

TEQC - 中文教程

目录

- [第 1 节](#) - 简介: teqc
 - [第 2 节](#) - UNAVCO 万维网支持和联系
 - [第 3 节](#) - 数据类型
 - [第 4 节](#) - 基本操作模式
 - [第 5 节](#) - 操作系统和硬件
 - [第 6 节](#) - 标准输入, 标准输出和标准误差
 - [第 7 节](#) - 关于语法的一般概念
 - [第 8 节](#) - 使用 teqc 进行 RINEX 格式化和 RINEX 格式校验
 - [第 9 节](#) - 使用 teqc 进行 RINEX 标题编辑和提取; 配置选项和文件简介以及 teqc 选项层次结构
 - [第 10 节](#) - 配置选项和命令行选项
 - [第 11 节](#) - 使用 teqc 进行质量检查 (qc) 模式
 - [第 12 节](#) - 带有多个文件输入
 - [第 13 节](#) - 使用 teqc 进行时间窗口化
 - [第 14 节](#) - 用 teqc 拼接
 - [第 15 节](#) - 用 teqc 转换
 - [第 16 节](#) - 特殊转换注意事项和选项
 - [第 17 节](#) - 波长因素: 根据它们 teqc 可以做什么
 - [第 18 节](#) - 基本命令: 预览
 - [第 19 节](#) - 在脚本中使用 teqc: 批量模式的使用方法
 - [第 20 节](#) - teqc 的 qc 模式和原始的 UNAVCO QC 之间的差异
 - [第 21 节](#) - teqc 的 qc 模式输出解释
 - [第 22 节](#) - 看似“奇怪”行为
-

01 简介: teqc

本文档描述并作为 teqc (发音为“tek”) 主要功能的教程。虽然 teqc 的功能不仅仅适用于 RINEX 文件, 但大多数用户可能使用的最常见的数据格式是 RINEX 格式, 可以是输入格式, 也可以是输出格式, 或两者兼而有之。因此, 本文档中使用了三种基本 RINEX 格式的简写:

- [OBS](#) 用于 RINEX 观测数据文件,
- [NAV](#) 用于 RINEX 导航消息文件,
- [MET](#) 用于 RINEX 气象数据文件。

此外, teqc 当前处理 RINEX 版本 1 和 2 文件 (至版本 2.10), 但尝试编辑 RINEX 版本 1 文件将导致自动转换为 RINEX 版本 2.XX (特别是 2.10) 文件。

如果您的主要兴趣是将原生二进制格式转换为 RINEX, 请直接转到[转换](#)部分 ([第 15 节](#) 和 [第 16 节](#))。

如果您的主要兴趣是编辑, 请直接进入[元数据编辑/提取 \(第 9 节\)](#) 或 [RINEX 格式 \(第 8 节\)](#) 或 [窗口 \(切割\) \(第 13 节\)](#) / [拼接 \(第 14 节\)](#) 操作部分。

如果您的主要兴趣是 qc -ing RINEX 或本机二进制数据, 请直接转到 teqc 的 [qc 模式 \(第 11 节\)](#) 部分。

02 UNAVCO 万维网支持与联系

- 有关 [teqc 的](#) 信息。
- 本[教程文档](#)。
- 关于 teqc 的[常见问题解答](#)（“常见问题解答”）。
- teqc [开发和发布日志](#)（包括下一版本的错误修复）。
- 每个正式版本的[发行说明](#)列表，其中包含本教程中可能没有的详细信息。
- 还有一个 [teqc 电子邮件论坛](#)和电子邮件存档。您可以搜索 teqc 电子邮件，或按日期或线程（电子邮件主题）列出 teqc 电子邮件。

要订阅或取消订阅 teqc 电子邮件论坛，请使用上述 URL。

要向 teqc 电子邮件论坛发送消息：

地址 - 收件人: [teqc @unavco.org](mailto:teqc@unavco.org)
subject--主题: <可选，但有助于维护线程>
message-- <消息体；评论，错误报告，问题，
关于 teqc >的
难题

- 如有其他问题，未解决的问题或可能的错误报告，请联系 UNAVCO 的 [teqc 技术联系人](#)。

03 数据类型

- RINEX 数据文件：

RINEX 版本 2.XX (2.10) 文件是 teqc 期望处理和/或生成的默认文件类型。为了获得有效的 RINEX 版本 2.XX 文件，该文件必须符合伯尔尼大学（或参见 [RINEX 2.10](#)）文档“RINEX: Receiver Independent Exchange Format Version 2”中的规范。每个文件必须存在最小的头记录集。这些是 RINEX 2.10 文档中指定的非可选标头记录。（注意：目前没有支持 [RINEX 3.00 的](#)计划。）

可以使用 teqc 读取 RINEX 版本 1 文件并在输出时转换为 RINEX 版本 2.10 文件。此外，如果输出（例如 读取并识别 rinex 版本/类型并在 RINEX 输出上转换为 RINEX VERSION / TYPE），则会识别以下标题行的小写版本并将其转换为大写。

每个 RINEX OBS 文件必须有一个带标题行的标题（从该行的第 61 个字符开始），以下结尾：

```
RINEX VERSION / TYPE      （必须是第一行）
PGM / RUN BY / DATE
标记名称
观察员/机构
REC# / TYPE / VERS
ANT# / TYPE
APPROX POSITION XYZ
天线: DELTA H / E / N.
WAVELENGTH FACT L1 / 2    （默认值）
#/观察类型
第一次 OBS 的时间
END OF HEADER              （必须是版本 2.xx 的标题的最后一行）
```

其中有效信息（即格式版本= 1,2 或 2.XX; 文件类型='O'; 卫星系统='', 'G', 'R', 'S', 'T' 或 'M'）必须出现在第一行，必须在 # / TYPES OF OBSERV 记录行上指定观察的数量和类型，并且必须给出 L1 和 L2 波长因子的默认值。如果期望描述符字符串，则其他标题记录中的其余字段可以是空白的，或者如果期望数值（即使它为零），则具有某个数值，但是这些其他标题行必须存在或不存在非空白信息。观察数据通常遵循 END OF HEADER 标题记录。（请注意，RINEX 版本 1 没有 END OF HEADER 字段，

每个 RINEX NAV 文件必须有一个带标题行的标题（从行中的第 61 个字符开始），结尾为：

```
RINEX VERSION / TYPE      （必须是第一行）
PGM / RUN BY / DATE
END OF HEADER              （必须是版本 2.xx 的标题的最后一行）
```

其中有效信息，即格式版本= 1,2 或 2.XX; 文件类型=

用于 GPS 导航消息文件

'G' 的 G' 用于 GLONASS 导航消息文件

'E' 用于 针对用于 QZSS 导航消息文件 'I'

的北斗/罗盘导航消息文件

'J' 的伽利略导航消息文件 'C'

'对于

SBAS 导航消息文件的 IRNSS 导航消息文件 "H"

必须出现在第一行。星历数据通常遵循 END OF HEADER 标题记录。（请注意，RINEX 版本 1 没有 END OF HEADER 字段，而是有一个空行。）

每个 RINEX MET 文件必须有一个带标题行的标题（从行中的第 61 个字符开始），以：

```
RINEX VERSION / TYPE      （必须是第一行）
PGM / RUN BY / DATE
标记名称
# / 观察类型
END OF HEADER              （必须是版本 2.xx 的标题的最后一行）
```

其中有效信息（即格式版本= 1,2 或 2.XX; 文件类型='M'）必须出现在第一行以及 # / TYPES OF OBSERV 标题记录行中指定的观察数量和类型。气象数据通常遵循 END OF HEADER 标题记录。（请注意，RINEX 版本 1 没有 END OF HEADER 字段，而是有一个空行。）

- Trimble *.dat 下载文件集：

来自最近的接收器（4000 系列到 NetRS 代）的 Trimble *.dat 文件可通过 teqc 读取。一些 *.dat 数据记录尚未编码到 teqc 中（例如旧的 GPS 可观察记录 0, 1, 2 和 7）。但是，teqc 应该能够读取整个文件并报告尚未编码的记录。

MES 文件不是必需的，但如果它们存在并且您正在使用目标文件名（而不是 stdin），则 teqc 将使用匹配的 MES 文件来处理每个目标输入文件以帮助解析某些元数据。teqc

不使用 ION 和 EPH 下载文件。

- Trimble RT17 流:

Trimble 4000 SE / SSE / SSi 及后续接收器的 Trimble RT17 格式可通过 `teqc` 读取。目前，只有带有星历信息和记录 57h（GPS 或 MET 可观测量）的记录 55h 已被编码，但 `teqc` 将再次报告并跳过其他记录。

- Trimble 标准接口协议（TSIP）:

最小的支持。

- Ashtech 下载文件（B-，E-，S-和 D-文件）:

来自各种 Ashtech 接收器的 Ashtech 下载文件格式（例如，Z-12，Z-18，GG24，L-XII，LM_XII）可通过 `teqc` 读取。通常情况下，Ashtech “平滑” 的伪距没有应用，但可以打开。在 B 文件“版本”为 3 或更高的情况下，对 RINEX 的转换应该可以正常工作 - 对于任何最近的 Ashtech 固件都是这种情况。对于 B 文件的版本号为 2 或更小的转换，相位转换将出错。（您可以使用 `teqc` 选项+ `diag` 找到版本号。）

- Ashtech RS-232（实时）流:

来自各种 Ashtech 接收器（例如，Z-18，Z-18，GG24，G-12）的 Ashtech RS-232（实时）二进制数据格式可通过 `teqc` 读取。这包括二进制 MBEN，PBEN，SNAV 和 DBEN 记录。与 Ashtech 下载格式一样，Ashtech “平滑” 伪距在默认情况下不会应用，但可以打开。

- Ashtech R 文件:

可以从某些接收器（例如 Z-12）下载 R 文件格式，这可以由 `teqc` 直接读取。

- Ashtech U 文件:

可以从 micro-Z 接收器下载 U 文件格式，这可以由 `teqc` 直接读取，包括新的“数据模式 7”格式。

- 徕卡 LB2 格式:

`Teqc` 可以读取 Leica MC1000，CRS1000 和 CRS1500 以及后来的系统 500 和 1200 接收器使用的 Leica LB2 格式。转换假设接收器上仅使用一个天线端口。

- 徕卡 MDB 格式:

`Teqc` 可以从系统 500 和 1200 接收器读取 Leica MDB 格式。

- 徕卡 DS 格式:

`Teqc` 可以读取早期 Leica 接收器使用的 Leica 200/300 DS 文件集格式，例如 CR233，CR244，SR299，SR299E，SP299P，SR260，SR261，SR399，SR9500。将使用带有后缀 `.cmp`，`.dat`，`.eph`，`.int`，`.met`，`.obs` 和 `.pnt` 的文件。具有其他后缀的文件（例如，`.chn`，`.alm`，`.atf` 和 `.atl`）目前被忽略。（注意：Leica 可能从未表示/支持 `.met` 文件，该文件本可用于 RINEX MET 转换。但是 如果用户遇到任何示例，`teqc` 中会有一些原型代

码。)

- 拓普康 TPS / Javad JPS:

如果格式为 Topcon TPS, 则可以使用 teqc 读取 Topcon 和 Javad 接收器的二进制数据, 这或多或少等同于 Javad JPS 格式。虽然根据需要进行了新的开发, 但 teqc 并未读取所有 TPS / JPS 消息。但是, 以下警告适用: 如果要使用的一组 TPS / JPS 消息不是默认消息, 或者如果以某种方式更改了默认消息集中采用的消息顺序, 请确保更新的消息集保持“epoch 同步”, 即消息[~~]和[RD]位于包含代码, 载波相位和当前时期收集的其他类型测量的消息之前。如果用户违反此条件, 他或她可能无法使用 teqc 或 Topcon 的 Pinnacle™和其他 TPS 后处理软件正确处理相应制造商的格式数据文件。

- Septentrio 二进制格式:

Septentrio 接收器的 Septentrio 二进制格式 (SBF) 可以用 teqc 读取。

- Navcom 二进制文件:

可以使用 teqc 读取 Navcom 接收器的二进制数据。

- 来自 TurboRogue / TurboStar 和 Benchmark ACT 接收器的 ConanBinary:

来自 Turbo 系列 Rogue 接收器的 ConanBinary 数据可通过 teqc 读取。(来自原始 Rogues 的 ConanBinary 数据按 SV 编号排序, 而不是按时间排序, 并且无法使用 teqc 正确转换)。

- TurboBinary

使用 teqc 可读取 TurboBinary 数据。这包括具有正常速率数据, 高速率 (50Hz) 数据的数据, 以及包含 LC 数据的所谓“30-1 秒”数据。但是, LC 数据的提取不是自动的 (因为它不是标准的 RINEX 版本 1 或 2), 您必须使用 -0. obs 选项将 LC 数据指定为观察数据类型之一。

来自 AOA Benchmark ACT 接收器的 TurboBinary 数据可以通过 TurboBinary 的常规方式进行转换, 或使用特殊的 Benchmark 转换选项。例如, 使用 -aoa tbY 选项 (对于“Y *无代码”接收器), C / A 导出的 L1 相位值用于 RINEX “L1” 可观测值, 而不是 Y1 无码 L1 相位值。

- IGS RTigs 格式:

支持, 但 IGS 可能会添加 teqc 可能无法读取的新功能。

- JPS Soc 格式:

支持的。

- 加拿大马可尼二进制格式:

Teqc 可以读取加拿大 Marconi 二进制格式，主要用于 CMC Allstar OEM (CMT-1200) 接收器。到目前为止，这包括记录 ID 21, 22, 23, 63 和 126 - 足以写入 RINEX [OBS](#) 和 [NAV](#) 文件。转换将包括连续时期之间的多个 175 ns 时钟复位 (Allstar 的时钟转向签名)。

- u-blox UBX 格式:

Teqc 可以读取 u-blox UBX 格式，但仅解码消息 NAV-CLOCK (0x0122)，NAV-POSLLH (0x0102)，RXM-EPH (0x0231) 和 RXM-RAW (0x0210) 中的信息。

- 罗克韦尔十二生肖二进制格式:

Teqc 可以读取 Rockwell Zodiac 接收器中使用的二进制格式。到目前为止，这包括消息 (记录) ID 1000, 1002 和 1102 - 足以写入 RINEX [OBS](#) 文件。(Rockwell Zodiac 格式中没有包含 SV 星历表或 MET 数据的记录。)

- 摩托罗拉 Oncore 格式:

除了 L1 阶段，Teqc 会正确地将 Oncore 格式转换为 RINEX [OBS](#)。

- ARGO 格式:

可以使用 teqc 读取两个 ARGO 格式文件类型 *.dat 和 *.orb。ARGO *.dat 文件等同于 RINEX [OBS](#) 文件; ARGO *.orb 文件等同于 RINEX [NAV](#) 文件。

- 德州仪器 (TI) 4100 GESAR, BEPP / CORE 和 TI-ROM 格式:

可以使用 teqc 读取 TI-4100 的二进制数据，但尚未测试所有记录类型的代码。(这是因为到目前为止还没有遇到某些记录类型的例子。例如，到目前为止，在实际文件中只遇到了 TI-ROM 格式的记录 1，这足以写入 RINEX [OBS](#) 文件。这些转换器主要用于读取遗留数据。迄今为止，teqc 可以读取所谓的 GESAR 和/或 BEPP / CORE 数据 (可以是混合物)，也可以读取原始的 TI-ROM 格式。根据数据中的记录类型，不仅可以提取 P1 / CA, P2, L1 和 L2，还可以提取信噪比 (S1 和 S2) 和多普勒 (D1 和 D2)。

- 其他格式的状态:

- 对于包含记录 0 或 7 的旧 Trimble *.dat，请尝试 Berne 的 TRRINEX0 程序
; 如果这不起作用，试试 Trimble 的 dat2rin 程序; 如果不起作用，请联系 Trimble 寻求帮助
- 对于版本 3 之前的 Ashtech B 文件，尝试 Ashtech 的转换程序 (直接联系 Ashtech); 如果这不起作用，试试伯尔尼的 ASRINEX0 ; 如果不起作用，请联系 Ashtech 获取更多帮助

04 基本操作模式

teqc 有三种不同的基本操作模式:

- [转换](#)
- 编辑: [元数据提取和/或编辑](#), [格式化](#), [窗口化 \(切割\)](#) 和/或[拼接](#)

- [质量检查 \(qc\)](#)

这些模式中的任何一种都可以单独使用，或者彼此组合使用。例如，您可以使用 `teqc` 的一些方法是：

- 检查 RINEX 文件是否符合 RINEX 版本 2.XX 规范； 例如，标识缺少的非可选标头字段
- 修改（编辑）RINEX 文件中的任何现有 RINEX 标头字段，并输出生成的已编辑 RINEX 文件
- 质量检查有效的 RINEX [OBS](#) 文件，但没有 RINEX [NAV](#) 文件或二进制星历表（`qc "lite"` == 没有位置信息）
- 质量检查有效的 RINEX [OBS](#) 文件或文件使用有效的 RINEX [NAV](#) 文件或文件中的星历数据（`qc "完整"` == 位置信息可能）
- 窗口或剪切（指定时间子窗口）和/或拼接两个或多个 RINEX 文件
- 将某些原生二进制格式（例如，Trimble *.dat）转换（转换）为 RINEX [OBS](#) 和/或 [NAV](#) 文件

这些操作模式单独工作或彼此协同工作。例如，Trimble 二进制流可以转换为 RINEX [OBS](#) 和 [NAV](#) 文件； 填写了 [OBS](#) 文件中的空标题字段（例如 MARKER NAME）； 如果在数据流中遇到足够的卫星星历表，则在单个 `teqc` 运行中具有流 `qc -ed`，显式时间窗口以及从 `qc -lite` 自动切换到 `qc -full` 。 首次使用者注意到以下内容可能也会有所帮助：

- 关于 `teqc` 使用的文件名进行了最少量的假设。本质上，文件名可以是 OS 的任何有效名称，除了没有输入文件名可以以 “ - ” 或 “+” 字符开头，并且最好避免使用带有空格（如空格）的名称。Berne 推荐的 RINEX 文件命名约定虽然对于 `teqc` 来说不是必需的，但是可以接受，并且可以在命令行或使用 `teqc` 的脚本中使用。
- 一般来说，`teqc` 现在不仅适用于来自 NAVSTAR GPS 系统的数据和导航信息，还适用于 GLONASS，Galileo，Beidou / Compass，QZSS，IRNSS（L 波段）和 SBAS（例如 WAAS，EGNOS）。。
- `teqc` 是 100% 非交互式的； 它不会查询用户的输入或查找是否正常。您可能会收到 `stderr` 的“通知”，“警告”或“错误”。如果出现问题（通常是“错误”或使用问题），`teqc` 会通知用户并终止。
- 通常，`teqc` 不使用硬连线数组大小，而是根据需要分配和释放内存。只要您的计算机有足够的计算机内存，就不应该遇到数组绑定问题。
- `teqc` 对内存使用保守。

`teqc` 的基本设计是面向命令行的，遵循 UNIX shell 模型。对于本文档的其余部分，将假定用户使用的是 UNIX OS，并且熟悉基本的 UNIX 命令。当程序在其他操作系统上移植和测试时，将包括特定于其他操作系统（例如 DOS）的文档。

05 操作系统和硬件

迄今为止，`teqc` 已经过 UNAVCO 人员和其他用户的测试：

- Linux x86
- Solaris Sparc 2.3 及更高版本
- Solaris x86 2.6 及更高版本
- HP-UX 10.20 及更高版本（PA-RISC 平台）
- DEC Digital-UNIX OSF1 V4.0
- DEC Alpha Linux
- IBM AIX 4.3
- SGI IRIX 5.3

Macintosh OSX

Microsoft (95/98 / NT) / 2000 / XP)

随着时间的推移可能包括对其他平台的支持。如果您需要在上面未列出的 UNIX 平台上提供支持，并且可以在该平台上使用 ANSI 或 K&R (Kernighan 和 Ritchie - 又名“传统”) C 编译器提供访客登录，请联系 [teqc 技术联系人](#)。

06 标准输入，标准输出和标准误差

另一个基本设计特性是使用标准输入 (stdin)，标准输出 (stdout) 和标准错误 (stderr)。teqc 可以在管道中接受 RINEX 格式流或二进制数据流作为 stdin 而不是文件作为输入。Stderr 保留用于报告 teqc 遇到任何文件或 stdin 中不喜欢或不理解的内容时可能出现的问题。Stdout 用于转储用户请求的 ASCII 产品，与命令行语法一致。然后 teqc 的输出在当前有以下警告：stdout 和 stderr 必须能够分开。

因此，鼓励用户使用可以明显区分 stdout 和 stderr 的 shell。对于 UNIX，这包括 Bourne shell (sh) 和 Korn shell (ksh)。对于 UNIX C-shell (csh) 或 DOS -它们不允许你将 stdout 和 stderr 分别指向不同的文件 - teqc (即+ err filename) 的一个选项可用于发送到 stderr 的内容到另一个文件，以避免 stdout 和 stderr 否则会去同一个地方时的不愉快。对于本文档的其余部分，将假定 UNIX 用户使用 sh 或 ksh 虽然以下示例应该允许用户轻松使用 csh 或 MS DOS shell 来实现相同的结果。

虽然本教程的其余部分假设您将使用 sh 或 ksh，但您可以轻松地使用 teqc 与 csh 或 DOS 来控制 stdout 和 stderr。使用 csh 或 DOS (或任何其他 shell) 时，可以使用命令 options + out, ++ out, + err 和/或++ err 让 teqc 在内部将 stdout 和/或 stderr 重定向到特定文件。从而：

sh 或 ksh:

```
teqc { rest of command } 2> err.txt [stdout to screen]
```

任何 shell:

```
teqc + err err.txt { rest of command } [stdout to screen]
```

要么

sh 或 ksh:

```
teqc { rest of command } > out.txt [stderr to screen]
```

任何 shell:

```
teqc + out out.txt { rest of command } [stderr to screen]
```

应完全相同。您甚至可以在同一个 teqc 执行中使用+ out 和 + err 来写入相同的文件名：

sh 或 ksh:

```
teqc { rest of command } > temp 2>&1
```

任何 shell:

```
teqc + out temp + err temp { 其他命令 }
```

此外，您可以使用++ out 或++ err 追加到现有文件：

sh 或 ksh:

```
teqc { rest of command } >> out.txt 2 >> err.txt
```

任何 shell:

```
teqc ++ out out.txt ++ err err.txt { rest of command }
```

要仅使用 stdout 或 stderr 附加，或使用++ out 或++ err:

任何 shell:

```
teqc { rest of command } >> out.txt [stderr to screen]
```


任何 shell:

```
teqc ++ out out.txt { rest of command } [stderr to screen]
```

要么

任何 shell:

```
teqc { rest of command } 2 >> err.txt [stdout to screen]
```

任何 shell:

```
teqc ++ err err.txt { rest of command } [stdout to screen]
```

简而言之，无论您使用什么 shell，都应该有一种方法可以实现 stdout 和 stderr 重定向所需的功能。

07 关于语法的一般概念

teqc 的一般语法是:

```
teqc {options} [ file1 [ file2 [...]]]
```

或 (DOS 外壳除外):

```
teqc {options} <stdin
```

或类似的

```
..... | teqc {options}
```

命令行末尾列出的文件或 stdin 是每次执行 teqc 时要处理的目标。这是提到的，因为选项中可能存在其他文件名，但选项中列出的任何文件都是处理配置的一部分，不被视为处理的目标。

甚至只执行

TEQC

(即，不存在目标) 是允许的; 这将返回 teqc 关于它当前基于 CPU 时钟的 GPS 周的最佳猜测 (但这个猜测的好坏取决于 CPU 的时间设置的程度)。

有一个助记符管理每个选项的使用-和+:

领先-:

表示意图输入内容 (除了标准输入或目标文件列表),

或表示意图关闭某些选项

领先+:

表示输出内容的意图 (除了 stdout 和/或 stderr),

或表示打开某个选项的意图

对于某些选项，领先-和+做同样的事情。 例如， 获得帮助

```
teqc -help
```

和

```
teqc +帮助
```

两者都广泛使用 stderr。(遵循助记符，只有+帮助 应该提供帮助，但为什么在用户请求帮助时进一步混淆问题? 因此，两者都在这里工作。) 此外，[RINEX 标题编辑选项标志](#)以相同的方式工作: 例如，-0.mo 与+ 0.mo 相同。您可以将-0.mo 视为输入一些标题信息 (标记名称)，或使用+ 0.mo 强制输出标题信息。

为了尝试检测可能的命令行输入错误，请在命令行末尾以字符 “ - ” 或 “ + ” 开头的目标文件名’ 目前不被允许。以字符’ - ’ 或’ + ’ 开头的任何内容始终假定为命令行选项。

有时，特别是在添加和调试新功能（例如，新转换器）时，您可能会被警告消息淹没到 `stderr`。通常可以通过包含 `-warn` 选项来抑制其中大部分：

```
teqc -warn { rest of command }
```

08 使用 `teqc` 进行 RINEX 格式化和 RINEX 格式验证

假设您有三个 RINEX 文件（使用 Berne 推荐的命名约定）：`fbar0010.97o`，`fbar0010.97n` 和 `fbar0010.97m` 分别是您的 [OBS](#)，[NAV](#) 和 [MET](#) 文件名。让我们假设您首先要验证您的 `fbar *` 文件是否可以解释为 RINEX 版本 2.XX 格式兼容，即它们的格式是这样的，它们具有所有的位和片段，使它们看起来像 RINEX 文件（但不是说它们中的信息必然有效）。一种方法是执行：

```
teqc fbar0010.97o
teqc fbar0010.97n
teqc fbar0010.97m
```

您看到被转储到屏幕上的是经过重新处理的 RINEX 版本 2.XX 格式。原始目标文件中的所有信息都将保留在输出版本中（- 如果不是，则存在错误，因此请[报告](#)此情况）。

或者你可以执行：

```
teqc fbar0010.97o> temp0010.97o
teqc fbar0010.97n> temp0010.97n
teqc fbar0010.97m> temp0010.97m
```

在这种情况下，重新处理的 RINEX 文件被重定向（`stdout`）并保存为一组 `temp *` 文件。

执行上述三个命令后，执行以下操作可能也是有益的：

```
diff fbar0010.97o temp0010.97o | more
```

查看原始目标文件和重新处理的文件之间的某些差异可能是什么。如果原始文件是由 `teqc` 生成的，那么你不应该看到任何差异（- 而且，如果你这样做，那就有一个 bug，所以请[报告](#)这个）。

事实上，如果上述任何输入 RINEX 文件 `fbar0010.97 *` 存在格式问题，`teqc` 将输出 `stdout`（或者在第一组示例中输出到屏幕，或者重定向到第二组中的文件）示例）直到遇到问题，此时它将使用 `stderr` 报告问题并终止。TEQC 关于 RINEX 文件应该是什么的几乎没有猜测；如果文件有问题，在 `teqc` 继续进行之前必须使用人或其他程序来修复它。

现在假设您有一个 RINEX 文件列表，您希望检查 RINEX 版本 2.XX 格式合规性，但不想保存任何重新处理的标准输出。有几种方法可以做到这一点（例如，在 UNIX 中）：

```
teqc fbar0010.97o> / dev / null
teqc fbar0010.97n> / dev / null
teqc fbar0010.97m> / dev / null
```

或者，使用命令行选项 `+v`（任何 shell）：

```
teqc +v fbar0010.97o
teqc +v fbar0010.97n
teqc +v fbar0010.97m
```

要么：

```
teqc +v fbar0010.97o fbar0010.97n fbar0010.97m
```

或（UNIX shell 正则表达式）：

```
teqc +v fbar0010.97 [mo]
```

甚至（例如，如果使用 ksh）：

```
teqc + v`ls fbar0010.97 *`
```

本质上的+ V 选项做了三件事：

1. 关闭转储到 stdout - 所以 `teqc + v fbar0010.97o` 应该比 `teqc fbar0010.97o> / dev / null` 执行得更快，
2. 抑制文件“拼接” - 所以 `teqc` 理解输入文件不一定是相同的 RINEX 类型，并且
3. 向 `stderr` 转发一条短消息，说明每个输入文件在执行文件结束时符合 RINEX 版本 2.XX 规范，或者如果遇到问题则转发给 `stderr` 的错误消息。

`teqc` 还检查目标是否适当的时间排序。对于 [OBS](#) 和 [MET](#) 文件，正在检查的时间标记是观察和/或事件时期。对于 [NAV](#) 文件，检查三个时间标记：TOC（“时钟时间”）时期，TOE（“星历时间”）时期和 TOW（“传输时间”）。对于 [OBS](#) 和 [MET](#) 文件，需要时间排序。对于 [NAV](#) 文件，对每个星历进行三次健全性检查。

当输入相同类型的多个目标文件时，`teqc` 会查看此时间排序是否仍然是顺序的（尽管当前允许的时间段完全相同） - RINEX NAV 文件除外，其中信息在使用前已经过排序。因此，假设 `fbar0020.97o` 中的数据确实遵循 `fbar0010.97o`，执行

```
teqc + v fbar0020.97o fbar0010.97o
```

将在 `fbar0010.97o` 的第一个观察时期导致错误消息和程序终止（假设没有其他错误）。

09 使用 `teqc` 进行 RINEX 标题编辑和提取；

配置选项和文件简介以及 `teqc` 选项层次结构

正如经验丰富的 RINEX 文件用户所知，任何或所有 RINEX 标头信息可能都不正确。原则上，任何使用 ASCII 文件编辑器的人都可以修改任何此类信息，例如 UNIX OS 上的“ed”，“ex”或“vi”，逐个文件（顺便提一下），突出了 RINEX 最严重的漏洞之一 - 易于故意或非故意的数据篡改。

然而，经常发生的是，需要在大量 RINEX 文件上校正相同类型的信息，并且需要在所有受影响文件的这些字段中放置相同的校正信息。在这种情况下，使用 `teqc` 的 RINEX 修改（编辑）功能可能更容易。

例如，假设需要更正纪念碑（标记）名称以读取 [OBS](#) 文件 `fbar0010.97o` 中的“ foobar 站点 ”。这可以通过执行来完成

```
teqc -0.mo "foobar site" fbar0010.97o> temp0010.97o
```

在这种情况下，更正的文件现在是 `temp0010.97o`。该 `-0.mo` 选项指定任何原碑名 [OBS](#) 正在处理的文件是用字符串“被覆盖在 foobar 的网站 ”。注意命令行上的双引号封装了包含空格的字符串作为空格。如果您希望将纪念碑名称更改为“ foobar ”，则可以执行

```
teqc -0.mo "foobar" fbar0010.97o> temp0010.97o
```

要不就

```
teqc -0.mo foobar fbar0010.97o> temp0010.97o
```

请注意，在这种情况下，替换字段中没有空格（即新的标记名称），因此双引号是可选的。

有一种类似的机制可以更改 RINEX 文件中的每个头字段，除了 1) RINEX 注释，在这种情况下，用户只能在 RINEX 头中添加更多注释，以及 2) RINEX 文件的第一个头记录。而不是在这里列出所有可能的选项，通过将 `++config` 选项与 RINEX 目标文件一起使用，`teqc` 更容易实现：

```
teqc ++ config fbar0010.97o
```

这会将所有可更改的标头信息和当前值（即 [OBS](#) 文件中的那些信息 fbar0010.97o）转储到 stdout。相关选项 + config 仅显示已通过命令行或其他方式设置的选项。要查看差异，请尝试：

```
teqc -0.mo foobar + config fbar0010.97o
```

```
teqc -0.mo foobar ++ config fbar0010.97o
```

基本上，+ config 意味着：“告诉我 teqc 的内部默认选项设置/值已被覆盖”； ++ config 表示：“告诉我如何设置所有 teqc 选项，包括内部默认值”。

执行公正

```
teqc ++ config
```

将显示 teqc 的通用默认配置选项（加上关于进入 stderr 的 GPS 周的几行）。

当仅执行 teqc ++ config（即没有目标文件）时，两个选项 -st [art_window]和-e [nd_window]显示 teqc 能够处理的总可能时间范围- 低于 a 的分辨率飞秒（1e-15 秒）。这些选项的参数格式为 YYYYMMDDhhmmss.sss。在内部，12 位用于存储年份的值，使 teqc 能够处理 4096 年。因此，随着公元 1980 年开始的内部日历（GPS 日历始于 1980 年 1 月 6 日），teqc 的日历将不会很快过时 - 例如 teqc2000 年 1 月 1 日通过了 Y2k 过渡，1999 年 8 月 22 日完成了 GPS 周 1024 翻转。无论您是否指定，teqc 始终使用定义的时间窗口，其中执行 teqc ++ config 显示该时间窗口的最大边界。请注意，您不能在 1980 年 1 月 1 日之前使用时间。

执行 teqc ++ config 之后，您可能会注意到某些配置选项以 [stuff]之类的结尾。括号中的字符和括号本身是可选材料，仅用于使用户更容易理解选项；只有前面左边的字符[用于标识配置选项。因此-0.mo 和-0.mo [nument]和 -0.monument 以及-0.moe_and_curly 都意味着完全相同：用户试图用下一个参数设置纪念碑名称。但是，与选项标志名称的其余部分一样，括号中只允许使用可打印字符；不允许有空格。

您可以将此配置信息重定向到文件，该文件称为配置文件：

```
teqc ++ config fbar0010.97o> my_obs_config
```

可以轻松编辑 此 ASCII 配置文件（即上例中的 my_obs_config）以包含（希望）正确的信息。对于熟悉 RINEX [OBS](#) 头字段的人来说，各种-0 标志的含义应该是相当明显的：

```
-OS [ystem]
```

卫星系统（G =仅 GPS，R =仅 GLONASS，E = Galileo，S =仅 SBAS 等，M =混合=多个系统）

```
-0.pr [ogram]
```

用于创建 RINEX 文件的程序（注意：将在 1999 年 1 月 7 日之后的第一个版本中过时并被忽略）

```
-0r [un_by]
```

程序用户名

```
-0D [吃]
```

程序执行日期（注：在 1999 年 1 月 7 日之后的第一版中将被淘汰并被忽略）

```
-oo [操作结束时切]
```

网站运营商的名称（观察员）

```
-0.ag [ency]
```

代理商的名称

```
-0.mo [nument]
```

纪念碑（标记）的名称

-0. mn

纪念碑（标记）编号

-0. rn

收件人号码

-0. rt

接收器类型

-0. rv

接收器软件/固件版本

-0. an

天线编号

-燕麦

天线类型

-0. px [WGS84xyz, M]

在 WGS84 笛卡尔坐标系中的近似地心位置，以米为单位

-0. pe [母鸡, M]

天线中心校正，以米为单位

-0c [omment]

原始标题评论（注意：使用+ 0c [omment]附加新评论）

-0. int [erval, 仲]

采样间隔，以秒为单位

-0. st [艺术]

第一次观察时期的日期和时间

-0E [ND]

上次观察时期的日期和时间

-0. def_wf

L1 和 L2 的默认波长因子（参见 teqc 处理[波长因子](#)的注释）

-0. obs [_types]

可观察量列表和文件数据部分中的可观察量本身

还有一些其他选项可用于输入信息，但永远不会使用+ config 或++ config 输出：

-0. dec [imate]

OBS 时期的模数抽取为#time units（默认为秒）； -0. dec 30 或-0. dec 30s 或-0. dec .5m 导致名义上在 00 和 30 秒输出的时期； 毫秒跳跃应该 自动计算

-0. pg [EO, DDM]

WGS84 地理坐标中的近似地心位置，十进制度的纬度和经度以及以米为单位的高程（此输入转换为 WGS84 笛卡尔坐标）

-0. sl [蚂蚁]

输入垂直顶心天线校正为倾斜高度，天线直径和垂直相位中心偏移（E 和 N 假设为零）（此输入转换为笛卡尔顶心校正 h 0 0）

+ 0C [omment]

添加一个新的注释字段（注意：您不能使用-0c [omment]更改现有注释）

-0. rename_obs

更改 RINEX [OBS](#) 标头中 #/ TYPES OF OBSERV 中可观察对象的字符标识，但不重新排列文件中的数据； 小心使用！

-O.mod_wf

设置一组特定 SV 的波长因子不同于默认波长因子（这不会出现在 RINEX [OBS](#) 标题中作为 WAVELENGTH FACT L1 / 2 记录;请参阅 teqc 处理[波长因子](#)的注释）

-O.mov [ing] 1

迫使 RINEX [OBS](#) 天线位置最初处于运动（流动）状态（特别是不管二进制标志是否进行转换）； 请注意，此选项有一个参数：1 表示打开，0 表示关闭

RINEX [NAV](#) 文件有一组类似的编辑/提取标志，你可以通过检查来获得：teqc ++ config fbar0010.97n | 更多

-Na- α

电离层 α 参数

-Nb 的 η

电离层 β 参数

-N.leap

UTC 时间模型的闰秒

-N.UTC

UTC 时间模型 A0 A1 tw

-N.pr [ogram]

用于创建 RINEX 文件的程序（注意：将在 1999 年 1 月 7 日之后的第一个版本中过时并被忽略）

-NR [un_by]

程序用户名

-ND [吃]

程序执行日期（注：在 1999 年 1 月 7 日之后的第一版中将被淘汰并被忽略）

-ns [ystem]

卫星系统（G = GPS, R = GLONASS, E = Galileo, S = SBAS 等）

-Nc [omment]

原始标题评论（注意：使用+ Nc [omment]附加新评论）

同样用于编辑你也可以使用

+ NC [omment]

附加一个新的注释字段（注意：您不能使用-Nc [omment]更改现有注释）

RINEX [MET](#) 文件有一组类似的编辑/提取标志，你可以通过检查来获得：teqc ++ config fbar0010.97m | 更多

-M.pr [ogram]

用于创建 RINEX 文件的程序（注意：将在 1999 年 1 月 7 日之后的第一个版本中过时并被忽略）

-mr [un_by]

程序用户名

-MD [吃]

程序执行日期（注：在 1999 年 1 月 7 日之后的第一版中将被淘汰并被忽略）

-M.obs [_types]

气象可观测列表和文件数据部分中的满足可观察量本身

-M.mo [nument]

纪念碑（标记）的名称

-M. mn

纪念碑（标记）编号

-mc [omment]

原始标题评论（注意：使用+ Mc [omment]附加新评论）

同样用于编辑你也可以使用

+ MC [omment]

附加一个新的注释字段（注意：您不能使用-Mc [omment]更改现有注释）

-M.rename_obs

更改 RINEX [MET](#) 标题中 #/ TYPES OF OBSERV 中的计量可观察量的字符名称，但不重新

排列文件中的数据；小心使用！

让我们回到 fbar0010.97o 的元数据。假设上面的配置文件 my_obs_config 现在包含更正的 RINEX [OBS](#) 字段，您可以执行（在 sh 或 ksh 中）：

```
teqc`cat my_obs_config` fbar0010.97o> temp001a.97o
```

或（使用另一个 teqc 命令行选项）

```
teqc -config my_obs_config fbar0010.97o> temp001b.97o
```

该-config this_config_file 指定正在输入称为一个配置文件 this_config_file。

最后两个示例命令之间存在重要差异，但此时不应该显而易见（即，执行 diff temp001a.97o temp001b.97o 应该表明两个修改后的文件是相同的）。这里介绍了 teqc 中使用的[选项分层过程](#)。为了充分利用 teqc 的功能，强烈建议用户最终熟悉这种[层次结构](#)。使用配置选项时要记住的主要规则是：

遇到的配置选项的第一个设置/值是使用的设置/值（忽略相同配置选项的后续设置/值）；除了三种特殊情况：输入多个配置文件，qc 模式的多个 [NAV](#) 文件以及其他 RINEX 注释。现在需要的只是知道处理配置选项的顺序。

处理的第一个配置选项是命令行选项，从左到右处理。例如，执行：

```
teqc -0.mo "foobar site" -0.mo foobar ++ config fbar0010.97o
```

在这里，命令行上有两个相同的配置选项-0.mo，用于更改标记名称，但有两个不同的参数。用哪个？答案是遇到的第一个，在这种情况下是左边的那个。为了说服自己，也尝试：

```
teqc -0.mo foobar -0.mo "foobar site" ++ config fbar0010.97o
```

在上面的示例执行中（使用 sh 或 ksh）：

```
teqc`cat my_obs_config` fbar0010.97o> temp001a.97o
```

在`猫 my_obs_config`接通配置文件的内容 my_obs_config 到命令行配置选项，所述第一线，其中（在该文件的顶部）变得等同于最左边的命令行选项，则进行到最后一行（在其中文件的底部）等同于最右边的命令行选项。

处理的下一个配置选项来自特殊的环境变量（如果存在），称为\$ teqc_OPT。实际上，可执行文件会查找与可执行文件名称匹配的环境变量。所以如果你做 cp teqc my_teqc 然后使用 my_teqc 作为可执行文件，在这种情况下，它将查找名为\$ my_teqc_OPT 的环境变量。

因此，如果设置\$ teqc_OPT（假设为 sh 或 ksh）：

```
export teqc_OPT = " - 0.mo foobar"
```

然后执行并比较


```
teqc ++ config fbar0010.97o
teqc -0.mo "foobar site" ++ config fbar0010.97o
```

您将看到第一种情况下的纪念碑名称设置为 foobar（使用环境变量\$ teqc_OPT）和第二种情况下的 foobar 站点（使用命令行选项）。

处理的下一个配置选项是所有指定的配置文件。以下列方式形成这些列表。首先，存储所有-config 命令行参数或\$ teqc_OPT 中的任何参数。然后（再次假设我们的可执行文件仍称为 teqc），查找环境变量\$ teqc_CONFIG，其中可能包含其他配置文件的名称或完整路径和名称。如果\$ teqc_CONFIG 如果存在，则它包含的名称将从命令行附加到此配置文件列表（如果有）的末尾。现在所需要的只是找到每个配置文件（如果存在），这是以下列方式完成的。

对于每个可能的配置文件名，首先检查名称，以查看它是否以/字符开头，假设是 UNIX 样式的目录命名约定。（使用 DOS 样式约定时，使用\；使用 MacIntosh 样式约定，使用 a : .）如果名称以/开头，则 teqc 假定配置文件名是绝对的，即它包含完整文件名前面的路径。在这种情况下，teqc 将只尝试这条路径和名称。如果文件存在且可以读取，则将其作为配置文件处理；如果它不存在或无法读取，teqc 将移动到列表中的下一个文件。

如果配置文件名不以/开头，则 teqc 假定配置文件名是相对的，即它仅包含部分路径和文件名，或者可能只包含文件名。此时，teqc 查找\$ teqc_PATH 环境变量，设置方法与其他\$ * PATH 环境变量相同，即使用：分隔不同的路径。如果这个\$ teqc_PATH 存在环境变量，其中的每个路径都用作相对配置文件名的前缀。如果找到可以读取的文件，则将其作为配置文件处理，然后 teqc 转到列表中的下一个配置文件。如果找不到或无法读取，teqc 会尝试将下一个路径作为前缀，依此类推。

如果\$ teqc_PATH 不存在，teqc 将始终尝试路径，即目前的工作目录。如果你使用\$ teqc_PATH，你可能想要确保这一点。是你的一条路；在这种情况下，teqc 不会自动为您包含它。

总而言之，处理配置选项的层次结构是：

1. 首先从左到右处理命令行选项
2. any -config 参数存储为配置文件列表
3. 如果存在，则根据与上面 1) 和 2) 相同的层次处理 环境变量\$ teqc_OPT
4. 如果存在，则将环境变量\$ teqc_CONFIG 附加到从命令行和环境变量\$ teqc_OPT 中提取的配置文件列表的末尾。
5. 然后处理累积的配置文件列表，从头到尾：
 - 如果配置文件的名称是绝对的，则仅检查该路径和名称
 - 如果配置文件的名称是相对的：\$ teqc_PATH 中的路径列表将被使用，直到找到并读取文件或者\$ teqc_PATH 中的所有路径都用完为止；如果\$ teqc_PATH 不存在，则只有相对路径。试过了。
 - 然后从左到右，从上到下处理每个配置文件
6. 以上未设置的任何配置选项在 teqc 可执行文件中都有一个默认配置选项，或者在输入 RINEX 或本机二进制格式（通过 stdin 或文件）的情况下，使用原始信息，或者如果该信息不存在，它是 null。

如果在 qc 模式下使用 teqc，则 [qc 短报告段](#) 和 [qc 长报告段](#) 中将包含已找到和读取的配置文件列表。这似乎是一种令人难以置信的不必要的可能性，但你可以放心，有一个原因。到目前为止，您只是暴露于-0.*选项，用于修改 RINEX [OBS](#) 文件中的标头字段。这些选项有 20 多种。RINEX [NAV](#) 文件有一组类似的-N.*

选项，RINEX [MET](#) 文件有-M.*选项。一般用于 teqc

有大约十几个不同的选项，质量检查模式（qc）有大约 7 打不同的选项。要将所有这些选项塞入一个文件（当然可以），如果您只想更改一个或两个参数，则会创建一个几乎无法管理的文件。由于 teqc 查看每个文件的方式，每个配置文件都不需要指定所有选项：更多是作为一组具有随机顺序的命令行选项而不是严格格式化的文件。如果想法尚未开始流淌如何利用这种灵活性，那么[脚本部分中](#)的示例将向您展示如何轻松使用此[选项层次结构](#)。

此外，你真的不需要使用任何配置文件（可能只是用作命令行选项的几个配置选项）都可以让 teqc 在大多数情况下产生合理的输出，如本文档中的第一个示例所示。

10 配置选项和命令行选项； 有什么不同？

从语法上讲，似乎所谓的配置选项和 teqc 的普通 UNIX 命令行选项之间没有区别。对于许多选项，实际上没有区别。这就是整个 teqc 界面设计的美妙之处！

但是，对于 teqc 配置选项的一个重要子集，存在一个重要区别。这些都是有论据是字符的字符串可能封装空白，如选择-O.mo 详细讨论[上面](#)。目前，这些只是-O.，-N.，和-M.选项将编辑 RINEX 标题字符字段，加上 + Oc，+ Nc 和+ Mc 选项以包含任何新的 RINEX 注释。这些选项的重要区别是：

- 如果文本字符串包含空格，则文本字符串必须由一组“（双引号）分隔”（尽管始终允许使用一组双引号分隔任何文本字符串）
- 如果文本字符串包含“（双引号）或\（反斜杠）字符本身，则每个实例必须在前面加上\，即\"或\\

使用+ Oc 选项，这里显示了一些示例（希望）澄清可以做什么。同样的操作方式与任何其他文本字符串选项，如发生-O.mo。

创建一个配置文件 comment_config，其中包含：

```
+ Oc "\"\"这是一个\"评论。\"\" \"\"
```

并执行

```
<prompt>
teqc -config comment_config RINEX_OBS | grep 评论

\"这是一个\"评论。\"\"评论
```

其中 RINEX_OBS 是任何 RINEX [OBS](#) 文件。或者使用命令行（例如使用 ksh）：

```
<prompt> teqc + Oc "\"\"这是一个\"评论。\"\"\"\" \"\"RINEX_OBS | grep 评论

\"这是一个\"评论。\"\"评论
```

请注意，由于 shell 对两个反斜杠的元字符解释，因此需要两个额外的反斜杠。这里需要类似的反斜杠 shell 解释：

```
<prompt> teqc + Oc long = 123W34 \ '45 \"RINEX_OBS | grep long =
```

```
long = 123W34'45"评论
```

其中必须包含一个额外的\（反斜杠），而不是 teqc，但是要抑制 shell 的'（单引号）的元字符含义。（在后一个示例中，请注意文本字符串不是由一组“（双引号）”分隔，因为文本字符串中不包含空格。）或者使用\$ teqc_OPT，执行（例如使用 ksh）：

```
<prompt> export teqc_OPT="+ Oc \"\\\"\\\"\\\"这是一个\\\"\"注释。\\\"\"\\\"\"\\\"\"\\\"\"\\\"\"\\\"\""
```

```
<prompt> echo $ teqc_OPT
```

```
+ Oc\"\"\"\\\"这是一个\\\"\"评论。\\\"\"\\\"\"\\\"\""
```

```
<提示> teqc RINEX_OBS | grep 评论
```

```
\"\\\"这是一个\"\"评论。\"\"\"评论
```

其中很多都需要额外的反斜线用反斜杠元字符解释由 shell 来建立环境变量的正确值\$ teqc_OPT。

尝试时会出现一个小缺陷：

```
<prompt> unset teqc_OPT
```

```
<prompt> teqc`cat comment_config` RINEX_OBS | grep 评论
```

```
\"\\\"这是一个\"\"评论。\"\"\"评论
```

（初始的 unset 命令只是消除\$ teqc_OPT，如果它有一个值。）请注意，原始注释字符串中的每个多个空白区域都会折叠到 RINEX 注释中的单个空白区域。（在 teqc 解释 cat 命令期间发生空白崩溃。）在这种情况下，这个结果是 UNIX shell 工作方式可以实现的最佳结果。 要完成 teqc 接口的对称性，使用+ config 或++ config 选项时，RINEX 文本字段中的任何现有“（双引号）或\字符都会扩展为”

或者在结果配置输出中输入\\。通过这种方式，配置输出可立即用作有效且完整的 teqc 配置文件。

11 使用 teqc 进行质量检查（qc）模式

teqc 的很大一部分用于卫星定位数据的质量检查或 qc -ing。随着 2018 年的 teqc 发布，这包括使用来自 NAVSTAR GPS, GLONASS, Galileo, Beidou / Compass, QZSS, IRNSS 和/或 SBAS 中的任何一个的广播导航消息的轨道计算 - 基本上是 GNSS 的任何组合。

要在 qc 模式下使用 teqc，请尝试，例如：

```
teqc + qc fbar0010.97o
```

（假设该 RINEX 时刻 [NAV](#) 文件 fbar0010.97o 是不是 在当前目录中）。首先，注意+ qc 选项：即打开 qc。接下来请注意，只有 RINEX [OBS](#) 文件作为目标文件。你在这里做的是运行 qc -lite 模式，即没有任何卫星定位信息。

如果 fbar0010.97o 是有效且完整的 RINEX [OBS](#) 文件，执行上述操作应该会生成一些内容。究竟是什么（此时）应该只取决于 teqc 中的默认 qc -mode 设置顺便提一下，这应该代表大多数用户对常规 qc 处理的期望。但是，几乎所有 qc 参数都可以通过适当的配置选项来实现。是的，您可以通过以下方式查看 qc 配置选项以了解您可以产生的影响：

```
teqc + qc ++ config 2> / dev / null | 更多
```

（注意：请注意这与执行 teqc ++ config 2> / dev / null 有何不同。简而言之，使用+ qc 选项，您打开 teqc 的 qc 模式并使用 ++ config 请求完整配置选项集，所以现在你得到了一般 teqc 选项之后的所有 qc 模式选项。另外，使用了更多的管道，因为有很多 qc 选项！）

现在让我们来看看 teqc + qc fbar0010.97o 的执行情况（也许随后是更多的管道）终于产生了。假设运行成功，用户应该至少注意到两件事：1）执行期间应该有一组字符进入屏幕（实际上是 stderr）

```
qc lite >>>>>> ...
```

（可能还有其他一些非关键的 stderr 消息）。这只是 teqc 如何在观察时期中行进的线性模拟指示器，其中最终指标的 length 与 ASCII 时间图的“length”匹配（默认为 72 个字符）。2）ASCII 时间图有一个屏幕转储（实际上是 stdout），最后是一些简单的统计信息。（注意：stdout 的后一部分大致相当于原始 UNAVCO QC 程序写入* .YY S 文件的内容。）此外，将来，转储到 stdout 的内容可能会发生变化，尽管这可以设置为一个或多个配置标志。接下来，看看结果

```
teqc + qc ++ config 2> / dev / null | grep 报告
```

结果应该是这样的：

```
+ l [ong_report]
```

```
+ s [hort_report]
```

如果其中任何一个配置选项实际上以+开头，那么您的目录中应该有一个 fbar0010.97S 文件，这是 fbar0010.95o 上的 [qc 报告文件](#)。此报告有两个可能的细分：可能的[简短报告细分](#)，然后是可能的 [长报告细分](#)。你应该有 [短段](#)，如果你的配置说+ S ...，如果你的配置说，它应该是不存在的-s ...。目前， [短报告段](#)与转储到 stdout 的信息相同。您应该拥有[较长的报告细分](#) 如果您的配置显示+ l ...，如果您的配置显示-l ... 则应该不存在。（[长报告段](#)取代了原来的 UNAVCO 质量控制中的 stdout 。）

接下来，看看结果

```
teqc + qc ++ config 2> / dev / null | grep 情节
```

如果结果是

```
+情节
```

那么你可能有一个或多个绘图文件（采用 UNAVCO COMPACT 格式），例如：

fbar0010.ion 如果+离子被设置（电离层延迟可观察，以米为单位）

fbar0010.iod 如果设置+碘（电离层延迟可观察的导数，以米/分钟为单位）

fbar0010.mp1 和 fbar0010.mp2 如果设置了+ mp（MP1）和 MP2 可观察量，以米为单位）

fbar0010.sn1 和 fbar0010.sn2 如果设置了+ sn（L1 和 L2 的 S / N，任意单位）

可以使用 UNAVCO 的 gt 或 qcview 可执行文件以图形方式查看这些文件。使用-plot 的配置选项可以抑制所有打印文件。您还可以通过使用+ tec 将电离层延迟可观察及其导数转换为 TEC（总电子计数== 1e16 电子/平方公尺）及其衍生物的相对变化。

接下来，在您的工作目录中移动或创建 fbar0010.97n 并执行：

```
teqc + qc fbar0010.97o
```

要么

```
teqc + qc -nav fbar0010.97n fbar0010.97o
```

在这里，您要输入一个附加文件，默认情况下在第一个示例中，或者使用`-nav` 配置选项，后跟 RINEX [NAV](#) 文件的名称显式输入。您现在以 `qc -full` 模式运行 `teqc`，即存在卫星定位信息，如果存在足够的信息（卫星星历表，加上接收器 P1, P2 或 C / A 代码），则尝试天线的位置。没有其他标志可以使用 `qc -full` 模式。如果您提供二进制数据或 [NAV](#) 文件，或者 `teqc` 自动为提供的 [OBS](#) 找到匹配的 [NAV](#) 文件名文件名，您正在使用 `qc full`；如果您不提供 [NAV](#) 文件并且没有匹配的 [NAV](#) 文件，那么您使用的是 `qc lite`。如果您提供二进制数据，则没有先验方法来确定是否存在星历记录而不首先读取整个输入，因此在这种情况下假设 `qc -full`。

用于从 [OBS](#) 文件名构建匹配 [NAV](#) 文件名的区分大小写的算法如下，其中 YY 表示两位数，例如 YY == 97：

```
*.YY o ->寻找*.YY ñ
*.YY O ->寻找*.YY N.
*.obs ->查找*.nav
*.OBS ->查找*.NAV
```

此 [NAV](#) 文件自动匹配的其他情况可以轻松添加到代码中。

在执行期间（可能需要比 `qc -lite` 运行稍长的时间），您现在应该看到：

```
qc full >>>>>> ...
```

去 `stderr`（再次，可能还有其他 `stderr` 东西混入）。假设成功运行，发生的一般事情应与 `qc -lite` 运行时发生的事情类似，扩展为包括有关卫星位置（如高程）和天线位置的信息。假设+绘图配置，最大的附加项是附加的 COMPACT 绘图文件：

```
fbar0010.azi 和 fbar0010.ele（从 NAV 文件中存在星历的每颗卫星的天线观察卫星地心中心方位角和仰角）
```

其中一个基本设计标准是，当星历可用时，让 `teqc` 在每个卫星的基础上从 `qc -lite` 操作模式动态切换到 `qc -full` 操作模式。这对于分析实时数据流尤为重要。对于正常的“后处理”（即，文件可用），适用相同的方案。如果从供应 [NAV](#) 文件中没有可用于特定卫星的星历表，但是存在对该卫星的观测，则使用 `qc -lite` 模式完全处理观测值，即使 `qc -full` 模式通常是开启的。报告中提供了适当的指标，表明发生了这种情况。

12 使用 `teqc` 与多个文件输入或文件名包括

在您想要包含多个文件作为输入的情况下，总会出现这种情况，让我们看看其中一些案例。首先，可能有多个配置文件：

```
teqc -config my_config_1 -config my_config_2 -config my_config_3 {rest of command}
```

工作得很好，完全等同于：

```
teqc -config my_config_1, my_config_2, my_config_3 {rest of command}
```

要么

```
teqc -config my_config_1, my_config_2 -config my_config_3 {rest of command}
```

要么

```
teqc -config my_config_1 -config my_config_2, my_config_3 {rest of command}
```

请注意，如果在`-config` 选项标志之后将它们用作单个参数，则使用默认的逗号分隔符来分隔不同的配置文件名。选择了逗号分隔符，因为它通常不是 shell 元字符，很少用作文件名的一部分。如果在某些情况下，逗号最终成为输入文件名称的一部分，则在使用该文件名之前必须使用`-delim` 选项。例如，假设配置文件名为

strange, 一个存在, 并且由于某种原因要么你不能或不能将它重命名为不包含逗号名称, 但你想使用它并且你没有'知道 UNIX ln。好吧, 要做到这一点, 你可以使用

```
teqc -delim = -config strange, one {rest of command}
```

在这里你已经将分隔符更改为= (等号), 所以现在很奇怪, 一个被解释为单个配置文件名, 而不是两个叫做奇怪和一个。但是要预先警告: 就像其他大多数事情一样, 你只需要更改一次分隔符。

相同类型的方案适用于为 qc -full 运行输入多个 NAV 文件:

```
teqc + qc -nav fbar0010.97n -nav fbar0020.97n -nav fbar0030.97n {rest of command}
```

是完全相同的

```
teqc + qc -nav fbar0010.97n, fbar0020.97n, fbar0030.97n {rest of command}
```

回想一下配置文件的排序选项层次结构: 首先处理左侧的文件。为了使用+ qc 选项输入多个 NAV 文件, teqc 对每个 SV 的星历表进行排序, 因此确实没有必要保持任何特殊排序, 但如果你遵守 TOW 时间, 初始设置会稍快一些 -顺序多个 NAV 文件输入。目前, 命令行末尾的所有目标文件名都被认为是 teqc 单次执行的一部分。例如, 执行此操作时您应该期望什么:

```
teqc + qc -nav fbar0010.97n fbar0010.97o fbar0020.97o fbar0030.97o
```

你用+ qc 打开了 qc 模式。由于您使用-nav 提供 NAV 文件, 因此这将是 qc -full 运行。假设每个 OBS 文件都是 24 小时 (并且没有遇到任何错误), 您将收到的是从 1997 年 1 月 1 日第一次观察到结束的整个 3 天观察期的 qc 报告。1997 年 1 月 3 日的最后一次观察。(来自 fbar0010.97n 的星历信息可能会在 3 日之前有点过时, 但这对于这次讨论是可以接受的。)你不会得到什么 (如 teqc 目前已实施) 是三个单独的报告, 可能还有三组绘图文件, 即每个提供的 OBS 文件的一个报告 (和一组绘图文件)。

13 使用 teqc 进行时间窗口化。

如上面第 9 节所述, teqc 总是窗口输入数据 (介于 1980 年 1 月 1 日到 6075 年 12 月 31 日之间), 尽管这通常对用户是透明的。有八种不同的开窗策略供您注意; 提供 (或不提供) 不同的信息以使用这些不同策略中的每一种。

下表中的术语如下。括号内的值是由目标文件中的 teqc 确定的值 (即隐含的), 而非括号内的值由用户显式提供 (通过使用命令行的配置选项, \$ teqc_OPT, \$ teqc_CONFIG 或其他配置文件))。开始指的配置选项标志的参数-st [art_window], 增量是指配置选项标志的参数+ d ...或-d ..., 和结束指的是配置选项标志的参数-e [nd_window], dir 指的是+或-是否与 delta 配置标志一起使用。八种不同的窗口选项是:

1. [开始] [结束] (用户提供除目标文件以外的任何内容)
2. [开始] delta (dir == +) 例如+ dh 7 从开始起 7 小时
3. delta [end] (dir == -) 例如-dm 60, 距离结束 60 分钟
4. 开始 [结束]
5. [开始] 结束
6. 开始 结束
7. 开始 delta (dir == +或-)
8. delta end (dir == +或-)

其中缺失值由 teqc 计算。让我们来看看这些案例的含义, 你会发现这比最初出现的要简单得多。

对于情况（1），可能是最常见的情况，用户只提供目标文件列表。然后，teqc 快速搜索要处理的目标文件，以确定开始和结束时间。如果输入尚未写入快速搜索算法的文件类型或使用 stdin 作为目标，则必须使用显式窗口（case（6），（7）或（8））。目前，快速搜索算法仅针对 RINEX [OBS](#)，[NAV](#) 和 [MET 实现](#) 文件。开始时间将是列出的第一个目标文件中的第一个观察时期或事件时期（RINEX [OBS](#) 或 [MET](#) 文件）或第一个 TOE 时期（RINEX [NAV](#) 文件），结束时间将是列出的最后一个目标文件中的最后一个时期。（[如上所述](#)，如果您的目标文件没有按时间顺序列出，或者您的一个或多个目标文件不是内部时间顺序的，那么某些内容将在执行过程的后期发生，并且 teqc 会告诉您它。）

作为一个例子，让我们回到：

```
teqc + qc -nav fbar0010.97n fbar0010.97o fbar0020.97o fbar0030.97o
```

这里要处理的目标文件是 fbar00 * 0.97o。快速搜索查看 fbar0010.97o 的开头以确定窗口的开始时间，并查看 fbar0030.97o 的结尾以确定窗口的结束时间。你不需要担心这里的任何事情。

对于情况（2）和（3），用户提供带有参数的 + d ... 选项或 -d ... 选项。目前，..... 可以是 Y 年，M 表示月，d 表示天，h 表示小时，m 表示分钟，或 s 表示秒。领先+或者- 分别表示“从一开始就给我一个正时间增量”或“从结尾给我一个负时间三角洲”。让我们再次假设上面的 fbar00 * 0.97o 文件每个都是 24 小时的数据。然后配置选项 + dh 6 和文件 fbar0010.97o 将意味着：“从 1997 年 1 月 1 日 00: 00: 00.0 开始，到 1997 年 1 月 1 日 06: 00: 00.0 结束”，即距离（隐含）窗口启动。配置选项 -dh 6 和文件 fbar0010.97o 将意味着：“从 1997 年 1 月 3 日 18:00 00:00 开始，到 1997 年 1 月 4 日 00: 00: 00.0 结束”，即距离的 -6 小时的增量（暗示）窗口结束。

对于情况（4）和（5），使用配置选项 flags -st 或 -e 和参数，显式提供新的部分或完整的开始时间或仅部分或完整的结束时间。完整的开始或结束时间的论据很容易理解；它具有的格式 [YY] YYMMDDHHMMSS [.sss ...]，虽然元字符像：（结肠），（逗号），；（分号），/（斜杠），\（反斜杠），+（加号），-（减号），=（等于），_（下划线），~（tilde），#（磅）可用于提高可读性。即

```
19970229105937
```

```
970229105937
```

```
970229105937.000000
```

```
1997-02-29_10: 59: 37
```

所有意思是“1997 年 2 月 29 日 10: 59: 37.000000”，因为基准年是 1980 年（参见执行 teqc ++ 配置的结果：1980 年的基年意味着所有两位数的年份都假设发生在公元 1980 - 1979 年）注意：始终确保包含月份编号，日期编号，小时，分钟和秒的两位数字。

让我们假设我们回到我们的命令

```
teqc + qc -nav fbar0010.97n fbar0010.97o fbar0020.97o fbar0030.97o
```

但想在“1997 Jan 1 00: 09: 30.004”开始时间窗口。您可以使用配置选项 -st 970101000930.004，虽然有点麻烦，但事情会正常。事实上，有另一种方法可以使用部分参数格式（或“蒙版”格式）来使用 -st 选项，例如

```
-st 930.004
```

```
-st 9: 30.004
```

```
-st 00: 09: 30.004
```


所有这些都被解释为同一件事：9 分 30 秒。在这里，teqc 认识到您只提供部分开始时间（在这种情况下使用分钟和秒）。然后使用快速搜索算法获得实际开始时间（如果情况（1）或情况（2）），然后将掩码覆盖应用于开始时间并插入部分开始时间（仅更改分钟和秒，在这种情况下）。所以-st 或-e 的参数格式更接近：

```
[[[[[YY] YY] MM] DD] HH]毫米] SS [.sss ...]
```

换句话说，假设您提供整秒，如果需要，您可以进一步指定小数秒。秒数左侧的数字被解释为分钟，小时等。

对于情况（6），（7）和（8），您明确地指定了感兴趣的窗口，并且，再次，部分参数可以用于开始和/或结束时间。您可以提供显式（部分或完整）开始和结束时间，显式（部分或完整）开始时间与增量（a-la 情况（2）和（3）），或显式（部分或完整）结束时间有一个三角洲。

通过使用 case（7），您可以轻松强制您的 teqc 无论输入目标文件的开始和结束时间如何，处理都在一天的同一时间开始并跨越相同的时间长度。例如，假设您希望每天的开始时间为 00: 00: 00.0000000，并且需要 24 小时的 teqc 处理。这可以通过类似的方式轻松完成

```
-st 00:00:00 + dh 24 。
```

时间窗口的开始和结束时间都包括在内，即这些时间包含在窗口中。因此，如果在前面的示例中您想要提取短时间 24 小时，其中采样间隔可能是 30 秒，那么您可以使用

```
-st 00:00:00 + dm 1439.5
```

其开始时间为 00: 00: 00.0000000，结束时间为 23: 59: 30.0000000。

您可能会认为几乎所有可能的窗口可能性都已被涵盖。好吧，还没有。您甚至可以在整个时间窗口中引入漏洞，在此期间您对任何事情都不感兴趣。teqc 跳过指定窗口孔，包括孔边界期间发生的所有输入。这里使用配置选项-hole 后跟文件名。命名的文件是一个 ASCII 文件，列出了您在时间窗口中需要的孔。每个窗口孔的孔文件格式始终为

```
[YY] YY MM dd hh mm ss [.sss] [YY] YY MM dd hh mm ss [.sss]
```

其中只需要存在空白作为分隔符。因此，文件：

```
97 2 15 00 00 00 1997
```

```
2 15 03 00 00.00
```

```
97 2 15
```

```
06 00 00 97 2 15 09 00 00
```

完全没问题（尽管人类有点难以阅读）。一个更可读的文件（对于人类）做同样的事情是：

```
97 2 15 00 00 00 97 2 15 03 00 00
```

```
97 2 15 06 00 00 97 2 15 09 00 00
```

显然，这是每个所需孔的开始和结束时间列表。你可以在文件中列出多少个洞？尽可能多的（或直到计算机内存饱和）。对于每个 teqc 运行，您使用-hole 选项命名了多少个文件？目前，有一个，所以要确保它包含你想要执行的所有漏洞。还记得：每个洞都包括洞的确切开始和停止时间。

时间分配与选项 tbin

您还可以使用选项-tbin 调用时间分级样式的自动窗口。该-tbin 选项仅在版本 TEQC 2008 年 7 月 18 日或之后作出。

这是一个功能强大的选项，可以创建各种长度的 RINEX 或 BINEX 批量输出文件，例如每天，8 小时，每小时，30 分钟，15 分钟，5 分钟.....，来自一个或多个输入文件（制造商的格式，RINEX 或 BINEX，和往常一样，

输入文件都必须是相同的类型)。该-tbin 选项可以创建比输入文件(例如,从日常文件每小时文件)较小的输出文件,而且还从几个较短的文件,使长文件。它可以使用混合长度的输入文件。因此,输入可能是不同长度的文件列表,有些短于一天,输出可以分类到每日文件中。

两个选项用于时间分级。第一个和必需的选项是-tbin,它有两个参数,例如-tbin 1h myfile。第一个和必需参数(例如1h)是每个bin(输出文件)的持续时间。尾随字母指定时间单位:秒为秒,m为分钟,h为小时,d为天。您必须使用 d, h, m 或 s。允许以亚秒为单位的持续时间:如果指示了亚秒,则生成的文件名将扩展为在秒值显示时间到毫秒之后包含额外的“.ddd”。允许持续时间的任何值:警告 emptor! 如果输入1并且输入数据大部分时间都是一天,则需要生成数万个文件。

-tbin 的第二个参数,例如myfile,是输出文件名的前缀部分。前缀可以是4-char ID,也可以是其他一些没有空格的可打印字符串。

输出文件名的格式为<root> == prefix + doy-of-year, 其中文件名本身为:

a) <root> 0.yyo 如果 delta (bin 持续时间) 以天为单位是整数

b) <root> a.yyo - <root> x.yyo 如果 case (a) 不起作用,但 24 / (持续时间以小时为单位) 导致整数小时,小时由 a 到 x 的字母表示: a =以小时 0 开始, ..., x =小时 23 开始

c) <root> [ax] 00.yyo - <root> [ax] 59.yyo 如果情况 (a) 和 (b) 不起作用,但 60 / (以分钟为单位的 delta) 导致整数 00 = start 在分钟 0, ..., 59 =分钟 59 开始

d) <root> [ax] [00-59] 00.yyo - <root> [ax] [00-59] 59.yyo 如果情况 (a) - (c) 不起作用,但是 60 / (delta) 以秒为单位) 产生一个整数(或者也可能是所有剩余的情况.....) 00 =从第二个开始 0, ..., 59 =在第二个开始 59

在像 teqc 这样的命令中

```
teqc -tr d + obs ++ nav -tbin 1h mytbinfile inputdatafile
+ obs 和+ nav 之后 的额外裸 +符号意味着制作一系列的 obs 和 nav 文件,这些文件与使用-tbin 1h temp
参数创建的 1 小时时间段相匹配。在+后论点+观察和+导航选项使用的时候才有意义-tbin 选项。如果你在没有
-tbin 的情况下尝试了,你最终会得到一个名为+的文件。
```

选项+ nav +, +表示为时间仓制作单独的 NAVSTAR GPS 和 GLONASS 导航文件,但此语法也可以扩展为包括 SBAS, Galileo, Beidou / Compass, QZSS 和 IRNSS - 按此顺序。

例如,从 Trimble 转换为 RINEX 的命令

```
teqc -tr d + obs ++ nav myfull.nav -tbin 1h myhourly grnr2600.dat
将会把所有的 GPS 导航信息到一个文件 myfull.nav, 但 RINEX OBS 被 tbinned 为每小时文件:
TEQC: 创建文件 myhourly260a.97o ...
TEQC: 创建文件 myhourly260b.97o ...
TEQC: 创建文件 myhourly260c. 97o .....
```

如果 tbin 窗口短于约 2-4 小时,则此示例可能是您想要的。回想一下,当输入文件是 Trimble DAT / ION / EPH / MES 下载文件集(dat 文件)文件时,不需要-tr d 部分。

还有第二个选项-ast,对齐的开始时间,可以与-tbin 一起使用以指定第一个 bin 的开始时间。时间分级不需要此选项。如果未指定-ast 选项,则时间分级使用从第一个数据(obs, nav 或 met)开始的 00:00:00 的默认对齐方式。选项-ast -或-ast _表示开始对齐输出的第一个纪元; -ast [YYYY YY MM DD hh mm ss [.sssss]]表示在指定时间开始对齐。亚秒可以与-ast 一起使用。

请注意，常用且不同的选项-st 保留用于表示何时启动数据，默认情况下是找到的第一个纪元。不要混淆-ast 和-st。

选项 tbin 适用于所有这些情况：

N 制造商的格式或文件 BINEX -> MS RINEX 文件集

相同类型为 N RINEX 文件->同类型作为输入的 M 个 RINEX 文件

N 制造商的格式或文件 BINEX -> MS BINEX 文件

N RINEX 文件集-> MS BINEX 文件

与 RINEX 文件命名方案 TBIN 是：

每日= 0。[onghem]

每小时=[斧头]。[onghem]

分钟=[AX] 00。[onghem] - [AX] 59。[onghem]

second =[AX] [00-59] 00。[onghem] - [AX] [00-59] 59。[onghem]

subsec =[AX] [00-59] 00.000。[onghem] - [AX] [00-59] 59.999。[onghem]

哪里

- 用户提供

==每年的日子

[ax] 00 - 23 小时

== year modulo 100

[onghem] - RINEX 后缀，例如 RINEX obs 文件的“o”等。

请注意，teqc 应选择最粗糙的文件分区，以便执行您所要求的操作。使用哪个 filenames binning 取决于所选的-tbin 单位或-ast 时间单位。例如，如果使用默认的-ast（即未明确指定）并使用-tbin 30m test，则使用分钟 filenames binning；但如果-tbin 1h 测试或-tbin 60m 测试，则使用每小时文件分组。

为了创建具有时间分级功能的 RINEX，您可以很好地控制最终使用时间分区名称创建的文件；例如：

+ nav temp.gps, temp.glo + obs + -tbin 1h temp ==所有 GPS 导航消息都进入 RINEX 文件 temp.gps，所有 GLONASS 导航消息都进入 RINEX 文件 temp.glo（换句话说，两者都没有时间分级），但所有 RINEX obs 文件都有 时间限制（在这种情况下为 1 小时）

+ nav +, temp.glo + obs + -tbin 60m temp ==所有 GLONASS 导航消息都进入 RINEX 文件 temp.glo，但所有 RINEX GPS 导航文件和所有 RINEX obs 文件都是时间分区（再次，1 小时）

+ nav +, + + obs + + met + -tbin 3600s temp ==所有 RINEX（GPS 和 GLONASS nav, obs 和 met）文件都是时间分区（再次，1 小时）

+ tbin 3600s temp 是上一个命令的简写（注意+ in + tbin），即所有 RINEX 文件都是时间分区的。

创建时间限制 BINEX 的唯一区别是使用+ binex 选项（它采用其通常的参数），并且生成的 BINEX 文件的命名方式与时间限制的 RINEX 不同：

<root> =前缀+ GPS 周+'_' + day_of_week（太阳= 0，...，周六= 6）

daily = <root> .bnx

hourly = <root> [ax] .bnx

minute = <root> [ax] 00.bnx - <root> [ax] 59.bnx

second = <root> [ax] [00-59] 00.bnx - <root> [ax] [00-59] 59.bnx

subsec = <root> [ax] [00-59] 00.ddd.bnx - <root> [ax] [00-59] 59 .ddd.bnx

tbin 用法示例:

1) 您希望将输入分解为每日文件, 并将创建的 obs 文件名以 test 开头:

```
teqc -tr d + obs + -tbin ld test input.obs
```

例如, 如果数据从 2007 年开始: 165, 输出文件将命名为:

test1650.07o

test1660.07o

test1670.07o

test1670.07o

...

没有导航文件。

2) 您希望将输入分解为 1/12 恒星日的文件, 文件名以 sidx 开头, 并且您希望第一个文件的时间与第一天数据的 00h30m09s (通常是 GPS 时间) 对齐, 并且您希望数据从第一个文件中的 01h30m10s 开始; 使用:

```
teqc -tr d + obs + -tbin 7180.3409 sidx -ast 00: 30: 09.000 -st 01:30:10 input.obs
```

如果数据在 2007 年的一天开始: 165, 则输出文件将被命名为: sidx165a3009.07o (但数据在这里没有文件启动, 直到 1 点 30 分 10 秒)

sidx165c2950.07o

sidx165e2930.07o

...

注意恒星日是 23h 56m 4.091s, 所以 1/12 是 7180.3409 秒。

3) 输入一个 Topcon TPS 文件, cp_1p.tps 命令

```
teqc cp_1p.tps
```

将输入转换为 RINEX 作为单个流到 stdout。要创建每小时时间分区的 RINEX obs 文件, 以及 GPS 和 GLONASS 的不同导航文件, 请使用:

```
teqc + nav +, + + obs + -tbin 1h temp cp_1p.jps
```

创建 temp027m.03n ...

创建 temp027k.03o ...

创建 temp027k.03g ...

创建 temp027l.03o ...

创建 temp027l.03g ...

创建 temp027o.03n ...

创建 temp027m.03o ...

创建 temp027m.03g

4) 如果您希望导航消息在单个 RINEX 导航文件 (temp0270.03n) 中用于 GPS, 单个 RINEX 导航文件 (temp0270.03g) 用于 GLONASS, 则需要时间分段制作 1 小时的 obs 文件:

```
teqc + nav temp0270.03n, temp0270.03g + obs + -tbin 1h temp cp_1p.jps
```

创建 temp027k.03o ...

创建 temp027l.03o ...

创建 temp027m.03o ...

创建 temp027n.03o ...

创建 temp027o.03o ...

创建 temp027p.03o ...

5) 使用+ tbin 短手

```
teqc + tbin lh mytbinfile grnr2600.dat
```

是一个简称

```
teqc + nav +, ++ obs ++ met + -tbin lh mytbinfile grnr2600.dat
```

注意+ tbin 而不是-tbin。

6) 带有+ tbin 的BINEX 示例

```
teqc + binex 0x7f-03 + tbin 15m tmp input.obs
```

！注意！使用 RINEX OBS 默认的可观察列表

！注意！使用 RINEX MET 观测到默认列表

TEQC: 创建文件 tmp1488_1a00.bnx ...

TEQC: 创建文件 tmp1488_1a15.bnx ...

TEQC: 创建文件 tmp1488_1a30.bnx ...

TEQC: 创建文件 tmp1488_1a45.bnx ...

如果输入目标文件是 RINEX，则在新版本的 teqc 中使用-tbin 选项不需要 选项+ obs, + nav 或+ met（但如果输入是多个文件，则输入文件仍然具有是相同的类型）。这个简化的命令语法选择将在 2009 年 9 月之后的下一个完整版本中。

14 用 teqc 拼接

回想一下执行命令

```
teqc fbar0010.97o
```

基本上把 fbar0010.97o 的内容吐回到 stdout。假设您有 1997 年 1 月 1 日的 RINEX [OBS](#) 文件 fbar0010.97o 和 1997 年 1 月 2 日的 fbar0020.97o，并且您希望将它们组合成单个 RINEX [OBS](#) 文件。如果编写了 RINEX 标准，那么两个 RINEX 文件可以简单地相互连接以生成一个新的有效 RINEX 文件，即 UNIX cat 系统命令，这本来很容易：

```
cat fbar0010.97o fbar0020.97o> oops0010.97o
```

但是，唉，RINEX 标准不允许这种明显的简单性，因此文件 oops0010.97o 通常是无用的。

但是，teqc 负责处理两个文件之间的 RINEX 特性边界。从而

```
teqc fbar0010.97o fbar0020.97o> good0010.97o
```

或使用正则表达式（大多数 UNIX shell）

```
teqc fbar00 [12] 0.97o> good0010.97o
```

生成一个有效的 RINEX 文件，good0010.97o，并在边界添加注释：

RINEX FILE SPLICE 评论

（注意：此拼接注释仅出现在拼接的 RINEX [OBS](#) 文件中，因为当前的 RINEX 标准不允许在 RINEX [NAV](#) 和 RINEX [MET](#) 文件的标题之后进行注释。）多个文件可以拼接在一起，其中任何一个都可以用于任何会话长度。但是，顺序（像往常一样）必须是时间顺序的。在第一次开启之后来自文件的头信息被选中以仅保留相关部分，并且可以通过包括-phc = delete post-header comments 选项来进一步减少这一点，例如

```
teqc -phc fbar00 [12] 0.97o> good0010.97o
```

接收器时钟复位信息不会通过 RINEX [OBS](#) 文件的接头边界传输。因此，如果在第一个文件 [OBS](#) 文件中存在毫秒接收器时钟复位，并且第二个 [OBS](#) 文件将这些毫秒复位初始化回零，则在 [OBS](#) 接头的边界处将存在 n 毫秒接收器时钟跳转。

如果需要，可以在单个命令中组合窗口和拼接操作。将任何[窗口选项](#)与拼接过程结合使用。

15 转换 teqc

teqc 正在被增强，以处理来自各种接收器的许多本机二进制格式。目前，teqc 处理许多双频（L1 和 L2）和一些单频（L1）接收器的通用格式。teqc 对所有本机二进制格式的一般用法是类似的。您需要指定三件事：

1. 一般类型的接收器
2. 来自此接收器的一般类型的本机二进制格式，
3. 你有兴趣提取什么。

（不同二进制格式的 big-endian / little-endian 问题由 teqc 自动处理，所以不要担心。）

Teqc 还读取非本机格式，目前仅限于 RINEX 格式，ARGO 格式和 BINEX。您可能已经确定了，默认情况下会采用 RINEX 格式。要强制 teqc 解释其他非本机格式的输入，请使用：

- -binex 用于 BINEX
- - IGS RTigs 格式的标题
- -soc 为 JPL Soc 格式
- -argo 为 ARGO 格式

（注意：RINEX 没有相应的-rinex 标志，因为它始终被认为是默认值。）

对于本机或接收器特定格式，可能需要选项标志来指定接收器的一般类型，其参数用于指定接收器格式：

- -as for Ashtech
 - d - B / E / S / D 下载文件集（需要 B 文件）
 - s - RS-232 流格式
 - r - R 文件
 - u - U 文件（注意：对于 U 文件，总是需要-ash u）
- -Leica for Leica
 - lb2 - LB2
 - mdb - MDB
 - d - DS 下载文件集（需要 OBS 文件）
- -topcon for Topcon
 - tps - TPS 格式
- -Javad for Javad
 - jps - JPS 格式
- -septentrio for Septentrio
 - sbf - Septentrio 二进制格式
- -nct for Navcom Technology
 - b - Navcom 二进制格式
- -tr for Trimble
 - d - DAT / ION / EPH / MES 下载文件集（需要 dat 文件）

- s - RS-232 RT17 流格式
- tsip - TSIP
- -aoa 或 -jpl 用于 TurboRogue / TurboStar 或 Benchmark 接收器
 - cb - ConanBinary
 - tb - TurboBinary
- -cmc for Canadian Marconi Corporation
 - 全明星 - 全明星格式
- -ublox for u-blox
 - ubx - UBX 格式
- -摩托罗拉的摩托罗拉
 - oncore - Oncore 格式
- 罗克韦尔的岩石
 - z - 黄道带格式
- -ti for Texas Instruments
 - g - TI-4100 GESAR 和 BEPP / CORE 格式
 - rom - TI-4100 ROM 格式

为了转换为 RINEX，用户可以指定主要感兴趣的类型文件；如果未指定，则假定为 RINEX [OBS](#)。例如，使用 receiver 参数（即格式规范）或在格式规范的末尾附加 o 意味着默认情况下提取 [OBS](#)，依此类推：

- -tr d 或 -tr -do: Trimble DAT 到 RINEX [OBS 的转换](#)
- -aoa cbn: 将 ConanBinary 转换成 RINEX [NAV](#)
- -ash rm: 将 Ashtech R-file 转换成 RINEX [MET](#)

例如，假设文件 fbar.bin 包含 Trimble RT17 for GPS 周 866，1996 年 8 月 11 日 - 1996 年 8 月 17 日，来自 Trimble SSE 接收器。然后，执行

```
teqc -tr s -week 866 + nav fbar2240.96n fbar.bin> fbar2240.96o
```

让我们剖析命令行。首先，-tr 选项标志告诉 teqc 目标文件来自 Trimble 接收器。-tr 的参数是 s（相当于这样），它告诉 teqc 本机格式是 RS-232 RT17 数据流，并且您希望将已转换的 RINEX [OBS](#) 发送到 stdout。但一般来说，RT17 文件 fbar.bin 允许包含 GPS 观测和星历表的两种记录类型。命令行选项+ nav fbar224.96n 告诉 teqc 将 fbar.bin 中的任何星历信息转储到 RINEX [NAV](#)file fbar2240.96n ；如果 fbar.bin 中没有星历记录，则执行完成后 fbar2240.96n 将为空。

如果你有一个 Trimble *.dat 文件，fbar.dat，那么类似的命令行应该是：

```
teqc -tr d -week 866 + nav fbar2240.96n fbar.dat> fbar2240.96o
```

现在，选项-week 866（或使用-week 96: 224 的替代格式）怎么样？通过这样做，你明确告诉 teqc 观测数据在 GPS 周 866 开始 ；它可能会运行到 GPS 周 867（或更晚），但 teqc 将跟踪此情况。如果你执行了较短的命令

```
teqc -tr s + nav fbar2240.96n fbar.bin> fbar2240.96o
```

在 1996 年 8 月 11 日 - 1996 年 8 月 17 日的那一周，你的 CPU 系统时间已经设置得到纠正，然后 teqc 会计算 GPS 周并使用它（在向 stderr 发出警告后：召回执行只是

TEQC

是一种方法，你可以找出 GPS 周 teqc 认为是什么）。

为什么必须指定 GPS 周？事实证明，Trimble RS-232 GPS 观测记录（以及其他一些原生格式）中没有信息表明它来自哪个 GPS 周；这是辅助信息，必须记录在观察记录的内容之外。（这里是在天宝星历记录 GPS 信息一

周，但难保会有 任何星历记录在任意 RT17 数据段中，更不用说在所有观察记录之前的星历记录。类似的论点适用于 Trimble *.dat 文件：Trimble 的 DAT 观察记录中没有 GPS 周信息 - 尽管 GPS 周出现在通常位于 *.dat 文件中的其他记录中。此外，当使用带有 DAT 文件的 teqc 作为目标文件时 - 而不是 stdin-- teqc 将尝试查找[名称匹配的 MES 文件](#) 以帮助解决 GPS 周问题。但是，同样不能保证匹配的 MES 文件存在。)

顺便提一下，使用-week 选项时，GPS 周可以通过几种格式提供：

- 周周 (周= GPS 周，例如 - 周 866) -
- 周[YY] YY: DOY (YY =年，DOY =一年中的某一天，例如-week 96: 228 或-week 1996: 228)
- week [YY] YY: MM: DD (YY =年，MM =月，DD =日，例如-week 96: 8: 15 或-week 1996: 8: 15)

您还可以使用/ (斜杠) 作为分隔符而不是:(冒号)。请记住：您在数据开始时指定 GPS 周。

全部或部分 RINEX 标题字段 PGM / RUN BY / DATE 在转换期间由 teqc 自动填充。程序字段填写可执行文件的名称 (本例中为 teqc) 及其当前版本号。

日期由系统时间查询填写，我们假设系统时间设置正确。在 UNIX 系统上，此日期为 UTC，然后将其写入 RINEX 文件。在 Microsoft 系统上，此日期可能是也可能不是 UTC。对于 Microsoft Windows 95/98 / NT 系统，应根据特定时区或本地时间与 UTC 之间的已知偏移量设置日期。对于这些情况，获得的日期应正确为 UTC。

对于 Microsoft DOS 或 Windows (或 Windows 95/98 / NT / 2000 / XP，如果 teqc 无法确定操作系统)，teqc 将查询环境变量 \$ UTC_MIN_OFFSET，如果设置，应包含应添加到系统时间以产生 UTC 的分钟数值。夏令时和标准时间之间的切换必须手动完成。如果未设置此环境变量，将查询系统时间并将其作为“Lc1”=本地时间放入日期字段。

现在检查命令行：

```
teqc -tr sn -week 866 + obs fbar2240.96o fbar.bin> fbar2240.96n
```

这里-tr 选项标志的参数是 sn，即你的主要兴趣是 fbar.bin 中的星历信息，它作为 RINEX [NAV](#) 文件转储到 stdout (这里重定向到文件 fbar2240.96n)。在 +观察 fbar2240.96o 选项指示 TEQC，如果任何观测记录在目标中遇到 fbar.bin，他们需要被解码，并写成 RINEX [OBS](#) 文件 fbar2240.96o。同样，需要选项-week 866 来确定观测数据的时期，而不是星历数据。

同样，Trimble *.dat 文件的类似命令行将是：

```
teqc -tr dn -week 866 + obs fbar2240.96o fbar.dat> fbar2240.96n
```

如果为 RS-232 文件 fbar.bin 执行上述命令，并且没有设置环境变量 \$ teqc_OPT 或 \$ teqc_CONFIG (或者如果设置了它们，但它们不包含任何-O.*或-N.*标题修改选项)，然后你会发现 fbar2240.96o 和 fbar2240.96n 中的大多数 RINEX 标题字段都是空白的。为什么？与 RS-232 观测记录中的 GPS 周一样，Trimble RS-232 数据记录中没有字段可以保存占用这些 RINEX 字段的信息类型。关于 teqc 自动填充的唯一字段是 初始 RINEX VERSION / TYPE 的字段记录 (暗示)，默认的 WAVELENGTH FACT L1 / 2 记录 (在这种情况下由接收器类型暗示) 和 #/ TYPES OF OBSERV 记录。但是，您可以通过使用命令行 \$ teqc_OPT, \$ teqc_CONFIG 或其他配置文件指定自己的-O.*和/或-N.*选项来覆盖这些空值。这是在编辑和转换模式下同时使用 teqc。上述转换程序也可以加窗。但是，目前还没有为任何二进制格式编写快速搜索算法，因此必须使用显式窗口 ([窗口选项](#))

(6)，(7) 或 (8)) 或指定从开始的窗口增量时间 ([窗口选项](#) (2))。

转换程序也可以是 `qc -ed`。这里假设您有一个名为 `fbar.dat` 的 Trimble *.dat 文件。对于正常类型的 qc 操作，请尝试以下方法：

```
teqc + qc -week 866 [ -st 960811000000 ] + dh 24 -tr d \
+ obs fbar2240.96o + nav fbar2240.96n fbar.dat | 更多
```

现在，stdout 将包含你现在对 `qc -mode` 执行的期望，但是 RINEX [OBS](#) 文件仍然使用选项 `flag + obs` 输出到文件 `fbar2240.96o`。该 `-st` 选项是可选的，由方括号表示。您可以使用显式窗口（在此示例中，使用 `-st` 选项和 `+ d *` 选项进行[窗口选项](#)（7））。如果转换为 RINEX [OBS](#)，`teqc` 的自动识别功能

可以消除指定输入格式的需要。除了 Ashtech U 文件（总是需要 `-ash u`）之外，所有上述格式都开发了自动识别功能。要确保 `teqc` 能够识别特定文件，请使用 `+ mdf` 选项。从而：

```
teqc + mdf fbar.dat
```

应该回来

可能的 `fbar.dat` 格式：Trimble 下载

```
teqc: ...退出
```

任何转换的假定标准输出始终是 RINEX [OBS](#)。因此具有自动识别功能：

```
teqc -tr d trimble.dat> RINEX_OBS
```

可以简化为：

```
teqc trimble.dat> RINEX_OBS
```

自动识别大部分时间都可以使用，但不能保证！这是因为自动识别基于在文件开头只读取少量字节（通常只有 1-4 个字节）。如果您在命令行上手动测试文件，这可能是最有用的。要在脚本中使用 `teqc`，请使用显式接收器/格式选项。

要查看，使用二进制数据时需要记住以下几点：

1. 您可能需要指定主要感兴趣的记录类型，例如，使用 Trimble 数据的 `-tr` 选项，下载 (*.dat) 格式的 `d` 参数或 RT17 流格式的 `s`。如果不执行 qc 模式，则将与此记录类型对应的 RINEX 文件类型转储到 stdout，例如，如果使用 `-tr do`，则将 RINEX [OBS](#) 文件信息转储到 stdout。（对于大多数情况，在执行 qc 模式时，qc 信息被转储到 stdout。）
2. 您应指定二进制流启动的 GPS 周，或者您接受计算 GPS 周的当地时间的计算机系统版本。对于大多数格式，您可能首先尝试不使用 `-week` 选项，但偶尔包含初始 GPS 周的记录已损坏，伪造或丢失，并且（根据具体情况）`teqc` 可能会尝试使用 GPS 的系统时间周。此外，一些 Trimble MES 文件被发现包含奇怪的年份，如“19116”！
3. 如果执行 qc 模式，则必须使用 `+ d *` 标志或其中一个显式[窗口选项](#)（6） - （8）提供有关感兴趣的时间窗口长度的一些信息。如果做前者，则第一次数据观察的时间成为窗口的开始时间。`-st` 和/或 `-e` 选项的部分参数格式也有效。
4. 如果执行 qc 模式，则使用 stdout 转储[短报告段](#)的副本。为了捕获如果不执行 qc 模式将转到 stdout 的 RINEX 文件类型，请使用 `+ obs`，`+ nav` 或 `+ met` 选项指定 RINEX 文件名。

此外，可以在转换期间使用[时间分级](#)选项。有关转换问题等，请联系 [teqc 技术联系人](#) 寻求帮助。

16 特殊转换注意事项和选项

有一些转换器选项并非特定于特定的本机二进制格式。有

-L2 表示仅 L1（无 L2 跟踪）接收器

-P 以指示 P 无码接收器

在需要时使用这些选项有助于设置 RINEX [OBS](#) 可观察的默认设置。对于仅 L1 和 P-codeless 的接收器，请同时使用 -L2 和 -P。

可以在任何时候使用的两个有用选项，但有时在转换之前非常有用，是元数据提取选项 + meta 和 + mds - 它们也可以与 RINEX 一起用作目标文件。例如，

```
teqc + meta trimble.dat
```

应该返回关于 Trimble DAT 文件 trimble.dat 的 19 行元数据摘要（假设自动识别功能正常工作）。执行

```
teqc + mds trimble.dat
```

返回“元数据简短” - 只是文件的开始和结束时间的简写，加上文件大小（以字节为单位）。如果其中任何一个终止于如下行：

```
周: #####
```

这表明您应该使用 -week 选项将起始 GPS 周设置为指示值，例如

```
teqc -week ##### + meta trimble.dat
```

有几种转换器选项特定于特定的本机二进制格式。

- Trimble *.dat 或 RT17 数据格式

有一组选项可以从转换的 RINEX [OBS](#) 文件中删除半波长相位数据（平方模式）。这些是 -L1_2 或 -L2_2，分别去除平方 L1 或平方 L2。在 teqc 当前处理的二进制数据类型中，这些标志可能使用的唯一类型是 Trimble *.dat 或 RT17 数据流格式。

此外，从 P-codeless 接收器（例如 SD, STD, SST）转换 *.dat 时，您可能必须使用 -P 选项。这通知 teqc 数据没有 P 代码，并且它对某些标志执行位清理。如果没有这种位清理，您很可能只能获得 L2 可观察性。

当使用最新一代的接收器（例如 SSE, SSi, 4700, 5700）时，可以报告针对特定 SV 的几个平方 L2 数据的时期。通常，这些时期通过使用（平方）L2 数据添加到 RINEX [OBS](#) 文件的 LLI 标志的适当位 1 进行转换（有关更多信息，请参阅 teqc 对 [波长因子](#) 的处理）。通过使用选项 -L2_2 可以在转换期间完全删除这些 L2 可观察量，即没有将平方的 L2 数据传递给 RINEX [OBS](#) 文件。

当使用 Trimble DAT 文件作为目标文件（而不是 stdin）时，teqc 尝试查找具有相同路径和名称前缀的 Trimble MES 文件。名称匹配使用：

```
* dat - >寻找* mes
```

```
* DAT - >寻找* MES
```

如果找到匹配的 MES 文件，则读取它以获得起始 GPS 周和某些元数据（尽管不能保证该信息是正确的）。

- ConanBinary:

Teqc 不应该用于从早期的 Rogue 接收器转换 ConanBinary。此类 ConanBinary 中的数据是按顺序排序的，而不是按时间排序的，teqc 只会转换第一个 SV PRN 数据。（对于这种类型的 ConanBinary，请使用 JPL 转换器。）对于来自 TurboRogue / TurboStar 和 Benchmark 接收器的 ConanBinary，teqc 将正常工作。

- TurboBinary

TurboBinary 数据可包括正常速率数据（记录 0x68），1 秒速率数据（记录 0x1a），高达 50 Hz 的高速率数据（记录 0xdb 和 0xdc，加上使用记录 0x1a 中的信息），以及称为“30-1 秒”格式，它是正常速率数据（记录 0x68）和 1 秒 LC 数据（记录 0xde）的混合。默认转换是执行所有这些记录类型。但是，您可以使用以下选项定制转换：

-TBhr

遗漏高比率数据

-TB1s

遗漏 1 秒数据（这也删除了高速数据）

-TBnr

省略正常率数据

-TBLC

省略 LC 数据记录

各种选项组合的结果是：

（无选项=默认值）转换所有记录

-TB1s

仅转换正常费率数据

-TB1s -TBhr

（与-TB1 相同）仅转换正常速率数据

-TBnr

转换 1 秒和高速数据或 LC 数据（无论剩下什么）

-TBhr

转换 1 秒和正常速率的数据

-TBhr -TBnr

仅转换 1 秒数据

-TBLC

省略 LC 数据记录，保留正常率

-TB1s -TBnr

转换没有观察数据（哎呀！）

-TBLC -TBnr

转换没有观察数据（哎呀！）

对于使用固件日期约为 12 月 1 日 92（版本 2.5 或更早版本）收集的 TurboBinary 数据，有必要应用更正以获得有效的伪距。要激活此更正，请使用选项+ TB_ca_fix。

对于使用 Benchmark ACT 接收器收集的 TurboBinary 数据，您可能需要尝试使用-aoa tbY 选项来生成 RINEX [OBS](#) 文件。这将使用 C / A 导出的 RINEX L1 的 L1 相位值，而不是噪声较大的 Y1 无代码导出的 L1 相位值。

- Ashtech 数据格式：

对于除 U 文件“数据模式 7”之外的所有 Ashtech 格式，默认情况下不应用伪距的内部“平滑”校正；指定选项+平滑以启用此选项。（伪距平滑似乎默认在 Berne ASRINEX0 转换器和 Ashtech ASHTORIN 转换器中完

成。) 对于 U 文件数据模式 7，伪距在存储或不存在平滑校正的情况下存储； 在转换过程中无法改变这一点。

另外，可以从 `teqc` 观察到 RINEX L1 可能会注意到其他转换人员报告的 L1 可观察量略有变化（最多 1 厘米左右）。这是由于使用在 P1 代码数据块中报告的 L1 相位值而不是在 C / A 代码数据块中报告的 L1 相位值。麻省理工学院的初步测试表明，P1 代码数据块中的 L1 相位值导致 RMS 略低。目前，`teqc` 将继续报告此 L1 相位值，但您可以使用 `+ CA_L1` 选项切换到 L1 (C / A)。

对于 Ashtech 下载文件集，`teqc` 可能仅适用于“版本：3”类型下载。例如，较旧的“版本：1”和“版本：2”类型下载目前无法正确转换， 应使用 `ASRINEX0` 或 `Ashtech ASH2RIN` 转换器，或尝试使用 `Ashtech convert.exe` 程序将 B 文件更改为版本 3 B 文件。

Z-XII 的早期固件中的一个错误导致毫秒时钟重置太晚了。如果您注意到周期性的毫秒滑动（发生在报告的时钟复位时间之前），请在转换期间尝试使用 `+ Ashtech_old_clk_reset` 选项 。这应该从数据中删除此接收器固件工件。

17 波长因素：teqc 与他们的关系

波长因子，即指定 L1 或 L2 是以全波长还是半波长（平方模式）进行记录，可以在 RINEX 中以各种方式完成。简而言之，1) 有一个必需的 RINEX 标题记录 `WAVELENGTH FACT L1 / 2` 指定所有 SV 的 L1 和 L2 的默认波长因子，2) 可以有其他波长情况 `L1 / 2` 记录指定一组不同的波长因子对于特定的 SV，以及 3) 可以在 L1 或 L2 观测的 LLI 标志中使用比特 1 来指示与特定 SV 的最后一个波长因子 `L1 / 2` 记录相反的波长因子状态。任何 `WAVELENGTH FACT L1 / 2` 的可能值集记录是“1 0”，“1 1”，“1 2”，“2 0”，“2 1”和“2 2”。如果该频率的最后一个 `WAVELENGTH FACT L1 / 2` 记录和文件中较早的某个位置设置为“2”（半波长），则设置 LLI 标志的第 1 位表示全波长模式。同样，如果该频率的最后一个 `WAVELENGTH FACT L1 / 2` 记录和文件中较早的某个 SV 设置为“1”（全波长），则设置 LLI 标志的第 1 位表示半波长模式。)。 `teqc`

使用的方法是 RINEX 所有可能性的简单特定子集，但仍保留所有相同的信息。 将出现标题记录，并且仅使用“1 1”或“1 0”的 L1 / L2 状态。换句话说，RINEX 文件头中报告的默认设置始终是 L1 和 L2（如果存在）的全波长，即使对于平方接收器也是如此。通过在 L1 或 L2 观测值上设置 LLI 标志的适当位-1 来指示特定的半波长观察。期。

在平移过程中，您可以选择排除所有半波长观测。为此，请包括 `-L1_2` 或 `-L2_2` 以分别排除平方 L1 或 L2。这将在本机二进制格式转换为 RINEX 或在任何 RINEX 到 RINEX 操作期间都有效。波长因子 的 `teqc` 默认设置为 `+ L1_2` 和 `+ L2_2`，即包括所有半波长观测值。

18 基本命令：审查

以下示例假定 shell 环境变量 `$ teqc_OPT` 和 `$ teqc_CONFIG` 未设置或为空：

[`TEQC`](#)

强制所有初始化并根据系统时间报告当前的 GPS 周

`teqc + id`

识别您拥有的 `teqc` 版本，以及发送到 `stderr` 的其他信息，如您的计算机系统时间

[`teqc -help`](#) 或 [`teqc + help`](#)

完整的选项列表是 `stderr`

[`teqc + err my_help_file + help`](#)

完整的选项列表被写入文件 `my_help_file` 而不是 `stderr`； `+ err` 选项将所有 `stderr` 重定向到指定的文件

[`teqc ++ config`](#)

将当前配置转储到 `stdout`

[`teqc + qc ++ config`](#)

将当前配置和默认 `qc` 设置/值转储到 `stdout`

[`teqc + v RINEX_file`](#)

读取 RINEX 文件 `RINEX_file` 并验证其格式； 虽然向 `stderr` 发送了验证消息，但没有任何内容发送到 `stdout`

[`teqc ++ config RINEX_OBS_file`](#)

将当前配置和 [OBS](#) 头设置/值 `RINEX_OBS_file` 转储到 `stdout`（也可以使用 RINEX [NAV](#) 或 [MET](#) 文件）

[`teqc RINEX_file`](#)

读取和喷出 `RINEX_file`（可能有一些轻微的格式改进）返回到 `stdout`

[`teqc + dh 6 RINEX_OBS_file`](#)

读取并喷出标题，并将前 6 个小时的 `RINEX_OBS_file` 观察结果 返回到标准输出

[`teqc RINEX_file_1 RINEX_file_2`](#)

读取并拼接两个 RINEX 文件 `RINEX_file_1` 和 `RINEX_file_2` 作为单个 RINEX 文件返回到 `stdout`； 目标 RINEX 文件应该具有相同的类型和时间顺序

[`teqc -0.mo foobar RINEX_OBS_file`](#)

读取 `RINEX_OBS_file` 并将纪念碑名称改为“foobar”； 编辑过的 RINEX [OBS](#) 文件是 `stdout`

[`teqc + qc RINEX_OBS_file`](#)

读取 `RINEX_OBS_file`； 自动搜索 [匹配名称的](#) RINEX [NAV](#) 文件； `qc` [短报告段](#) 是 `stdout`； 完整的 [qc 报告](#) 和 [qc 绘图文件](#) 写入文件系统

[`teqc ++ sym`](#)

[符号](#)代码的[符号层次结构](#)和 `qc` ASCII 时间图的相关含义被写入 `stdout`

[`teqc -tr do + nav RINEX_NAV_file trimble.dat`](#)

转换 Trimble dat 文件 `trimble.dat`； RINEX [OBS](#) 文件向 `stdout` 注入； RINEX [NAV](#) 文件写入 `RINEX_NAV_file`

[`teqc -warn { rest of command }`](#)

关闭警告去 `stderr`； 其他功能仍然存在

19 在脚本中使用 `teqc`：批处理模式的替换

由于 `teqc` 是 100% 非交互式的，因此非常适合在脚本中使用并在后台运行。事实上，这正是它被设计为 100% 非交互式的原因。让我们看一个简单的脚本，将 Trimble *.dat 文件转换为 RINEX，然后 `qc` 生成的 RINEX 文件：

```
#!/bin/ksh
```

```
for file in $(ls *.dat)
```

```
do
```

```
do
```

```

teqc -O.int 30 -tr d + nav $ {file%dat} 97n $ file> $ {file%dat}
97o

teqc + qc -set_mask 15 -plot $ {file%dat} 97o> $ {file%dat} qc
DONE
#end of script

```

确保脚本是可执行的，即执行（在 UNIX 中）`chmod 755 脚本`，其中脚本，例如，是脚本文件的名称。使用它

脚本* .dat

让我们仔细看看执行这个脚本时发生了什么。shell 扩展了命令行上的* .dat，以包含工作目录中以 .dat 结尾的所有文件。这些文件名中的每一个都由脚本处理。脚本的第一行强制脚本在 Korn shell (ksh) 中运行。脚本的第一个实际部分是仅回显每个 .dat 文件的名称。接下来，每个 .dat 文件都被转换，并且标题中插入了 30 秒的采样间隔。生成的 RINEX [NAV](#) 和 [OBS](#) 文件将具有与 .dat 文件相同的前缀，但将以 97n 和 97o 结尾分别，而不是数据。接下来，将 RINEX 文件设置为 `qc -ed`，将高程掩码设置为 15°（覆盖默认值 10°）。由于我们不要求 COMPACT 绘图文件（选项 `-plot`），因此只创建了两个 qc 文件，这两个 qc 文件都具有与 .dat 文件相同的前缀。更明显的一个将以 qc 结束，从 stdout 重定向。这是 [简短的报告部分](#)（大约一页）。创建的另一个文件将以 97S 结束，并且是完整报告，包括 - 在这种情况下 - [短报告段](#)（就像转到 stdout 的那样）和更详细的[长报告段](#)。

如果要抑制[报告](#)中的[短报告段](#)（* .97S 文件），请在 qc 命令行中包含 `-s`。如果您不想要任何 [qc 报告文件](#)，请在命令行中包含 `-report`。没有办法通过命令行选项来抑制 [短文件段](#) 转到 stdout，但是你总是可以在 UNIX 上使用 `> / dev / null` 来消除 stdout。

显然，这个脚本（如编写的）根据仅在 1997 年收集的* .dat 文件的 Berne 命名转换创建了正确命名的 RINEX 文件，尽管该脚本适用于大多数* .dat 文件。

请注意，尚未指定 GPS 周。对于大多数* .dat 文件（至少来自具有最新固件的 Trimble 接收器），第一个记录之一通常包含数据开始的 GPS 周。如果此记录丢失或已损坏，则生成的 RINEX [OBS](#) 文件可能会有错误的观察日期，从而导致 qc 报告不佳。对于这些罕见的* .dat 情况，您必须使用 `-week` 选项明确说明 GPS 周。

20 teqc 的 qc 模式与原始的 UNAVCO QC

本节仅适用于熟悉原始 UNAVCO 质量控制程序（不再受支持）且正在向新 `teqc + qc` 过渡的人员。

界面差异：

- 使用 `teqc` 命令行选项或环境变量 `$ teqc_CONFIG` 中的等效格式 替换文件 `qc.inp`，或者使用 `-config filename` 选项访问一个或多个配置文件； 所有选项都有合理的默认值，因此用户最初不需要关注细节
- 删除批处理模式的文件 `qc.fil` ； 请改用命令行脚本
- 消除 `qc.tim` 或 `qc.sym` 等辅助文件
- 大致，
 - 原始 QC 标准输出就像 `teqc + qc *.YY S` [qc 报告文件](#)
 - 原始 QC `*.YY S` 文件类似于 `teqc + qc` 标准输出（[qc 简短报告段](#)）

内部差异（或新 QC 结果为何与旧 QC 结果略有不同）：

- （名义上）`teqc` 中的一个文件传递[原始 QC 通常需要每个 RINEX [OBS](#) 文件的两个完整传递]

- 处理 NAVSTAR GPS, GLONASS, Galileo, 北斗/指南针, QZSS, IRNSS 和 SBAS [仅适用于 NAVSTAR GPS 的原始 QC]
- ASCII 时间图 的明确定义的[符号层次结构](#) [原始 QC 只有部分定义的符号层次结构]
- 立方样条 xyz 拟合 SV 轨道的高程, 方位角估计[原始 QC 使用直接线性拟合高程, 方位角只对固定天线有利并且不光滑; 算法偶尔导致方位角误差大]
- 改进的多径算法
- 改进了观测数据缺口的检测和报告[原始 QC 通常没有检测到并报告了各个 SV 的长数据缺口, 并且无法报告所有 SV 的短数据缺口]
- 正确识别高程掩模下面的 SV 数据[原始 QC 经常错误地报告高度掩模下面的时期 w /数据, 实际上, 当 SV 远高于高程掩模时接收器停止跟踪]
- 电离层延迟滑动的正确计数和“每次滑动的观察”[原始 QC 将计算第一次观察 w / L1 和 L2, 因为 SV 在观察后开始被再次跟踪, 然后设置为电离层延迟滑动]
- 卫星仰角和方位角是 WGS 84 椭圆柱地球模型的原因[原始 QC 假设球形地球模型, 导致中纬度地区的高程误差高达 1/5°]
- 对每个 SV 的预期观测数量的更准确计数[对于某些数据集, 原始 QC 有时报告的观测数量超出预期, 导致“%”值超过 100%]
- ASCII 时间图上显示的时钟复位符号[原始 QC 未显示时钟复位的符号]
- 在 ASCII 时间图中正确报告指标[原始 QC 通常没有显示某些时钟复位, 有时错误地显示时钟复位发生时没有, 以及指标的其他小问题]
- 不会产生任何不需要的辅助文件 (如 AUXFIL 或 temp.orb)

21 解释 teqc 的 qc 模式输出

teqc 的质量检查输出分为两部分, [简短报告部分](#)和 [长报告部分](#)。在 [简短的报告部分](#)包括 ASCII 时间图和各种参数的总结报告。该 [长的报告部分](#)给出了一些参数, 或者通过 SV 或抬高的更详细的分类 (如 QC 满)。

简短报告部分:

在简短报告段中, qc 输出中最紧凑的信息之一是 ASCII 时间图。在该图中, 针对每个卫星显示各种类型的质量指标的视觉概要作为时间的函数。SV PRN 编号显示在图的左侧和右侧。ASCII 时间图的宽度由-w [宽度]选项控制, 通常设置为 72, 但可以在 1 到 255 之间变化。宽度为 72 和 24 小时的观察数据, 每个 ASCII 字符“bin”代表正好 20 分钟的时间。根据明确定义的[符号层次结构](#), ASCII 时间图中每个点中显示的每个字符是由该二进制位表示的所有观察时期发生的最重要的注释项。

可以通过执行以下命令将整个 ASCII 绘图符号表的列表转储到 stdout:

```
teqc ++ sym
```

同样, 通过在任何 qc -mode 运行中包含+ sym 选项, 可以在短报告段中包含符号层次结构表的摘要。但是在本教程中, 让我们仔细看看符号及其层次结构。每个“SV”行上使用的符号如下所示, 第一个符号在符号层次结构中具有最高优先级, 通过列表递减:

C

发生了一次时钟滑动; 时钟滑动是对所有被观察 (跟踪) 的卫星发生的 MP1 和 MP2 滑动, 其值与毫秒到-msec_tol 指定的分辨率的毫秒级相同 (以毫秒为单位); 使用-cl 选项 关闭检测

米

类似于时钟滑动，但只有一些（即并非所有）被观察（跟踪）的卫星具有 MP1 或 MP2 滑动，这是一个整数毫秒，或者不同卫星的整数是不同的； 注意：如果毫秒滑移容差为 1e-2（参见-msec_tol），则随机 MP1 或 MP2 多径滑动将被标记为 m，而不是 M，1 或 2 的概率大约为 2: 100 （见本表中的其他地方）

一世

发生电离层延迟（相位）滑移； 使用-ion 选项 关闭检测

中号

MP1 和 MP2（代码）滑动都发生了，但不是整数毫秒； 使用-mp 选项 关闭检测

1

只发生 MP1（代码）滑动，但不是整数毫秒； 使用-mp 选项 关闭检测

2

仅发生 MP2（代码）滑动，但不是整数毫秒； 使用-mp 选项 关闭检测

-

对于 qc 满，卫星高于高程掩模，但接收器没有明显记录数据； 对于 qc -lite（没有星历信息），数据间隙也必须小于指定的最大值（参见-gap_mx 的参数，以分钟为单位）

+

（仅限 qc）卫星低于高程掩模，并收集了一整套相位和代码数据

^

（qc 仅限）卫星低于高程掩模，并收集了一组部分相位和代码数据

。

SV 的相位和/或代码数据为 L1，仅 C / A 和 A / S 关闭； 如果 qc 满，卫星高于高程掩模

:

SV 的相位和/或代码数据仅为 L1 和 P1，A / S 关闭； 如果 qc 满，卫星高于高程掩模

~

SV 的相位和/或代码数据为 L1，C / A，L2，P2 和 A / S 关闭； 如果 qc 满，卫星高于高程掩模

*

SV 的相位和/或代码数据为 L1，P1，L2，P2 和 A / S 关闭； 如果 qc 满，卫星高于高程掩模

,

SV 的相位和/或代码数据为 L1，仅 C / A 和 A / S 开启； 如果 qc 满，卫星高于高程掩模

;

SV 的相位和/或代码数据为 L1，仅 P1 和 A / S 为开； 如果 qc 满，卫星高于高程掩模

Ø

SV 的相位和/或代码数据为 L1，C / A，L2，P2 和 A / S 接通； 如果 qc 满，卫星高于高程掩模

ÿ

SV 的相位和/或代码数据为 L1，P1，L2，P2 和 A / S 打开； 如果 qc 满，卫星高于高程掩模

大号

锁定丢失指示器由接收器设置为 L1 和/或 L2； 使用-lli 选项 关闭检测

-

（下划线）（仅限 qc）卫星在地平线和高程掩模之间，没有接收方收集的数据； 使用-hor 选项 关闭指示器

，，

（空白）qc lite：没有跟踪卫星； qc full：没有卫星计算在地平线以上（+ hor 选项）或以上掩模（+ hor 选项或两者都是-hor 和+掩模选项）（另请参阅-set_hor 和 -set_mask 选项）

我们来看一个简单的例子。假设地平线设置为 0°，高程掩模设置为 20°。我们还假设接收器在达到 25° 仰角时开始跟踪特定卫星并继续跟踪卫星到达 5° 仰角。我们还假设在跟踪期间没有发生滑动并且设置了+ hor 选项。对于 qc full（SV 星历可用），SV 符号轨道可能看起来像下列之一：

A / S on: _____ -- oooooooooooooooooooooo ++++++ ____
A / S 关闭: _____ -- ***** ++++++ ____
1 2 3 4 5 6

在时间（1），SV 上升到地平线（0°）以上。在时间（2），SV 上升到高程掩模（20°）以上，但是接收器在时间（3）上升到 25° 之前不开始跟踪。在时间（3）和（4）之间，所有相位和代码可观察量由接收器收集（L1，L2，C / A 和 P2 用于 A / S 开启；L1，L2，P1 和 P2 用于 A / S 关闭）。随着 SV 在时间（4）下降到高度掩模以下直到接收器在时间（5）处以 5° 的高度停止跟踪，继续收集数据。SV 最终在时间（6）处设定在地平线以下。对于 qc lite，上面的 SV 符号轨道将显示为：

A / S on: oooooooooooooooooooooooooo
A / S 关闭: *****
1 2 3 4 5 6


作为 teqc，没有关于卫星高程的信息。





如果设置了-hor 选项，则上面的 qc 完整 SV 符号轨迹将显示为：

A / S on: --ooooooooooooooooooooo ++++++
A / S 关闭: - ***** ++++++
1 2 3 4 5 6

因此，您可以使用+ hor 设置确定所有可能的内容，除了分别在时间（1）和（2）的 SV 的上升和设置时间。对于 qc lite，SV 符号轨道将保持与之前相同，因为选项-hor 和+ hor 是无意义的。

在 SV 符号轨道中出现的不在上述示例中的任何其他符号都属于“不太好”的类别，尽管看到很多- qc 完整输出中的符号也“不好”。让我们更详细地了解这些其他指标可能是什么。

第一个“不太好”的类别通常可被视为“缺失数据”。正如刚才提到的，，这意味着使用提供的星历计算 SV 高于高程掩模，但没有观察数据存在于此 SV。换句话说，所有数据都丢失了。

以下—“所有数据缺失”指标是部分缺失数据指标。例如，如果 A / S 通常打开，则会看到： 或者， 表示该 bin 中至少有一个观察时期，其中 L2 可观察量（即 L2 和 P2）缺失。您不太可能看到 y，因为这将需要 Y 代码接收器（能够在 A / S 打开时跟踪 P1）。（例外情况是 qc 来自 Ashtech 接收器的数据，如 Z-XII。无论 A / S 是打开还是关闭，都会报告所有 SV 的 C / A 和 P1 伪距观察值。）如果 A / S 通常关闭，则看： 或。 表示在该 bin 中至少有一个观察时期，其中 L2 可观察者缺失。A ~ 表示由于某种原因，即使 A / S 关闭，接收器也无法跟踪 P1，因此接收器记录了 C / A 可观察量。

如果您使用的 P 代码（非 Y 代码）接收器在每个时期报告每个 SV 的 C / A 和 P1 伪距（如 Ashtech Z-XII），您可能需要使用 -Y 选项，通知 teqc 这不是来自 Y 代码接收器的数据，而是将数据分析视为来自 P 代码接收器。

无编码接收机数据的特殊处理

您可能拥有使用“无代码”或“平方”接收器收集的数据集。对于这些接收器，伪距 P1 和 P2 永远不会被记录，并且默认的 qc 报告将显示所有观察结果都是不完整的，因为它们都不能进行 P2 观察（尽管 C / A 可以代替丢失的 P1）。此外，上面示例的 SV 符号轨迹将显示为：

```
A / S on:  _____ -- //////////////// ++++++

A / S 关闭:  _____ - ..... ++++++

1 2 3 4 5 6
```

其中（正确地）表示缺少 L2 和 P2，即使可观察的 P2 不可能（因此从不存在），并且 L2 可能总是存在。

您可以通过包含 -P 选项（当然还有 + P 中的默认选项）通知 teqc 数据将被解释为无代码接收器收集的数据。在这种情况下，统计数据忽略了 P 代码的缺失，数据指标变化只有两种可能性（从最初的八种可能性）：

。

SV 的相位和/或代码数据是 L1, C / A; 如果 qc 满，卫星高于高程掩模

Ø

SV 的相位和/或代码数据是 L1, C / A 和 L2; 如果 qc 满，卫星高于高程掩模

请注意，A / S 状态（开或关）被忽略，因为这与处理平方数据无关。然后，当使用 -P 选项时，上面的 SV 符号轨迹将显示为：

```
A / S on:  _____ -- ooooooooooooooooooooo ++++++ ____

A / S 关闭:  _____ -- ooooooooooooooooooooo ++++++ ____

1 2 3 4 5 6
```

即 A / S 是打开还是关闭没有区别。你可能偶尔会看到。数据指标：

```
A / S 任何:  _____ -- ooo.oooooooo.ooooooo ++++++ ____
```


1 2 3 4 5 6

指示可观察的 L2 缺失的一个或多个观察时期。

可以通过执行以下命令将为无代码接收器修改的整个 ASCII 绘图符号表的列表转储到 stdout:

```
teqc -P ++ sym
```

其他指标

下一组“不太好”的指标是针对可观测量的单据。常见类型的滑移是由 I 符号表示的电离层延迟（相位）滑移。这些经常发生在 SV 处于低海拔时，无论是上升还是设置，因此上面的示例 SV 符号轨迹可能真的显示为：

```
A / S on: _____ -- Ioooooooooooooooooooo ++ I + I__
```

```
A / S 关闭: _____ --我***** ++ I + I__
```

1 2 3 4 5 6

在这个例子中，在接收器开始跟踪 SV 并且 SV 接近设置之后不久检测到电离层延迟滑动。

另一种常见类型的滑动是针对多径（代码）可观察量 MP1 和 MP2 中的一个或两个。同样，这些经常发生在低海拔地区。有三个符号：M 表示 MP1 和 MP2 上的滑动，1 表示仅在 MP1 上滑动，2 表示仅在 MP2 上滑动。请注意，多路径滑动指示符在[符号层次结构中的](#)优先级较低，因此如果在 ASCII bin 中的不同观察时期出现电离层延迟滑动和多径滑动，则只会看到 I 符号（假设选项配置为+离子）和+ mp）。如果使用-ion 来抑制电离层延迟滑动检测，上面的 SV 符号轨迹现在可能显示为：

```
A / S on: _____ -- Moooooooooooooooooooo ++ 2 + M__
```

```
A / S 关闭: _____ -- M ***** ++ 2 + M__
```

1 2 3 4 5 6

有趣的是，记录的观测中电离层延迟或多径滑动的发生不仅是天线环境的函数（意味着一直回到发射 SV），而且也是特定接收机的函数。由于内部滑移检测和相位可观察的复位，来自某些接收器的数据显示几乎没有电离层延迟滑动和几乎所有多径滑动。其他接收器不会重置相位可观测测量，并显示比多径滑动更多的电离层延迟滑动。

最后一种类型的滑动是“n 毫秒时钟滑动”，其中 n 的值通常为 1，由符号 C 表示在 SV 符号轨道中。如果正在跟踪的所有 SV 在 MP1 中滑动并且 MP2 等于相同的积分毫秒数，则报告此滑动到-msec_tol 选项指定的容差。如果将公差设置为 1e-17（毫秒），您可能永远不会看到任何这些单据。但是，默认容差是 1e-2（毫秒），并且使用此值，teqc 的 qc 模式似乎非常能够检测这些滑动（如果它们存在）。

如果在 SV 符号轨道中看到 C 符号，这意味着什么？有几种原因，有些原因比其他原因更无害。如果是 C. 符号之前是观察间隙（没有为任何 SV 收集数据），观察时期标签中可能缺少一个或多个毫秒时钟复位。此外，如果使用 teqc 将两个 RINEX [OBS](#) 文件拼接在一起并且在第一个文件中发生时钟重置，则在第二个文件的第一个纪

元处将出现 C（因为 teqc 拼接不会将第二个文件中的观察时间修改为在第一个文件中重置累计时钟的原因）。然而，在其他情况下，如果观察时期是相当连续的，并且 C 指示符在 24 小时数据中出现两次或更多次，则接收器很可能不健康。

后一种可能性（接收器不健康）促使包含另一个滑动指示器，即“n 毫秒多径滑动”。据观察，有些接收器变得如此不健康，即使应该发生 n 毫秒的时钟滑动（即，给定特定的接收器，不存在毫秒时钟复位，即使它们是预期的）也没有找到，因为多路径滑动对于不同的 SV 具有不同的毫秒当量。简而言之，对于所有被跟踪的 SV，n 的值不是常数。在这种情况下，使用 m 符号。有一些概率（大约为 2：毫秒公差倒数），随机多径滑动将被记录为 m 而不是 M 或 1 或 2；所以，像对待任何多径滑动一样对待偶尔的 m。但是，如果您开始看到许多 m 个符号，特别是如果您在同一接收器的数据中看到 C 符号被报告，则怀疑接收器正在生病。

下一个“不太好”的类别是存在数据缺口。实际上有两种类型的差距。一个是所有 SV 的完整观察间隙。这可能是由于接收器被关闭并随后重新打开，在接收器本身内部或由于与接收器的通信故障之间的一段时间内丢失了所有数据。目前，SV 符号轨道中没有指示完整的观察间隙，除非（有时）它们存在于 qc lite 运行中。

另一种类型的间隙是 SV 数据间隙，其中接收器停止跟踪 SV 一段时间，即使它远远高于地平线或高程掩模，可能是由于阻塞。SV 数据间隙的确切定义取决于 teqc 正在以 qc full 或 qc lite 模式运行。对于 qc full，如果在 SV 高于高程掩模时存在一个或多个缺失的观察时期，则发生 SV 数据间隙。对于 qc lite，如果跟踪停止超过指定的最小时间（-gap_mn，再次），然后跟踪在指定的最大时间之前恢复，则会出现 SV 数据间隙（请参阅 -gap_mx 选项）。SV 符号轨道中使用的符号现在是-，因此 SV 数据间隙可能如下所示：

A / S on: _____ -- I o o o o o o o o o o - o o o o o ++ I + I ____

A / S 关闭: _____ -- 我***** - ***** ++ I + I ____

1 2 3 ab 4 5 6

发生在间隔（ab）。其含义与间隔（23）实际相同，即 SV 高于高程掩模，但未收到任何数据。 [符号层次中](#)

较低的 SV 数据的唯一其他指标是“失锁”指示符 L，当接收器发出 L1 或 L2 可观察的锁定丢失时使用该指示符。大量 L 符号可能表示不健康的接收器或天线。在 ASCII 时间图中很少会出现这种情况。

SV 符号后面的轨道是一个或四个，取决于你是分别使用 qc lite 还是 qc full。对于 qc lite，有一个标有“Obs”的符号行。这将使用十六进制表示记录接收器为每个 bin 跟踪的最大 SV 数。例如，如果此行上有 7，则跟踪 7 个 SV，以查找该时间段所代表的至少一个观察时期；如果此行上有 b，则跟踪 11 个 SV，以查找该时间段所代表的至少一个观察时期；等等。如果没有跟踪 SV，则显示（空白）而不是 0。如果一个或多个 s 出现在 qc lite 运行的“Obs”行中，这是发生完整观察间隙的最佳指标。

对于 qc 完整运行，“Obs”行被四行替换，标记为“-dn”，“+ dn”，“+ XX”和“Pos”。“+ XX”线最像 qc lite “Obs”线；的 XX 是由度仰角掩模（四舍五入至最接近的程度）代替，并且指示是高度角以上 SV 的最大预期数目，根据提供的星历，再次使用十六进制表示法。qc lite “Obs”行和 qc full “+ XX”行之间的差异是现实与理论之间的差异：“显示了可以看到的内容（在高程掩模之上）。

现实与理论之间的差异记录在“-dn”和“+ dn”行中，这是 SV 跟踪差异计数，并记录了两个界限。给出当前

高程掩码的差异。这些行可以被认为是未跟踪的 SV 的数量显示“好的新/坏消息”。行“-dn”记录了该时间段的所有观察时期的最小差异。“+ dn”行记录了该时间段的所有观察时期的最大差异。差异计数也以十六进制表示法显示，’ ’（空白）为 0。截至 2009 年 3 月 9 日，还存在反向差异指标，’ + ’，意味着在高程掩码下面有一个或多个 SV 的数据。

要使差异线正常工作，必须正确设置选项-max_rx_SVs。这表示接收器能够跟踪的 SV 的最大数量，并且当前具有默认值 12。如果在 qc_full 的情况下发生完整的观察间隙，则将显示一组 c 个字符（c 为十六进制为 12），至少在“+ dn”线上，如果间隙足够大，也可以在“-dn”线上。

举个例子，我们假设“-dn”行上有一个’ 1 ’；这意味着该时间段中的每个时期至少缺少 1 个可以跟踪的 SV。A’ 2 “-dn”行上的’ 意味着该时间段中的每个时期至少缺少 2 个可以跟踪的 SV，依此类推。“-dn”行上的’ ’（空白，即 0）表示在该时间仓中至少有一个纪元，其中跟踪了所有可能的 SV（给定接收者的信道限制）。

在“+ dn”行上，1 ’表示该时间段中至少有一个时间段缺少 1 个可以跟踪的 SV；一个“ 2 ”是指有在缺失本来可以跟踪 2 层的 SV 那时仓至少一个历元；等等。“+ dn”行上的’ ’（空白，即 0）表示该时间段中的每个时期都有可能的 SV（给定接收者’）

当填充非零条目（即不是’ ’或空白）时，顶部带有“-dn”的“-dn”和“+ dn”行形成一种简单的直方图，例如，昆明 IGS 网站，2007 1 月 17 日（TurboRogue 接收器只有 8 个频道）：

```
-dn | 1 12223321 122111123 11321111112211211 1111 | -dn
+ dn | 1211 1 11121
222421335433342222323335311223332122322232223211223 11 | + dn
```

接下来是可以在“+ XX”线上跟踪的理论 SV；再次，昆明 IGS 网站，2007 年 1 月 17 日：

```
+10 |
8898777777777777668aa989aabbbbbbaaaaaa9aaaa97899bbba9aaaaaaaaaaaa9999aa88
88 | +10
```

“Pos”线记录了在不同时期天线的计算代码位置的成功或不成功。通常，您应该看到每个仓位都记录了一个位置计算成功的 o。

ASCII 时间图中的最后一个符号行标记为“Clk”。该线表示观察时间标签中存在的具有+或-符号的所有毫秒接收器时钟复位，分别表示检测到正或负毫秒接收器时钟复位。可以放置在该线上的另一个符号是^，其在时钟符号层次结构中低于+或-，并且指示该时间仓中的至少一个错过的观察时期，但是观察采样间隔的正确值必须是 set（请参阅-0.int 选项）为此正常工作。添加此符号有助于揭示两件事：1）存在“微间隙”，即缺少数据周期小于-gap_mn 选项设置的数据周期，以及 2）识别毫秒时钟复位可能具有的短间隙发生了。例如，如果“Clk”符号轨道的一部分是：

```
Clk: + + + ^ ^ + + + ^ + + + + +
```

然后你可以怀疑在时间（1）丢失（正）毫秒接收机时钟复位，因为其余的识别复位和缺失的纪元指示符[^]的规律性，以及其他时间（2）和（3）的其他缺失的观察时期。数据中可能存在其他微小间隙，但它们的存在将被+符号隐藏。

顺便提一下，在“+ dn”线上存在 qc 全运行的另一个“微间隙”指示符。因为这是可以观察到的 SV 的数量与实际观察到的 SV 之间的最大差异。然而，对于缺少观察时期（没有观察到 SV），而不是根据星历表计算仅仅“可以观察到的 SV”， teqc 放置接收器允许的最大数量。因此，对于能够跟踪 12 个 SV 的接收器（参见-max_rx_SVs 选项），当缺少观察时期时，您还将在此线上看到 c（十六进制为 12）。

在 ASCII 时间图的“Clk”行之后是带刻度标记的比例尺。刻度线之间的间隔在“时间线窗口长度”处指示较低的几行。刻度标记应出现在刻度间隔的偶数值处。例如，如果时间窗口从 01: 39: 30.000（1 小时 39 分 30 秒）开始，并且勾选间隔为 3 小时，则刻度线将放置在 03: 00: 00, 06: 00: 00, 09: 00: 00，依此类推，以 ASCII 图上的最佳分辨率。tic 间隔是自缩放的，从 0.1 秒到 7 天，具体取决于时间窗口的长度。

在 ASCII 时间图的末尾，开始和结束时间和日期在比例尺的末尾。秒数仅在非零时打印出来。

在简短报告段中的 ASCII 时间图之后是报告摘要。首先是目标文件名称的列表，如果使用 qc -full，则使用 RINEX [NAV](#) 文件。

然后显示时间窗口的边界。如果第一和/或最后一个观察时期的时间与时间窗口的界限不匹配，则也显示这些时期。

如果使用了配置环境变量或任何配置文件，则会列出这些变量。

如果观察间隔不为零，则给出。

然后给出具有任何类型观测的卫星（SV）的总数。接下来是缺少 SV 的列表，默认情况下最大设置（可能是 32），或者使用-prNs 选项，适用于拥有任何成员的所有卫星系统。最后，如果执行 qc -full，则会给出没有星历信息的 SV 列表。

然后给出可以由接收器同时跟踪的 SV 的数量。它当前的默认值为 12，或者可以使用-max_rx_SVs 选项进行更改。

如果观察间隔不为零，则给出时间窗口中可能的观察时期的总数。这是从至少一个 SV 实际上具有“完整观察”的时期的数量。

“完整观察”的定义很重要，因此将在此详细定义。为了使 GPS SV 的观测“完整”，它必须具有：

1. P1 或 C / A 代码数据
2. P2 代码数据
3. L1 和 L2 阶段数据
4. L1 和 L2 的 S / N 均等于或高于指定的最小值
5. 如果 qc -full，SV 高程等于或高于高程掩码

然后给出可能的，完整的和删除的观察的数量。如果执行 qc -full，则首先给出高于地平线和高程掩模之上的可能观测数量。接下来，给出完整观察的数量； 如果 qc -full，这仅限于高程掩码之上的那些观察。接下

来，给出删除的观察数量； 如果 qc -full，这也仅限于高程掩模之上的那些观察。

如果设置了 multipath 选项（默认情况下），则给出平均多路径 RMS。如果使用移动平均窗口（默认情况下使用），则会给出有关此窗口长度的信息。如果 qc-full，多路径 RMS 仅用于高程掩模之上的观测。

然后给出检测到的毫秒接收器时钟复位的数量。接下来是接收器时钟的总漂移，平均接收器时钟漂移的估计，以及如果时钟复位的数量非零，则是以分钟为单位的重置之间的平均时间。

接下来给出报告 SV 数据间隙之前所需的时间长度。如果是 qc -lite，则还给出最大时间。

如果检测到 n 毫秒时钟滑动（+ cl 选项），则报告具有 n 毫秒时钟滑动的时期数。当具有多路径可观察量的所有 SV 必须具有相同大小的多径滑动到指定容差（毫秒分数）内时，会发生这种情况。

接下来是其他 n 毫秒多径滑动的数量，这些滑动不符合 n 毫秒时钟滑动的要求。给定非零容差，存在一些多径滑动落在容差内的可能性。因此，括号中给出了第二个值，这是时间窗口的多径滑动总数（没有高程掩模截止）。如果公差设置为 1e-2 毫秒，则由于偶然性，预计比率为 2: 100。显着较高的比率表明患病的接收者。

接下来，如果进行电离层延迟可观察（+ iod）或多径（+ mp）的导数，则给出 IOD 和/或多径漂移数的计数。如果 qc-full，根据高程面具进一步细分。为了在此处有资格作为计数，MP1 和 MP2 必须在特定 SV 的相同时期滑动（但不一定是相同的量）。

最后，打印一个“SUM”行，显示窗口的开始和结束时间，以小时为单位的时间窗口的长度，以秒为单位的观察间隔，可能的观察数量（如果 qc -full），数量完整的观察，完全可能的观察结果为百分比的比率（如果 QC -full），对于 MP1 和 MP2 的多路径的 RMS 值（通过升降掩模如果限制 QC，最后的“每个滑观测” - full）。

“每次滑动观察”需要一些解释。首先，“观察”意味着如上定义的“完整观察”，包括如果 qc -full 的高程掩模。第二，“滑动”意味着“在完成对该 SV 的完全观察的时期期间发生了 IOD 滑动和/或 MP1 和 MP2 滑动”。

通过使用+ ssv 选项，可以在短报告段中包含每个 SV 的一些附加信息。与主要的 SUM 线类似，每个 SV 显示的观测值包括：预期的观测数量，完整观测数量，删除的观测数量，完成与可能观测的比率，MP1 和 MP2 的多径 RMS 值，以及每次滑动观察。

长报告段：

长报告段包含 SV 和高程（如果 qc -full）的进一步信息细分。在长报告段中，经常引用各个 SV。主角表示卫星系统：

- G: NAVSTAR GPS（美国）系统
- R: GLONASS（俄罗斯）系统
- E: 伽利略（欧洲）系统
- C: 北斗/指南针（中文）系统
- J: QZSS（日语）系统
- I: IRNSS（印度）系统
- S: SBAS（例如 WAAS, EGNOS）

第一部分是一些处理参数的列表，后面是时间窗内的第一个和最后一个观察时期的时间戳，以及观察间隔。

接下来是每个 SV 的观测细分。如果执行 `qc -full` 并且 SV 具有星历数据，则前四列中的值具有以下含义：

`#+ hor:`

此 SV 的地平线以上的观测数量

`<ele>:`

具有观测值的时期的 SV 的平均海拔高度

`#+面具:`

此 SV 的高程掩码上方的观察数量

`<ele>:`

具有观测值的时期的高程掩模上方的 SV 的平均海拔高度

接下来是报告的和完整的观察数量。如果执行 `qc -full` 并且 SV 具有星历数据，则值仅用于高程掩码之上的历元； 否则，值适用于所有时期：

`#reprt:`

报告此 SV 的任何数据的观测数量

`#compl:`

为此 SV 报告的“完整”观测数量（参见 [“短报告段”](#) 中的定义）

接下来是 L1, L2, P1, P2 和 C / A 观测的数量。同样，如果执行 `qc -full` 并且 SV 具有星历数据，则值仅用于高程掩码之上的历元； 否则，值适用于所有时期：

`L1:`

该 SV 的 L1 观测数

`L2:`

此 SV 的 L2 观测数

`P1:`

此 SV 的 P1 观测数

`P2:`

此 SV 的 P2 观测数

`CA:`

此 SV 的 C / A 观测数

如果执行 `qc -full`，则会计算任何计算出高于屏幕高亮但未报告任何数据的 SV。

如果执行 `qc -full`，则任何没有星历数据但具有任何类型的观察数据的 SV 都用“*”标识。

接下来，给出摘要计数。如果找到 `qc -full` 和站点位置，则给出高程掩模下方的观测总数（即，由于低仰角而被排除的观测数量）。接下来，总结了不完整观察的原因（如果找到了位置位置，则高于掩模掩模）：缺少 L1, L2, P1 或 C / A 或 P2，或 L1 或 L2 的差 S / N。

在此之后是使用任何代码或阶段数据报告的观察数量。（注意：如果 SV 只有，例如，多普勒数据，则此处不会报告。）接下来是因任何原因删除的观察次数：低于高程掩码（如果 `qc -full`），缺少代码或相位数据和/或差的 S / N。 最后，给出了完整观察的次数。

接下来，重复接收器时钟偏移和漂移统计。

接下来是几个高程直方图中的一个，从地平线角度（默认为 0° ）到天顶（ 90° ）。这些需要对演示风格进行一些解释。直方图的 x 或水平标记可以使用如下内容：

```
5  =%1 | m 15  =%2 | m
```

在这些情况下，直方图实际上是双直方图，使用“=”符号显示百分比和“|”用于显示米的符号和用于显示两个直方图出现位置的“#”符号。因此，直方图线条如下：

```
##### |||||
## =====
```

在第一行显示约 7% 和 1.1 米，在第二行显示 4% 和 0.1 米。直方图条的最右侧的“>”表示条形向右延伸。箱的 y 或垂直标记是从地平线到天顶的度数。

电离层延迟的第一个直方图。目前，不是 qc 中离子 RMS 的量度（因此全部是 值为零），因此唯一有效的比例是具有电离层延迟滑动的观测值的百分比。

在此之下是每个 SV 的 MP1 RMS 摘要和所有 SV 的总统计，然后是 MP1 RMS 直方图。每个 SV 的摘要显示了掩模角度以上的观测数量（默认值为 10° ）以及 MP1 观测值被删除的数量（由于缺少一个或多个可观察量），其次是平均海拔高度（以度为单位），以及 MP1 rms 以米为单位。在高度和低于 25° 的高度上还有 MP1 滑动的细分，以及接收器报告的 L1 和 L2 滑动的情况。对于所有 SV 的总统计量，遵循各个 SV MP1 统计。高程统计和直方图显示每个高程区的观测总数，检测到的 MP1 单据数和数字上的平均 MP1 rms，然后是以米为单位的 MP1 rms 的直方图条（如“|”符号）和 MP1 滑动的观察百分比（“=”符号）。（回想一下，如果“|”和“=”重叠，则使用“#”符号。）

MP2 摘要如下，类似于 MP1 摘要。

最后两个直方图是 L1 和 L2 SNR 值。这些也是双直方图，显示平均 SNR 与“|”符号和带有“=”符号的 one-sigma 值。（回想一下，如果“|”和“=”重叠，则使用“#”符号。）在这些直方图中，两个标度都使用相同的单位，这些单位是任意的。如果制造商的格式二进制文件用作 qc 的输入，或者具有可观察的 S1 和/或 S2 的 RINEX [OBS](#) 文件，则单元是接收器特定的。如果使用不带 S1 或 S2 的 RINEX [OBS](#) 文件，则会显示 SNR 的缩放 RINEX 0-9 标志。

22 “奇怪”行为

- 如果执行 qc 完整模式（即隐式或显式提供的 [NAV](#) 文件或使用二进制目标文件），qc full >>>>>> ... 指示符可能会在某些文件数据集上暂停然后似乎继续前进。不要惊慌。一切都正常运作。这就是发生的事情：

qc 完全模式真正开始于 qc lite 模式。当使用目标文件（而不是标准输入）时，teqc 可以去文件中的任意位置。qc 全程运行的第一个主要目标是找到天线的伪距点位置。在可能的情况下，需要一定量的信息。teqc 必须有这些 SV 的星历信息。偶尔，这不会在文件的早期发生。当它发生时，teqc 开始重新读取并重新处理目标文件，现在知道天线位置。如果已经请求了绘图文件，那

么这就是它们被写入的时间。您看到的暂停是 teqc 返回并重新执行所有这些项目并返回到确定点位置时的时间所需的时间。

- 如果执行 qc 完整模式（即，隐式或显式提供的 [NAV](#) 文件或使用二进制目标文件）并且打开了绘图选项，但未找到任何位置（无论出于何种原因），则不会创建绘图文件。这是用于 qc 完全模式的逻辑的结果（参见上面的项目）。
- 二进制文件的直接 qc 产生的结果与 RINEX 文件的 qc 略有不同。这是由于二进制文件的直接 qc 被设计用于来自接收器的直接数据流，因此缺乏将数据流视为文件的能力。同样对于直接 qc，关于每个 SV 的高程和方位角的绘图文件信息将在第一个时期开始，其中天线位置和 SV 星历都是已知的，而对于 RINEX 文件的 qc，高程和方位角信息将是计算整个感兴趣的窗口。例如，通过使用 -nav 选项从同一站点和前一天预加载 RINEX [NAV](#) 文件，通常可以正确地解决此问题。

- 如果进行任何 qc 模式，用户打算输入 [NAV](#) 文件，但使用 + nav 文件名 而不是 -nav 文件名，原始文件文件名可能会被重新打开和销毁。已经实施了一个安全措施，以帮助在订购 qc 命令时防止这种情况：

```
teqc [options] + qc [options] + nav filename [rest_of_command]
```

换句话说，在指定 RINEX [NAV](#) 文件名之前打开 qc 选项，该命令不涉及二进制的转换。在这种情况下，程序将不允许在“写入”模式下重新打开文件名，如果发生这种情况，则会破坏原始文件。

- 当使用了 Borland DOS 的壳版本 TEQC，通过写入到文件 ASCII 线 TEQC 以新行（ '\ n' = CTRL J = 0x0A）终止。向 stdout 注入的 ASCII 行然后重定向到文件将以回车符（ '\ r' = CTRL M = 0x0d）终止，然后是换行符。这个添加的回车显然是由 DOS shell 完成的。

该 WATCOM DOS 的外壳版本 TEQC 在这两种情况下被添加的额外的回车结果，无论是作为书面文件 TEQC 和标准输出重定向到一个文件。

像往常一样，如果在 ASCII 模式下将 DOS 创建的文件 ftp 到 UNIX，则删除添加的回车符；如果在二进制模式下将相同的文件 ftp 到 UNIX，则添加的回车符将保留在文件中。

翻译参考：<https://www.unavco.org/software/data-processing/teqc/tutorial/tutorial.html>

最后修改时间：2018 年 10 月 26 日星期五 14:13:59 UTC