 ID 和波形 ID，仅用于 CSS 3.0 文件中。CSS 3.0 是 SAC 可以处理的一种数据格式，应该是当初 SAC 商业化的产物，目前仍保留在 SAC 头段中。

3.4.4　台站相关变量

`knetwk, kstnm`

地震台网名和台站名。

`istreg`†

台站地理区域。

`stla, stlo, stel, stdp`

台站纬度 (-90 到 90 度)、经度 (-180 到 180 度)、高程 (单位 m，目前未使用)、相对地表的深度 (单位 m，目前未使用)。

`cmpaz, cmpinc, kcmpnm, kstcmp`

一个台站至少需要三个正交的通道/分量才能完整地记录地面运动物理量。`cmpaz` 和 `cmpinc` 指定了单个通道记录的方向矢量。

图 3.2 给出了 SAC 所使用的 NEU 坐标系，需要注意的是这是一个左手坐标系。图中蓝色箭头

为通道所记录的方向矢量，若地面运动与该方向一致，则为正，否则为负。其中，头段变量 cmpaz 表征通道的方位角，其定义为从 N 向开始顺时针旋转的角度，即图中的角度 ϕ；cmpinc 表征通道的入射角，定义为相对于 U 方向向下旋转的度数，即图中的角度 θ。

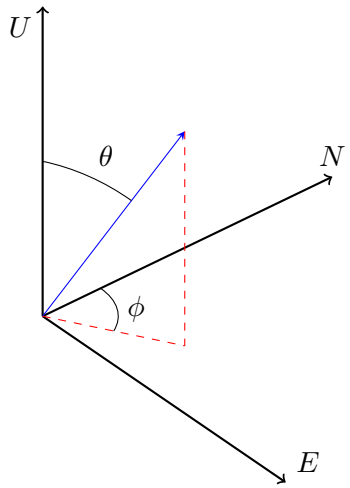

图 3.2: cmpaz 和 cpminc 示意图

根据定义，地震仪标准通道的 cmpinc 和 cmpaz 值如下表：

表 3.3: 标准地震通道的 cmpaz 和 cpminc

方向	cmpaz	cmpinc
N	0	90
E	90	90
U	0	0

对于非标准方向的地震通道来说，很容易根据 cmpinc 和 cmpaz 的值，将其旋转到 NEU 坐标系或者 RTZ 坐标系，这些将在 “分量旋转” 一节中说到。

kcmpnm 用于存储分量名称。SEED 格式规定通道名的三个字符中的最后一个代表通道的分量方位，比如通道名 BHE 表示该通道为东西向。通常 kcmpnm 可以取为 E、N、Z。由于很多台站的水平分量并不严格是东西、南北方向，因而现在更倾向于用 1 和 2 代替 N 和 E。

kstcmp 为辅助型变量，表示台站分量，由 kstnm、cmpaz、cmpinc 推导得到。

lpspol

如图 3.2 所示，在左手坐标系下，若三通道都是正极性则为真，否则为假。

3.4.5 震相相关变量

a, f, tn

a 和 f 用于存储事件的初动时刻和结束时刻相对于参考时刻的秒数。

Tn (n=0–9) 用于存储用户自定义的时刻相对于参考时刻的秒数，常用于存储震相到时。

ka, kf, ktn

a、f 以及 Tn 都有一个对应的以 k 开头的字符型头段变量，称之为时间标识。时间标识用于说明对应的时间头段变量中所包含时间的含义。

比如头段变量 a 中通常包含 P 波到时，则此时 ka 的值可以设置为 “P”；头段变量 t1 中包含了震相 PcP 的到时，则一般定义 kt1 为 “PcP”。

在绘图时，若时间头段变量中有值，则默认会在该时刻处绘制一条垂线，若相应的时间标记有定义，则将时间标记的值显示在垂线附近。

Xmarker

震相相关的变量对可以构成一个辅助型变量。a 和 ka 可以构成 amarker，f 和 kf 可以构成 fmarker，o 和 ko 可以构成 omarker，tn 和 ktn 可以构成 tnmarker（n=0–9）。

这些辅助型变量可以在 listhdr 中使用。

3.4.6　仪器相关变量

kinst, iinst†, respn†

kinst 为记录仪器的通用名称，iinst 为记录仪器的类型，respn 为仪器相应参数。

3.4.7　其它变量

usern

usern（n=0–9）用于存储用户自定义的浮点型数值。

kusern

kusern（n=0–2）用于存储用户自定义的字符型值。

lovrok

若为 TRUE，则磁盘里的原始数据可被覆盖；若为 FALSE，则原始数据不可被覆盖。主要用于保护原始数据，一般来说很少用到，若是出于保护原始数据的目的，应优先考虑对原始数据做备份。

lcalda

全称为 Calculate Distance and Azimuth。若为 TRUE，则当事件和台站的坐标被写入或被修改时，头段变量 dist、gcarc、az、baz 将自动计算，否则不会被自动计算，SAC 头段中会存在信息的不兼容。

kdatrd

数据被读入计算机的日期（一般很少使用）。

3.5　SAC 中的时间概念

3.5.1　基本思路

SAC 的头段区有很多与时间相关的头段变量，包括 nzyear、nzjday、nzhour、nzmin、nzsec、nzmsec、b、e、o、a、f、tn（n=0–9）。正确使用它们的前提是理解 SAC 中的时间概念。这一节将试着说清楚这个问题。

首先，SAC 处理的是地震波形数据，SAC 格式里保存的是时间序列数据。先不管其它的一些台站经纬度、事件经纬度信息，就数据而言，至少需要一系列数据值以及每个数据值所对应的时刻。

在本节接下来的内容中，将严格区分两个高中物理学过的概念：时刻和时间。简单地说，在时间轴上，时刻是一个点，时间是一个线段。

一个简单的例子如下：

```
2014-02-26T20:45:00.000      0.10
```

```
2014-02-26T20:45:01.000     0.25
2014-02-26T20:45:02.000     0.33
2014-02-26T20:45:03.000     0.21
2014-02-26T20:45:04.000     0.35
2014-02-26T20:45:05.000     0.55
2014-02-26T20:45:06.000     0.78
2014-02-26T20:45:07.000     0.66
2014-02-26T20:45:08.000     0.42
2014-02-26T20:45:09.000     0.34
2014-02-26T20:45:10.000     0.25
```

其中第二列是数据点,每个数据点所对应的时刻放在第一列,格式为 "yyyy-mm-ddThh:mm:ss.xxx"。数据点是以 1s 的等间隔进行采样的。

若把这堆时刻以及数据点直接写入文件中, 将占据大量的磁盘空间, 读写也很不方便。考虑将某一个时刻定义为参考时刻, 并把其它所有的时刻都用相对于该参考时刻的秒数来表示, 这样可以简化不少。

比如取 "2014-02-26T20:45:00.000" 为参考时刻, 即

```
nzyear = 2014
nzjday = 57
nzhour = 20
nzmin  = 45
nzsec  = 00
nzmsec = 000
```

则上面的数据可以简化为

```
00.000     0.10
01.000     0.25
02.000     0.33
03.000     0.21
04.000     0.35
05.000     0.55
06.000     0.78
07.000     0.66
08.000     0.42
09.000     0.34
10.000     0.25
```

其中第二列是数据点, 第一列是每个数据点对应的时刻相对于参考时刻的相对时间, 下面简称其为相对时间。

显然参考时刻的选取是任意的, 若取 "2014-02-26T20:45:05.000" 为参考时刻, 则上面的数据简化为

```
-05.000     0.10
-04.000     0.25
```

```
-03.000     0.33
-02.000     0.21
-01.000     0.35
 00.000     0.55
 01.000     0.78
 02.000     0.66
 03.000     0.42
 04.000     0.34
 05.000     0.25
```

一般来说，会选取一个比较特殊的时刻作为参考时刻，比如第一个数据点对应的时刻，或者地震波形数据中的发震时刻。

下面还是回到以 “2014-02-26T20:45:00.000” 为参考时刻简化得到的结果。因为数据是等间距的，相对时间这一列完全可以进一步简化，比如用 “起始相对时间 + 采样间隔 + 数据点数” 或者 “起始相对时间 + 采样间隔 + 结束相对时间” 就完全可以表征第一列的相对时间。

SAC 选择了另外一种简化模式，“起始相对时间 + 采样间隔 + 数据点数 + 结束相对时间”，即头段变量中的 “b+delta+npts+e”，这其实是存在信息冗余的，这就造就了头段变量 e 的一些特殊性，后面会提到。

按照 SAC 的模式在对相对时间进行简化之后，整个数据可以表示为

```
nzyear = 2014
nzjday = 57
nzhour = 20
nzmin  = 45
nzsec  = 00
nzmsec = 000
b      = 0.0
e      = 10.0
delta  = 1.0
npts   = 11

0.10
0.25
0.33
0.21
0.35
0.55
0.78
0.66
0.42
0.34
0.25
```

似乎到这里就结束了。

地震学里的一个重要问题是拾取震相到时（时刻），所以还需要几个额外的头段变量来保存这些震相到时（时刻），不过显然不需要真的把 “时刻” 保存到这些头段变量中，不然上面的一大堆就

真是废话了。SAC 将震相到时（时刻）相对于参考时刻的时间差（即相对时间）保存到头段变量 o、a、f、tn 中。

综上，SAC 中跟时间有关的概念有三个：

参考时刻 由头段变量 nzyear、nzjday、nzhour、nzmin、nzsec、nzmsec 决定

相对时间 即某个时刻相对于参考时刻的时间差（单位为秒），保存到头段变量 b、e、o、a、f、tn（n=0–9）

绝对时刻 = 参考时刻 + 相对时间

3.5.2 一些测试

下面以一个具体的数据为例，通过修改各种与时间相关的头段来试着去进一步理解 SAC 的时间概念。

生成样例数据

```
SAC> fg seis
SAC> lh iztype
   iztype = BEGIN TIME
SAC> ch iztype IUNKN
SAC> w seis
```

lh 是命令 listhdr 的简写，用于列出头段变量的值。ch 是 chnhdr 的简写，用于修改头段变量的值。这里额外多做了一个操作修改 iztype 的操作，这是由于这个数据稍稍有一点 bug。

iztype 指定了参考时刻的类型，其显示为 BEGIN TIME，实际上其枚举值是 IB，也就是说这个数据选取文件第一个数据点的时刻作为参考时刻，那么 b 的值应该为 0。而实际上这个数据的 b 值并不为 0，这其实是这个数据的一点小 bug。这也从另一个侧面说明 SAC 只有在修改与时间相关的头段变量时才可能会检查到这个错误/警告，所以这里先将其修正为 IUNKN。

修改文件起始时间 b

```
SAC> r seis
SAC> lh kzdate kztime b delta npts e o a f

    kzdate = MAR 29 (088), 1981
    kztime = 10:38:14.000
        b = 9.459999e+00
     delta = 1.000000e-02
      npts = 1000
         e = 1.945000e+01
         o = -4.143000e+01
         a = 1.046400e+01
SAC> ch b 10
SAC> lh

    kzdate = MAR 29 (088), 1981
    kztime = 10:38:14.000
        b = 1.000000e+01
     delta = 1.000000e-02
      npts = 1000
         e = 1.999000e+01
         o = -4.143000e+01
```

```
        a = 1.046400e+01
```

修改 b 前后的变化仅在于 b 和 e 值的变化，而参考时刻以及其它相对时间并没有发生变化。

这意味着整段 SAC 数据中的任意一个数据点所对应的时刻[1] 都向后延迟了 0.54 秒！这样做很危险，因为 b 和 e 的绝对时刻被修改了，而其它头段如 o、a、f、tn 的绝对时刻却没有变。

使用的时候必须非常小心：

- 如果 o、a、f、tn 都没有定义，那么修改 b 值可以用于校正仪器的时间零飘[2]以及时区差异[3]。关于时区的校正，参考 “时区校正” 一节。
- 如果 o、a、f、tn 已经被定义，则修改 b 值会导致与震相相关的头段变量出现错误！[4]

修改文件结束时间 e

```
SAC> r ./seis
SAC> lh kzdate kztime b delta npts e o a f

    kzdate = MAR 29 (088), 1981
    kztime = 10:38:14.000
        b = 9.459999e+00
     delta = 1.000000e-02
      npts = 1000
        e = 1.945000e+01
        o = -4.143000e+01
        a = 1.046400e+01
SAC> ch e 0
SAC> lh

    kzdate = MAR 29 (088), 1981
    kztime = 10:38:14.000
        b = 9.459999e+00
     delta = 1.000000e-02
      npts = 1000
        e = 1.945000e+01
        o = -4.143000e+01
        a = 1.046400e+01
```

可以看到，修改前后所有变量均没有发生变化，即 e 的值是不可以随意改变的，根据上面的结果可知，e 的值是通过 b、delta、npts 的值动态计算的。这也与上一节说到的头段变量冗余问题相符合。不要试图修改 delta、npts，这不科学！

修改 o、a、f、tn

这几个头段变量完全是由用户自定义的，因而任何的定义、修改、取消定义都不会对数据的正确性产生影响，因而这里不再测试。

---

[1] 好长的修饰语

[2] 零飘，即仪器中的时刻与标准时刻不同。

[3] 时区差异可以理解成另一种零飘。

[4] 如果只定义了 o 值，或者 a、f、tn 为理论震相到时而非计算机拾取或人工拾取的到时，修改 b 也是没有问题的。有些乱，不多说了。总之不要随便修改 b 的值。

修改参考时间

```
SAC> r ./seis
SAC> lh kzdate kztime b delta npts e o a f

    kzdate = MAR 29 (088), 1981
    kztime = 10:38:14.000
         b = 9.459999e+00
     delta = 1.000000e-02
      npts = 1000
         e = 1.945000e+01
         o = -4.143000e+01
         a = 1.046400e+01
SAC> ch nzsec 15
SAC> lh

    kzdate = MAR 29 (088), 1981
    kztime = 10:38:15.000
         b = 9.459999e+00
     delta = 1.000000e-02
      npts = 1000
         e = 1.945000e+01
         o = -4.143000e+01
         a = 1.046400e+01
```

试图修改参考时刻，整个 SAC 头段，除了参考时刻外其它时间变量都没有发生变化。根据“绝对时刻 = 参考时刻 + 相对时间”可知，这导致所有 SAC 数据点的绝对时刻发生了平移，这一点理论上可以用于校正零飘或者时区，但是由于 SAC 不支持智能判断时间（比如不知道 1 时 80 分实际上是 2 时 20 分），所以修改时区时需要获取参考时刻 6 个头段变量，加上时区的校正值，再写入到参考时刻 6 个变量中，相对较为繁琐，因而若要校正时区，建议直接修改头段变量中的 `b` 值。

修改发震时刻

数据处理中一个常见的需求是修改发震时刻，这可以通过修改头段变量 `o` 来实现，但是经常需要将参考时刻设置为发震时刻。上面的测试表明，直接修改参考时刻是很危险的，所以 SAC 的 `ch` 命令提供了 `allt` 选项来实现这一功能，在“事件信息”一节中会具体解释。

3.5.3 总结

将 SAC 中的时间变量分为三类：

1. 参考时刻：即 `nzyear`、`nzjday`、`nzhour`、`nzmin`、`nzsec`、`nzmsec`；
2. 相对时间：即 `o`、`a`、`f`、`tn`；
3. 特殊的相对时间：即 `b`[1]；

第二类时间变量可以随意修改，即震相拾取。

第一、三类时间变量的修改会导致数据绝对时刻发生改变。一般通过修改第三类时间变量来校正时间零漂和时区差异。在设置了发震时刻后，应使用 `chnhdr` 命令的 `allt` 选项修改第一、三类时间变量。

[1] 由于 e 不可独立修改，所以不再考虑