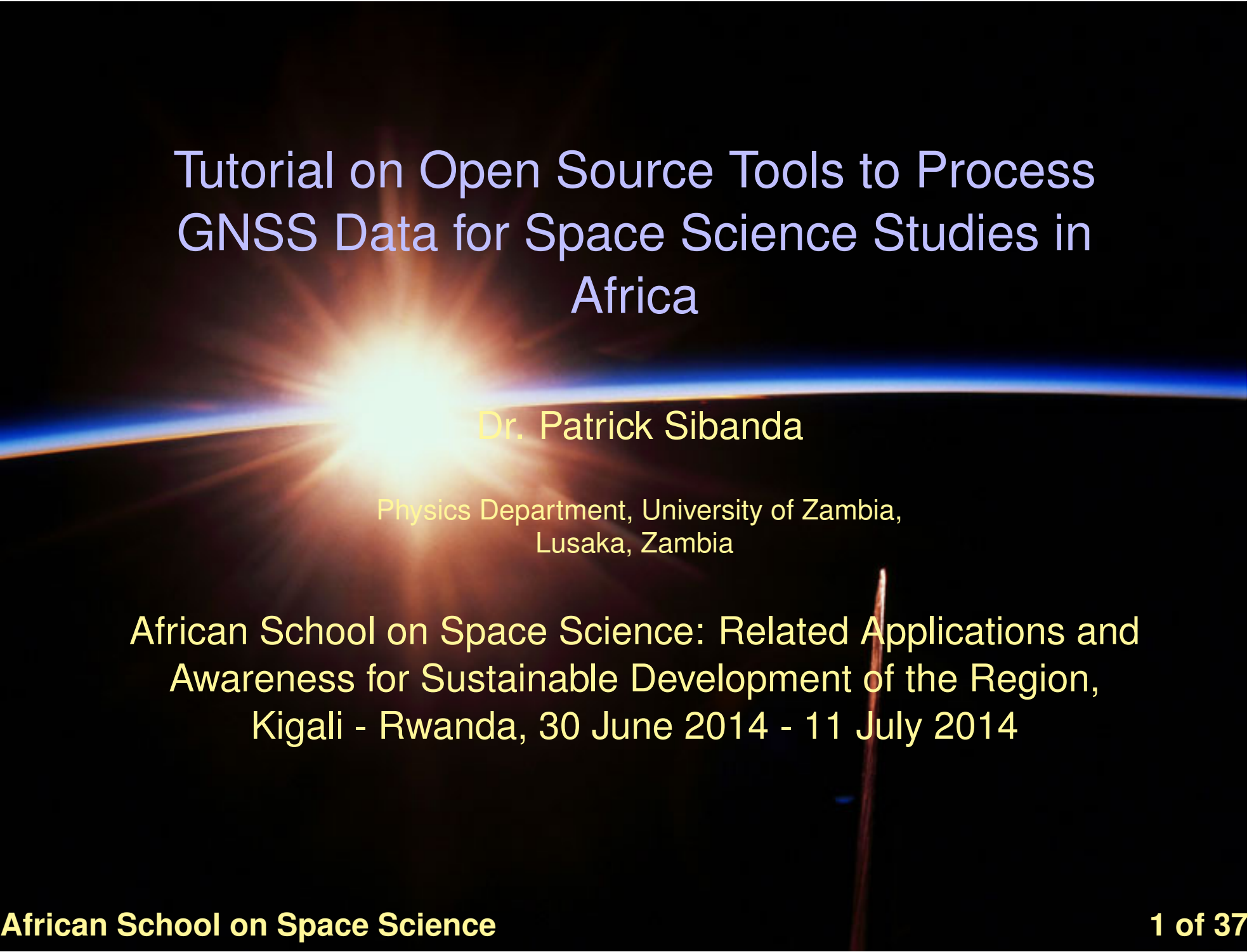


Tutorial on Open Source Tools to Process GNSS Data for Space Science Studies in Africa

Dr. Patrick Sibanda

Physics Department, University of Zambia,
Lusaka, Zambia

African School on Space Science: Related Applications and Awareness for Sustainable Development of the Region, Kigali - Rwanda, 30 June 2014 - 11 July 2014

The desired outcome:

...to visualize the data - in graphical form

In order to investigate the various trends such as

- Diurnal
- Seasonal
- geographical (latitude and longitude)

The desired outcome:

...to visualize the data - in graphical form

In order to investigate the various trends such as

- Diurnal
- Seasonal
- geographical (latitude and longitude)

..compute statistics such as

- mean, medians, variance etc

able to investigate the possible drivers of the observed phenomena

The desired outcome:

...to visualize the data - in graphical form

In order to investigate the various trends such as

- Diurnal
- Seasonal
- geographical (latitude and longitude)

..compute statistics such as

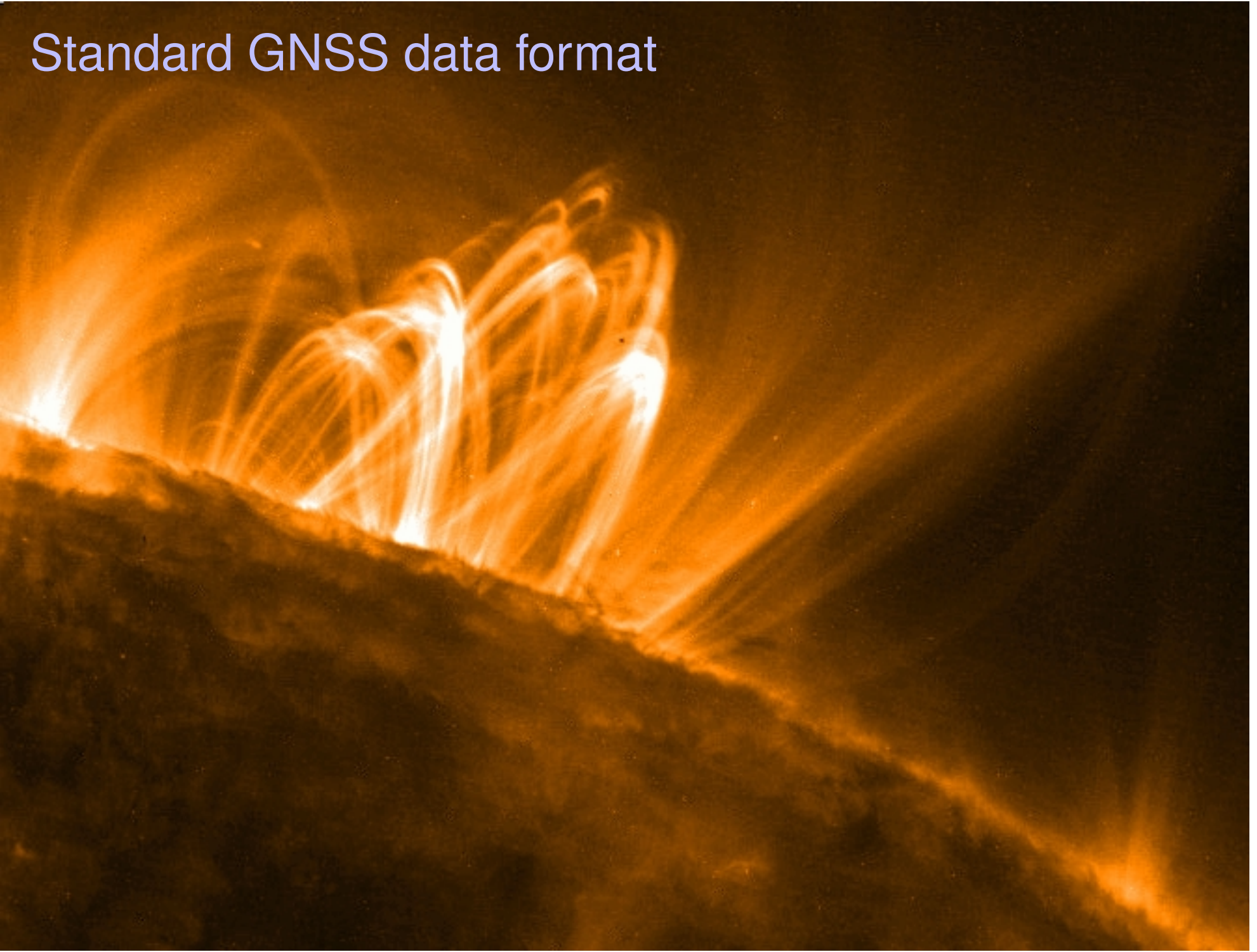
- mean, medians, variance etc

able to investigate the possible drivers of the observed phenomena

.. 2-D visualization

Observe and investigate how the disturbances propagate through space...

Standard GNSS data format



Standard GNSS data format

RINEX format

- compressed to enable easy transfer and sharing..

most common..

```
     2.11           OBSERVATION DATA    G (GPS)             RINEX
VERSION / TYPE
teqc  2012Oct31     UNAVCO Archive Ops  20121105 22:26:57UTCPGM /
RUN BY / DATE
Solaris x86 5.10|AMD64|cc SC5.8 -xarch=amd64|=+|=+          COMMENT
BIT 2 OF LLI FLAGS DATA COLLECTED UNDER A/S CONDITION       COMMENT
DODM                                                        MARKER
NAME
                                                            MARKER
NUMBER
Andy Nyblade        Pennsylvania State University
OBSERVER / AGENCY
4906K34480          TRIMBLE NETR8       4.14                REC # /
TYPE / VERS
4948353600          TRM59800.00     SCIT                    ANT # /
TYPE
  5147409.5002  3705355.3997  -682882.0001                  APPROX
POSITION XYZ
        0.0083        0.0000        0.0000                  ANTENNA:
DELTA H/E/N
     1     1
WAVELENGTH FACT L1/2
     7    L1    L2    C1    P2    P1    S1    S2           # /
TYPES OF OBSERV
    15.0000                                                 INTERVAL
    15                                                      LEAP
SECONDS
RINEX file created by UNAVCO GPS Archive.                   COMMENT
For more information contact archive@unavco.org             COMMENT
Monument ID: 23241                                          COMMENT
UNAVCO 4-char name:   DODM                                  COMMENT
4-char name from Log or data file: DODM                     COMMENT
Monument location: -6.18645594 35.74817297 1122.6189        COMMENT
Visit ID: 101228                                            COMMENT
End of DB comments                                          COMMENT
 SNR is mapped to RINEX snr flag value [0-9]                COMMENT
  L1 & L2: min(max(int(snr_dBHz/6), 0), 9)                  COMMENT
  2012     6    20     0     0    0.0000000     GPS         TIME OF
FIRST OBS
                                                            END OF
HEADER
 12  6 20  0  0  0.0000000  0 12G23G21G22G32G19G03G31G14G16G11G06G20
 122904135.546 7  95769481.87744  23387883.805    23387885.750
        45.300           29.500
 127464795.079 6  99323182.92142  24255732.281    24255738.211
        36.900           16.300
 133165397.211 6 103765258.85543  25340534.563    25340537.914
        38.300           19.300
 123478894.467 6  96217177.02643  23497213.945    23497221.164
        41.300           21.100
 115283738.579 8  89831627.54446  21937787.117    21937791.113
        50.100           40.100
 108090401.798 8  84226347.90147  20568936.492    20568943.184
        52.500           43.400
```

Standard GNSS data format

RINEX format

- compressed to enable easy transfer and sharing..
- not easy to work with

most common..

```
     2.11           OBSERVATION DATA    G (GPS)             RINEX
VERSION / TYPE
teqc  2012Oct31     UNAVCO Archive Ops  20121105 22:26:57UTCPGM /
RUN BY / DATE
Solaris x86 5.10|AMD64|cc SC5.8 -xarch=amd64|=+|=+          COMMENT
BIT 2 OF LLI FLAGS DATA COLLECTED UNDER A/S CONDITION       COMMENT
DODM                                                        MARKER
NAME
                                                            MARKER
NUMBER
Andy Nyblade        Pennsylvania State University
OBSERVER / AGENCY
4906K34480          TRIMBLE NETR8       4.14                REC # /
TYPE / VERS
4948353600          TRM59800.00     SCIT                    ANT # /
TYPE
  5147409.5002  3705355.3997  -682882.0001                  APPROX
POSITION XYZ
        0.0083        0.0000        0.0000                  ANTENNA:
DELTA H/E/N
     1     1
WAVELENGTH FACT L1/2
     7    L1    L2    C1    P2    P1    S1    S2           # /
TYPES OF OBSERV
    15.0000                                                 INTERVAL
    15                                                      LEAP
SECONDS
RINEX file created by UNAVCO GPS Archive.                   COMMENT
For more information contact archive@unavco.org             COMMENT
Monument ID: 23241                                          COMMENT
UNAVCO 4-char name:   DODM                                  COMMENT
4-char name from Log or data file: DODM                     COMMENT
Monument location: -6.18645594 35.74817297 1122.6189        COMMENT
Visit ID: 101228                                            COMMENT
End of DB comments                                          COMMENT
 SNR is mapped to RINEX snr flag value [0-9]                COMMENT
  L1 & L2: min(max(int(snr_dBHz/6), 0), 9)                  COMMENT
  2012     6    20     0     0    0.0000000     GPS         TIME OF
FIRST OBS
                                                            END OF
HEADER
 12  6 20  0  0  0.0000000  0 12G23G21G22G32G19G03G31G14G16G11G06G20
 122904135.546 7  95769481.87744  23387883.805    23387885.750
        45.300           29.500
 127464795.079 6  99323182.92142  24255732.281    24255738.211
        36.900           16.300
 133165397.211 6 103765258.85543  25340534.563    25340537.914
        38.300           19.300
 123478894.467 6  96217177.02643  23497213.945    23497221.164
        41.300           21.100
 115283738.579 8  89831627.54446  21937787.117    21937791.113
        50.100           40.100
 108090401.798 8  84226347.90147  20568936.492    20568943.184
        52.500           43.400
```

Standard GNSS data format

RINEX format

- compressed to enable easy transfer and sharing..
- not easy to work with
- only certain data are needed for a specific investigation

most common..

```
     2.11           OBSERVATION DATA    G (GPS)             RINEX
VERSION / TYPE
teqc  2012Oct31     UNAVCO Archive Ops  20121105 22:26:57UTCPGM /
RUN BY / DATE
Solaris x86 5.10|AMD64|cc SC5.8 -xarch=amd64|=+|=+          COMMENT
BIT 2 OF LLI FLAGS DATA COLLECTED UNDER A/S CONDITION       COMMENT
DODM                                                        MARKER
NAME
                                                            MARKER
NUMBER
Andy Nyblade        Pennsylvania State University
OBSERVER / AGENCY
4906K34480          TRIMBLE NETR8       4.14                REC # /
TYPE / VERS
4948353600          TRM59800.00     SCIT                    ANT # /
TYPE
  5147409.5002  3705355.3997  -682882.0001                  APPROX
POSITION XYZ
        0.0083        0.0000        0.0000                  ANTENNA:
DELTA H/E/N
     1     1
WAVELENGTH FACT L1/2
     7    L1    L2    C1    P2    P1    S1    S2           # /
TYPES OF OBSERV
    15.0000                                                 INTERVAL
    15                                                      LEAP
SECONDS
RINEX file created by UNAVCO GPS Archive.                   COMMENT
For more information contact archive@unavco.org             COMMENT
Monument ID: 23241                                          COMMENT
UNAVCO 4-char name:   DODM                                  COMMENT
4-char name from Log or data file: DODM                     COMMENT
Monument location: -6.18645594 35.74817297 1122.6189        COMMENT
Visit ID: 101228                                            COMMENT
End of DB comments                                          COMMENT
 SNR is mapped to RINEX snr flag value [0-9]                COMMENT
  L1 & L2: min(max(int(snr_dBHz/6), 0), 9)                  COMMENT
  2012     6    20     0     0    0.0000000     GPS         TIME OF
FIRST OBS
                                                            END OF
HEADER
 12  6 20  0  0  0.0000000  0 12G23G21G22G32G19G03G31G14G16G11G06G20
 122904135.546 7  95769481.87744  23387883.805    23387885.750
        45.300           29.500
 127464795.079 6  99323182.92142  24255732.281    24255738.211
        36.900           16.300
 133165397.211 6 103765258.85543  25340534.563    25340537.914
        38.300           19.300
 123478894.467 6  96217177.02643  23497213.945    23497221.164
        41.300           21.100
 115283738.579 8  89831627.54446  21937787.117    21937791.113
        50.100           40.100
 108090401.798 8  84226347.90147  20568936.492    20568943.184
        52.500           43.400
```

Standard GNSS data format

RINEX format

- compressed to enable easy transfer and sharing..
- not easy to work with
- only certain data are needed for a specific investigation

need to convert to format that are easier to work with - generally tables & extract only the necessary data

most common..

```
     2.11           OBSERVATION DATA    G (GPS)             RINEX
VERSION / TYPE
teqc  2012Oct31     UNAVCO Archive Ops  20121105 22:26:57UTCPGM /
RUN BY / DATE
Solaris x86 5.10|AMD64|cc SC5.8 -xarch=amd64|=+|=+          COMMENT
BIT 2 OF LLI FLAGS DATA COLLECTED UNDER A/S CONDITION       COMMENT
DODM                                                        MARKER
NAME
                                                            MARKER
NUMBER
Andy Nyblade        Pennsylvania State University
OBSERVER / AGENCY
4906K34480          TRIMBLE NETR8       4.14                REC # /
TYPE / VERS
4948353600          TRM59800.00     SCIT                    ANT # /
TYPE
  5147409.5002  3705355.3997  -682882.0001                  APPROX
POSITION XYZ
        0.0083        0.0000        0.0000                  ANTENNA:
DELTA H/E/N
     1     1
WAVELENGTH FACT L1/2
     7    L1    L2    C1    P2    P1    S1    S2           # /
TYPES OF OBSERV
    15.0000                                                 INTERVAL
    15                                                      LEAP
SECONDS
RINEX file created by UNAVCO GPS Archive.                   COMMENT
For more information contact archive@unavco.org             COMMENT
Monument ID: 23241                                          COMMENT
UNAVCO 4-char name:   DODM                                  COMMENT
4-char name from Log or data file: DODM                     COMMENT
Monument location: -6.18645594 35.74817297 1122.6189        COMMENT
Visit ID: 101228                                            COMMENT
End of DB comments                                          COMMENT
 SNR is mapped to RINEX snr flag value [0-9]                COMMENT
  L1 & L2: min(max(int(snr_dBHz/6), 0), 9)                  COMMENT
  2012     6    20     0     0    0.0000000     GPS         TIME OF
FIRST OBS
                                                            END OF
HEADER
 12  6 20  0  0  0.0000000  0 12G23G21G22G32G19G03G31G14G16G11G06G20
 122904135.546 7  95769481.87744  23387883.805    23387885.750
        45.300           29.500
 127464795.079 6  99323182.92142  24255732.281    24255738.211
        36.900           16.300
 133165397.211 6 103765258.85543  25340534.563    25340537.914
        38.300           19.300
 123478894.467 6  96217177.02643  23497213.945    23497221.164
        41.300           21.100
 115283738.579 8  89831627.54446  21937787.117    21937791.113
        50.100           40.100
 108090401.798 8  84226347.90147  20568936.492    20568943.184
        52.500           43.400
```

Computation of TEC from GNSS observables

...method to compute TEC from GPS observables

- described in literature e.g. Mannucci *et al.*, 1998 .

Computation of TEC from GNSS observables

...method to compute TEC from GPS observables

- described in literature e.g. Mannucci *et al.*, 1998 .

various tools to do this.. e.g

- GOPI_TEC - commonly used (GUI - relatively easy to use)

Computation of TEC from GNSS observables

...method to compute TEC from GPS observables

- described in literature e.g. Mannucci *et al.*, 1998 .

various tools to do this.. e.g

- GOPI_TEC - commonly used (GUI - relatively easy to use)

generally the desired output ...

- is tabulated TEC (VTEC) and other relevant parameters of interest such as time and date and lat & long

Computation of TEC from GNSS observables

...method to compute TEC from GPS observables

- described in literature e.g. Mannucci *et al.*, 1998 .

various tools to do this.. e.g

- GOPI_TEC - commonly used (GUI - relatively easy to use)

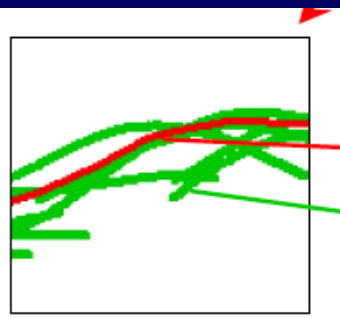
generally the desired output ...

- is tabulated TEC (VTEC) and other relevant parameters of interest such as time and date and lat & long

generally ...

- low level programming languages (e.g fortran or C) more suitable for this
- high level languages like Matlab and IDL quite slow for this

Taking the ASCII output files (*.CMN & *.STD)



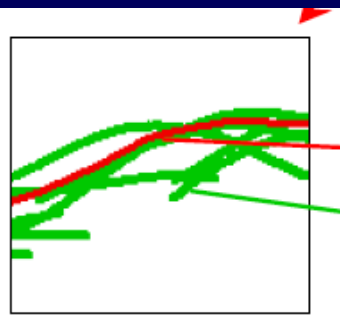
Red line – Average (2 sigma iterated) TEC over all PRNs (output file: *.std)

Green line – vertical TEC from individual PRNs, here data presented is **above 20 deg elevation** angles (output file: *.cmn, it contains all elevation angles)

Taking the ASCII output files (*.CMN & *.STD)

*.Cmn file - 10 columns separated by a tab

- Jdatet, Time, PRN, Az, Ele, Lat, Lon, Stec, Vtec, S4
- lines of file header
- vertical TEC from individual PRNs, > 20 deg elevation angles



Red line – Average (2 sigma iterated) TEC over all PRNs (output file: *.std)

Green line – vertical TEC from individual PRNs, here data presented is **above 20 deg elevation** angles (output file: *.cmn, it contains all elevation angles)

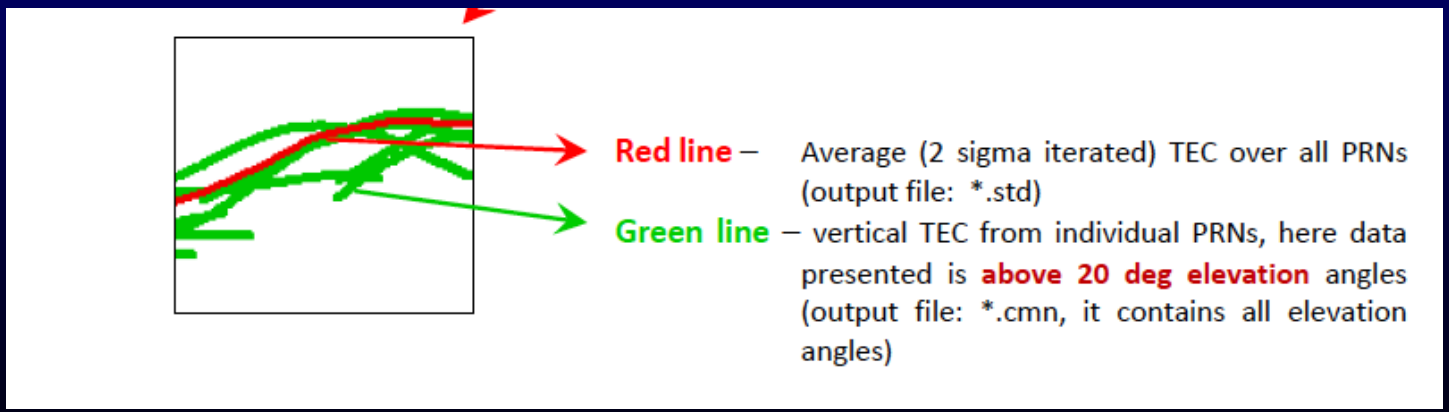
Taking the ASCII output files (*.CMN & *.STD)

*.Cmn file - 10 columns separated by a tab

- Jdatet, Time, PRN, Az, Ele, Lat, Lon, Stec, Vtec, S4
- lines of file header
- vertical TEC from individual PRNs, > 20 deg elevation angles

*.Std file - 4 columns separated by a tab

- text - header lines and for missing data
- Average (2 sigma iterated) TEC over all PRNs



data processing

large data volumes

often - have to process hundreds of files and and have to investigate many events

data processing

large data volumes

often - have to process hundreds of files and and have to investigate many events

need to automate the process

Processing the *.Cmn and *.Std files

Matlab and IDL

due to costs (recurring annually) - Not many groups in Africa have legal copies (PIRACY is a CRIME)

Processing the *.Cmn and *.Std files

Matlab and IDL

due to costs (recurring annually) - Not many groups in Africa have legal copies (PIRACY is a CRIME)

This tutorial

- introduce open source tools, easy to use to process this kind of data

Processing the *.Cmn and *.Std files

Matlab and IDL

due to costs (recurring annually) - Not many groups in Africa have legal copies (PIRACY is a CRIME)

This tutorial

- introduce open source tools, easy to use to process this kind of data
  - just the basic shell UNIX core utilities (recent Linux distributions very easy to use)

Processing the *.Cmn and *.Std files

Matlab and IDL

due to costs (recurring annually) - Not many groups in Africa have legal copies (PIRACY is a CRIME)

This tutorial

- introduce open source tools, easy to use to process this kind of data
  - just the basic shell UNIX core utilities (recent Linux distributions very easy to use)
- do away with the illegal copies Matlab or IDL

Processing the *.Cmn and *.Std files

Matlab and IDL

due to costs (recurring annually) - Not many groups in Africa have legal copies (PIRACY is a CRIME)

This tutorial

- introduce open source tools, easy to use to process this kind of data
  - just the basic shell UNIX core utilities (recent Linux distributions very easy to use)
- do away with the illegal copies Matlab or IDL
- on windows, install Cygwin - then run all UNIX commands

Cygwin
Install Cygwin
Update Cygwin
Search Packages
Licensing Terms

Cygwin/X

Community
Reporting Problems
Mailing Lists
Newsgroups
Gold Stars
Mirror Sites
Donations

Documentation
FAQ
User's Guide
API Reference
Acronyms

Contributing
Snapshots
Source in CVS
Cygwin Packages

Related Sites
Red Hat Cygwin Product

Cygwin

Get that Linux feeling - on Windows

Installing and Updating Cygwin Packages

Installing and Updating Cygwin for 32-bit versions of Windows

Run setup-x86.exe any time you want to update or install a Cygwin package for 32-bit windows. The signature for setup-x86.exe can be used to verify the validity of this binary using this public key.

Installing and Updating Cygwin for 64-bit versions of Windows

Run setup-x86_64.exe any time you want to update or install a Cygwin package for 64-bit windows. The signature for setup-x86_64.exe can be used to verify the validity of this binary using this public key.

General installation notes

When installing packages for the first time, `setup*.exe` *does not install every package*. Only the **minimal base packages** from the Cygwin distribution are installed by default. Clicking on categories and packages in the `setup*.exe` package installation screen will provide you with the ability to control what is installed or updated. Clicking on the "Default" field next to the "All" category will provide you with the opportunity to install every Cygwin package. Be advised that this will download and install hundreds of megabytes to your computer. The best plan is probably to click on individual categories and install either entire categories or packages from the categories themselves.

The latest net releases of the Cygwin DLL are numbered *1.n.x*, where "n" is currently "7" (e.g., 1.7.5). The *1.n.x* version numbering refers only to the Cygwin DLL. Individual packages like *bash*, *gcc*, *less*, etc. are released independently of the DLL. The `setup*.exe` utility tracks the versions of all installed components and provides the mechanism for **installing** or **updating** everything available from this site for Cygwin.

Once you've installed your desired subset of the Cygwin distribution, setup*.exe will remember what you selected so rerunning the program will update your system with any new package releases.

On Windows Vista and later, `setup*.exe` will check by default if it runs with administrative privileges and, if not, will try to elevate the process. If you want to avoid this behaviour and install under an unprivileged account just for your own usage, run `setup*.exe` with the `--no-admin` option.

The `setup*.exe` installer is designed to be easy for new users to understand while remaining flexible for the experienced. The volunteer development team is constantly working on `setup*.exe`; before requesting a new feature, check the wishlist in the CVS README.

Cygwin install on windows

Cygwin is

- Unix-like environment and command-line interface for Microsoft Windows.

Cygwin install on windows

Cygwin is

- Unix-like environment and command-line interface for Microsoft Windows.
- a large collection of GNU and Open Source tools which provide functionality similar to a Linux distribution on Windows.

Cygwin install on windows

Cygwin is

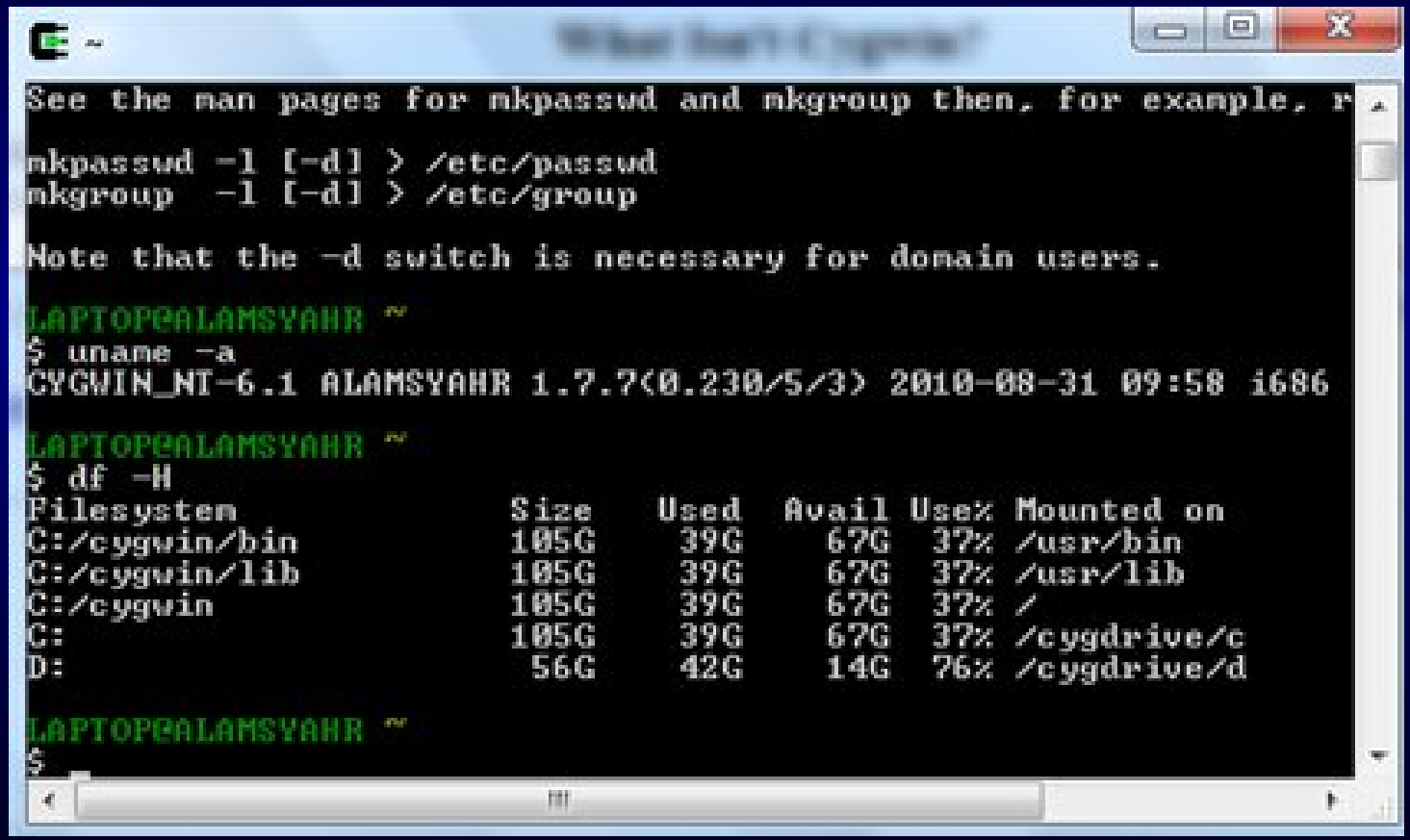
- Unix-like environment and command-line interface for Microsoft Windows.
- a large collection of GNU and Open Source tools which provide functionality similar to a Linux distribution on Windows.

To install

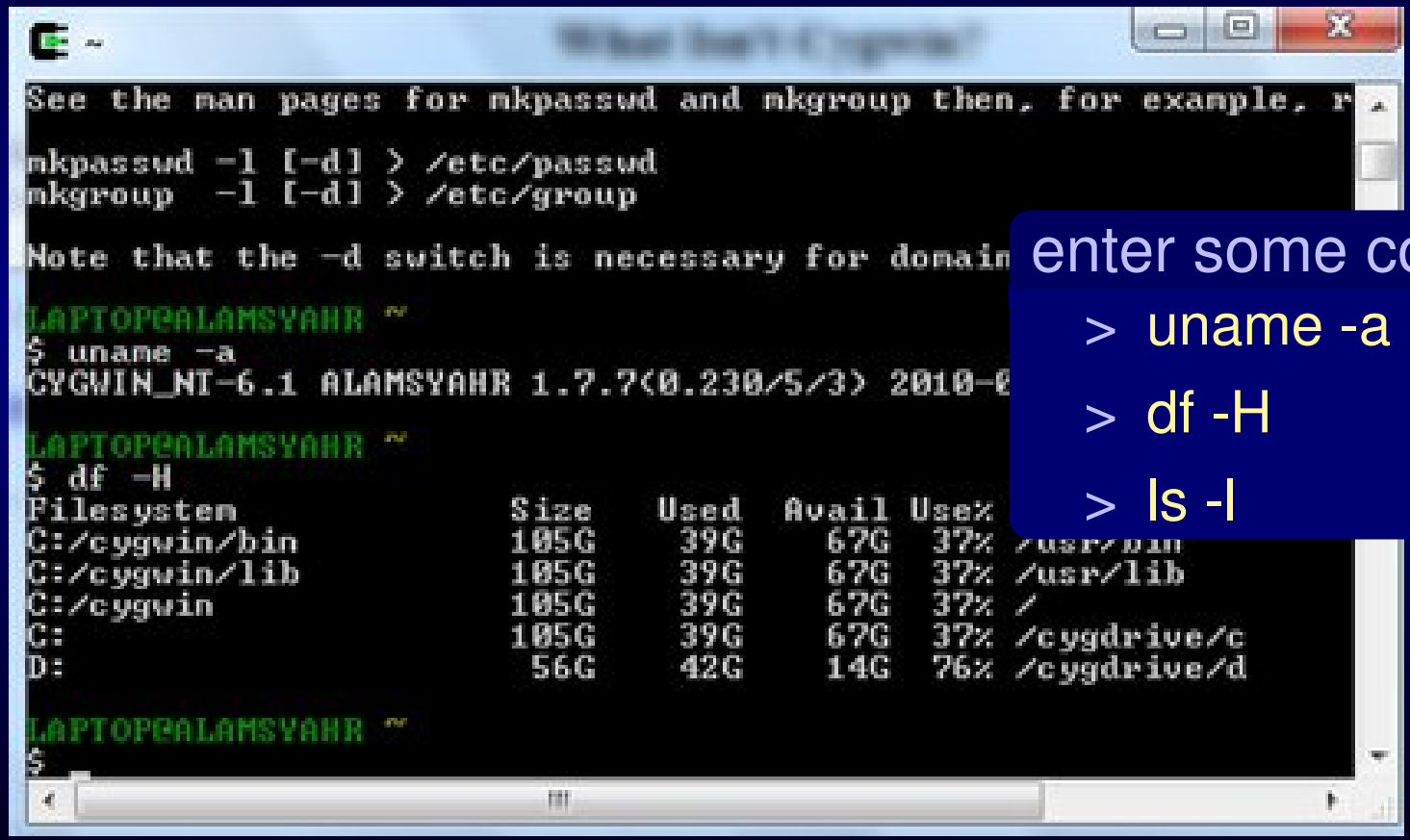
- go to `http://cygwin.com/` and run either setup-x86.exe to install the 32 bit version of Cygwin, or setup-x86_64.exe to install the 64 bit version of Cygwin
- GUI installer which can be run to download a complete cygwin installation via the internet

Click Start menu and Find Cygwin . Click on Cygwin Bash Shell to start the shell.

Click Start menu and Find Cygwin . Click on Cygwin Bash Shell to start the shell.



Click Start menu and Find Cygwin . Click on Cygwin Bash Shell to start the shell.



enter some command like

- > uname -a
- > df -H
- > ls -l

install cygwin components from the command line

- Two package managers apt-cyg of apt-get

install cygwin components from the command line

- Two package managers apt-cyg of apt-get

to install apt-cyg

```
> wget raw.github.com/transcode-open/apt-cyg/master/apt-cyg
> chmod +x apt-cyg
> mv apt-cyg /usr/local/bin
```

install cygwin components from the command line

- Two package managers apt-cyg of apt-get

to install apt-cyg

```
> wget raw.github.com/transcode-open/apt-cyg/master/apt-cyg
> chmod +x apt-cyg
> mv apt-cyg /usr/local/bin
```

use apt-cyg to install additional packages

```
> apt-cyg install nano
> apt-cyg install git
> apt-cyg install ca-certificates
```

install cygwin components from the command line

- Two package managers apt-cyg of apt-get

to install apt-cyg

```
> wget raw.github.com/transcode-open/apt-cyg/master/apt-cyg
> chmod +x apt-cyg
> mv apt-cyg /usr/local/bin
```

use apt-cyg to install additional packages

```
> apt-cyg install nano
> apt-cyg install git
> apt-cyg install ca-certificates
```

can also use cygwin's setup

setup-x86.exe -q -P packagename1,packagename2
See `http://cygwin.com/packages/` for the package list.

Linux, Awk and R

- will use basic Linux Shell Commands

Linux, Awk and R

- will use basic Linux Shell Commands

Linux Command Syntax

<command> <option(s)> <argument(s)>

Linux, Awk and R

- will use basic Linux Shell Commands

Linux Command Syntax

<command> <option(s)> <argument(s)>

familiarize with the following commands

1. echo
2. ls
3. mv
4. date
5. expr or bc
6. sort

Linux, Awk and R

- will use basic Linux Shell Commands

Linux Command Syntax

<command> <option(s)> <argument(s)>

familiarize with the following commands

1. echo
2. ls
3. mv
4. date
5. expr or bc
6. sort

- then use AWK or R to manipulate the ASCII files

AWK - to process the ASCII files

- a full-featured programming language for processing text files
- can write well-structured large programs
- exist on almost all Unix-like operating systems

AWK - to process the ASCII files

- a full-featured programming language for processing text files
- can write well-structured large programs
- exist on almost all Unix-like operating systems

only one syntactic construct to understand

- awk <search pattern> {<program actions >}

AWK - to process the ASCII files

- a full-featured programming language for processing text files
- can write well-structured large programs
- exist on almost all Unix-like operating systems

only one syntactic construct to understand

- awk <search pattern> {<program actions >}
- Each line of the input data is tested against the search pattern

AWK - to process the ASCII files

- a full-featured programming language for processing text files
- can write well-structured large programs
- exist on almost all Unix-like operating systems

only one syntactic construct to understand

- awk <search pattern> {<program actions >}
- Each line of the input data is tested against the search pattern
- If the line matches the search pattern the commands in parentheses are executed

AWK - to process the ASCII files

two special search patterns BEGIN and END

- program actions after BEGIN are executed before any line of the input data has been processed

AWK - to process the ASCII files

two special search patterns BEGIN and END

- program actions after BEGIN are executed before any line of the input data has been processed
- the program actions after END are executed after all lines of the input data have been processed

AWK example commands

```
echo -e "X\t0\t100\t2\nX\t100\t200\t4\nY\t0\t100\t3" > cov
```

Some simple awk one-liners without a search pattern would be.

```
awk '{print $1;}' cov
awk '{print $2;}' cov
awk '{print $4;}' cov
awk '{print $0;}' cov
```

As you can see Awk splits a tab-delimited file into variables. $0 contains the full input line, $1 column 1, $2 column 2, and so on. So to switch column 1 and 4 we simply type.

```
awk '{print $4"\t"$2"\t"$3"\t"$1;}' cov
```

The great thing about awk is that you can use it within pipes. The following command first sorts the file according to column 4 and then adds a line number as the first column. NR is a special variable that holds the line number of the input data.

```
sort -k4,4g cov
sort -k4,4g cov | awk '{print NR":"$0;}'
```

With that information you can now easily compute the sum and the mean of all coverage values.

```
awk '{SUM+=$4;} END {print SUM;}' cov
awk '{SUM+=$4;} END {print SUM / NR;}' cov
```

AWK example commands

With that information you can now easily compute the sum and the mean of all coverage values.

```
awk '{SUM+=$4;} END {print SUM;}' cov
awk '{SUM+=$4;} END {print SUM / NR;}' cov
```

For each line we add the fourth column to the variable SUM and when all lines have been process we simply output the sum. For the mean we have to divide by the number of lines, which is the final value of NR. The default initialization of SUM is 0 but we can make this also explicit using the BEGIN keyword.

```
awk 'BEGIN {SUM=0} {SUM+=$4;} END {print SUM;}' cov
awk 'BEGIN {SUM=1} {SUM+=$4;} END {print SUM;}' cov
```

Shell script to contain the commands

shell script

All Linux Commands - wrapped together into a shell script

first line of shell script

```
#!/bin/bash
```

make it executable then run it

```
> chmod +x myShell.sh
> ./myShell.sh inputs
```

Actual examples: process *.Cmn files

```
#!/bin/sh
#
# Purpose: Plot the GPS TEC for each prn from the GOPI
    software
# Unix progs: echo, awk, gnuplot, date, rm, mv
evel=40
#
if [ -z "$1" ]; then
  echo "Usage: $0 stationCODE doy year"
  exit 1
fi
stn=$1
doy=$2
doy=`echo $doy | awk  '{printf ("%3.3d",$1)}'`
yr=$3
#
```

Actual examples: process *.Cmn files

```
ddtte=`date -d "$yr -01-01 +$(( 10#$doy - 1 ))days" +%Y
    -%m-%d`
#
filename=$stn$doy'-'$ddtte.Cmn
filenameTEC=$stn$doy'-'$ddtte.Std # so we can
% plot thestd file aswel
echo 'processing file  ...' $filename
#
# Difne the data diretory
dir=/Volumes/SIBANDA32GB/Research/SA_Zambia/Data/Zambia/
    $yr
filz=$dir/$filename
filzTEC=$dir/$filenameTEC
```

Space Science is collaborative

```
# read data for each prn and output to a file named
    according to prn. Also select only the data for which
     elevation angle is greater than a chosen value, and
    ignoring negative TEC
awk '
$3 == n {
        if(f) close(f)
        #f="prn" n++".txt"
        f=sprintf("%3s%2.2d%4s","prn", n++, ".txt")
        next
}
{if ($5 >= '$evel' && $9 > 0 ) print $2, $9 > f
}' $filz
```

more processing

```
# insert blank lines in the files that constain gaps
    this is necessary when ploting in order not to
    connect lines where there are gaps
for i in `ls ./prn*.txt`
do
awk 'function abs(x){return (((x < 0.0) ? -x : x))} (abs
    ($1-prev)>1) {print ""} {print; prev=$1}' $i > prn.
    txt
mv prn.txt $i
# Extract a part of the filename for naming the output
    file
filenme=$i
ttl=`echo ${filenme:5:2}`
fil=`echo ${filenme:2:5}`
# for loop continue on next slide
```

```
# code continued from previous slide
  for kk in $fil ; do
   ((++II))
   files[$II]="${titlez[$II]}${COL_SEP}$kk"
  done
  for k in $ttl ; do
   ((++I))
   titlez[$I]="${titlez[$I]}${COL_SEP}$k"
  done
done
```

```
# compute averages for each epoch:- ie take all the TEC
    measured at some epoch and calcuate the mean TEC for
    the that epoch -- but first, we create a file temp.d
    containing only the time and TEC columns which we use
    ... the first value is a -24.00 which should really
    be 0.00 so we replace accordingly first and then do
    the averaging and finaly sort the output
awk 'NR > 5 {if ($5 > '$evel' && $9 > 0 )  print $2, $9
    }' $filz > temp.d
awk '{if($1<0)$1=0}
{
    sum[$1]+=$2
    cnt[$1]++
}
END {
    for (i in sum)
        printf "%8.5f    %6.2f    %6d    %6.3f\n", i, sum[i
            ], cnt[i], sum[i]/cnt[i]
}' temp.d  | sort -n -k1 > TECavgFile.txt
```

Finally - done almost all we want

Re-sample the data

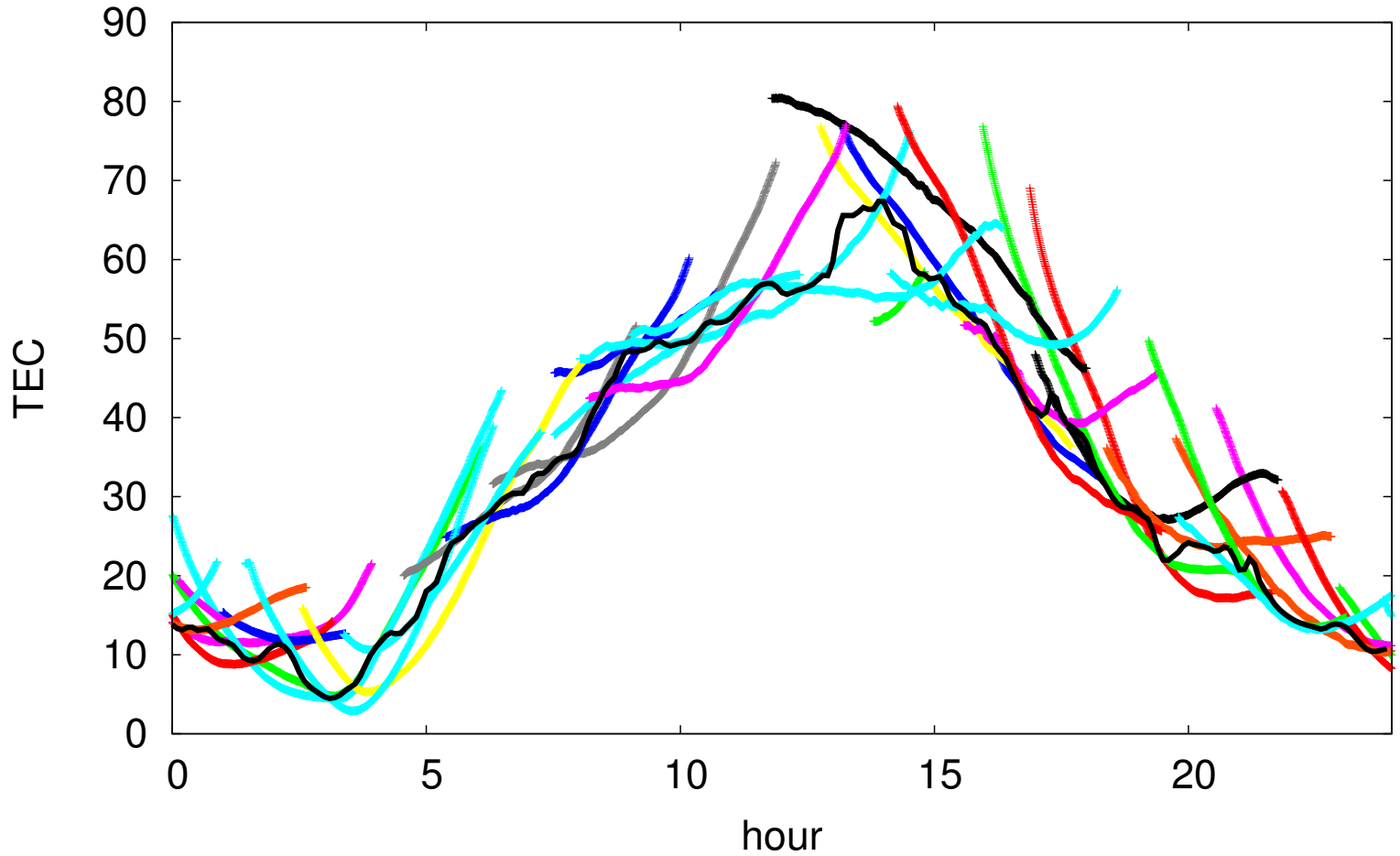
```
# This part resamples the averaged TEC every 3 minutes,
    interpolates using nearest neighbour when time
    interval is missing.
awk 'p/0.05<cnt&&$1/0.05>cnt{printf "%5.2f   %-6.3f\n",
    sprintf("%.5f",0.05*cnt++),(pv+$4)/2;p=$1;pv=$4;next}
   {p=$1;pv=$4}$1*10000%500==0{printf "%5.2f   %-6.3f\n"
       ,  $1, $4;cnt++;next}' TECavgFile.txt >
       TECavgResample.txt
# Its done now...we go ahead and plot the data.
```

For a good impression of a data set - need a picture

GNUPLOT best for this purpose

```
gnuplot << EOF
set size 1.5,1.3
set terminal postscript eps enhanced color "Helvetica"
    30
set output "prnTEC_Plot.eps"
set ylabel "TEC"
set xlabel "hour"
set xrange [0:24]
set yrange [0:90]
set nokey
# This is one way of looping through the files
title_list = '${files[*]}'
item(n) = word(title_list,n)
filename(n) = sprintf("prn%2.2d.txt", n)
plot for [i=1:$I] filename(i) using 1:2 w p pt 1 t item(
    i-1), 'TECavgResample.txt' u 1:2 w l lt -1 lw 4
EOF
```

The output.....



Investigation of TIDs for the storms in 2012

here the storms..

```
# ---------------------------------
# Defining the storm periods for the year:- begin day
    and end day
# ---------------------------------
# stormDates=[07 11 03;
#             22 26 04;
#             13 18 07;   && $9 < 40
#             29 03 09;
#             07 11 10;
#             12 16 11];
```

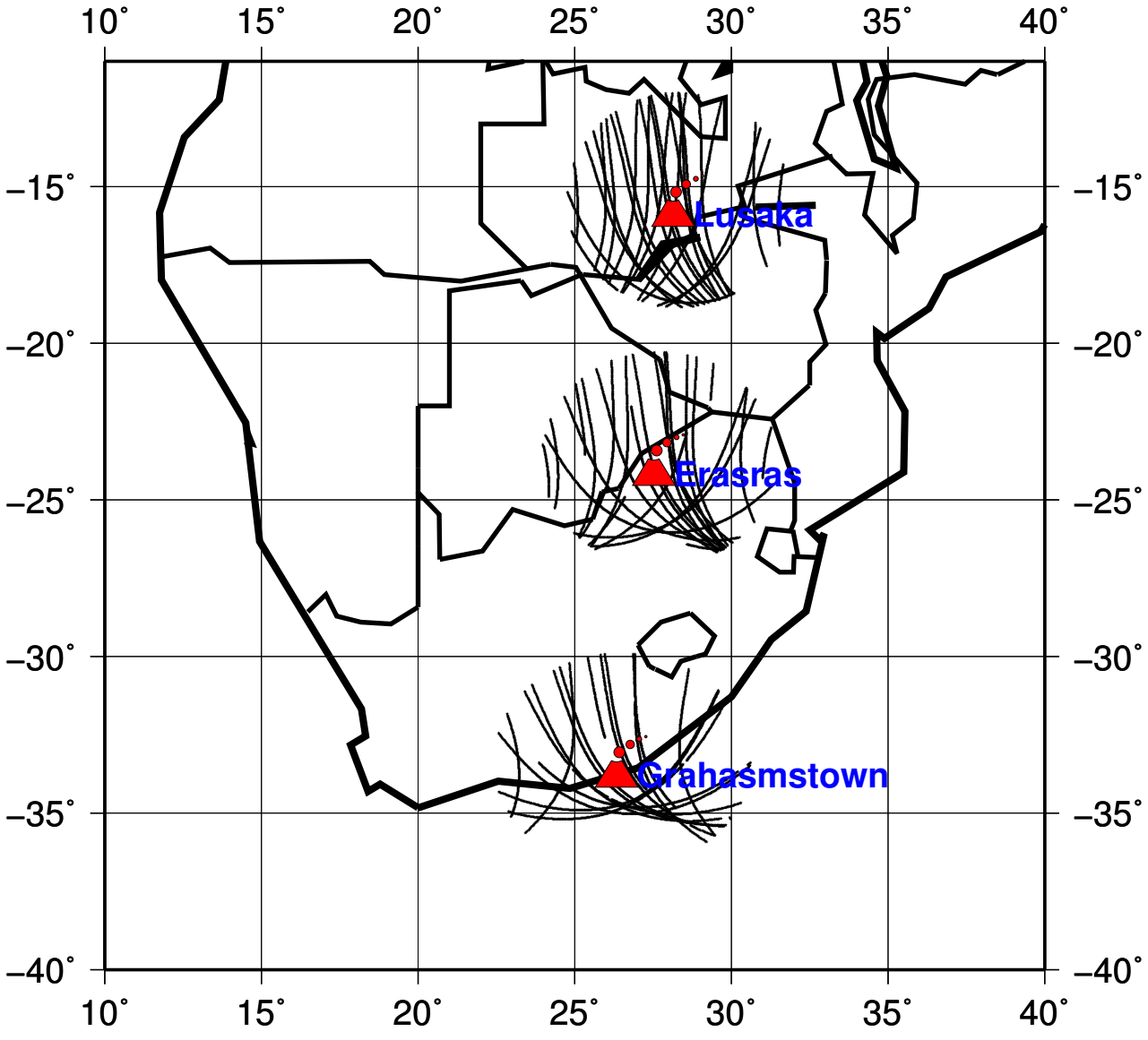
this GMT code to plot GPS array along a meridian

```
#!/bin/bash
#                GMT Script
#
# Purpose:       Plot GPS array and GPS ipp
gmtset BASEMAP_TYPE plain
gmtset COLOR_MODEL rgb
gmtset PAGE_ORIENTATION PORTRAIT
st_bndry=/usr/local
proj="Q0/5.2i"
# zone="g"
zone="10/40/-40/-11"
anot="a5g5WESN"
#
evel=40
# ----------------------------
psbasemap -R$zone -J$proj -B$anot -K -X1.5 -Y1.5 >
    GPSArrays.ps
pscoast -J -R -Dc -B -N1/2.0p -A1000 -W3p -O -K >>
    GPSArrays.ps
# ----------------------------
```

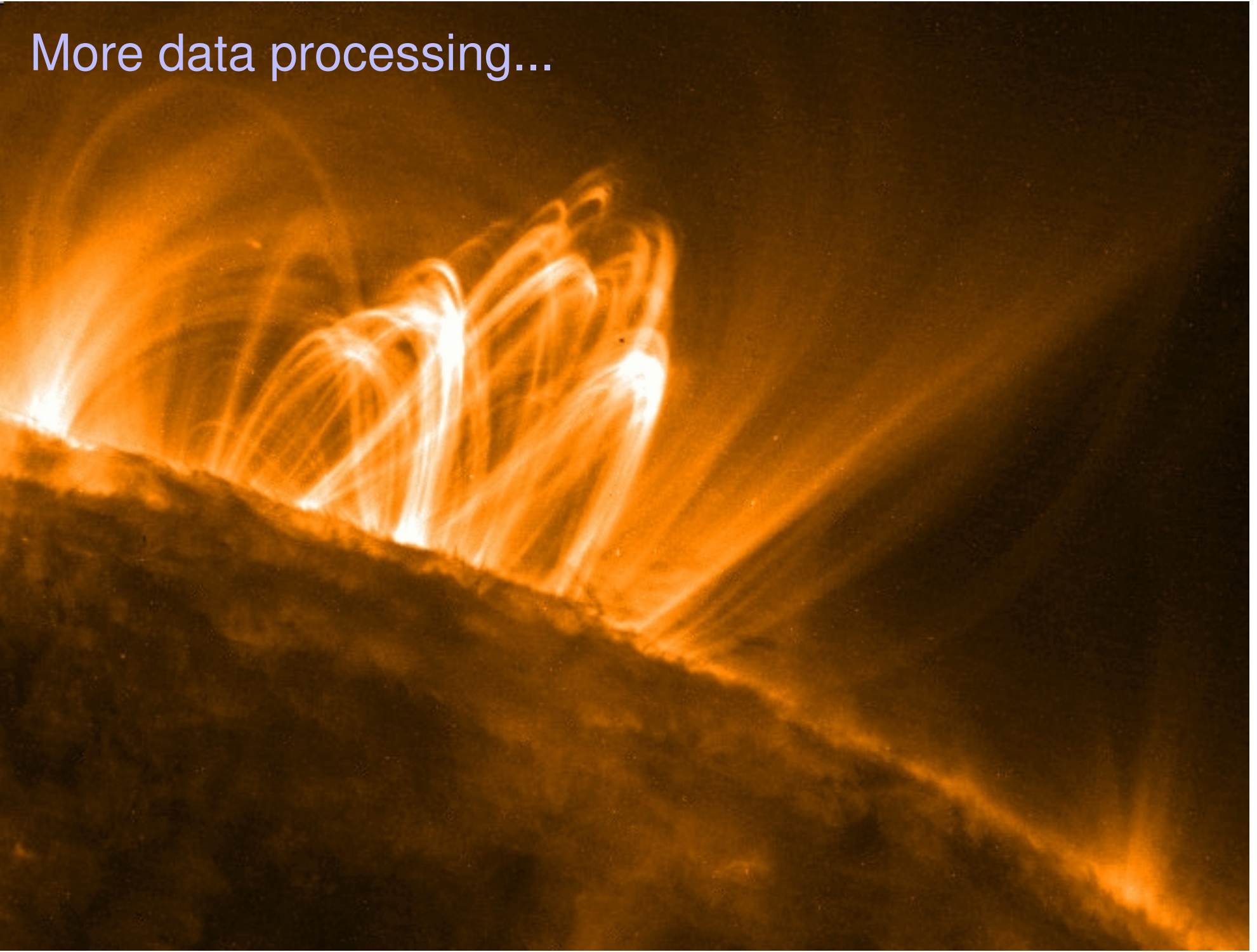
this GMT code to plot GPS array along a meridian

```
# Array 1
for i in zamb198* ERAS198* GRHM198*
do
awk 'NR > 5 {if ($5 > '$evel' && $9 > 0 )  print $7, $6
   }' $i > temp.d
psxy -R -J temp.d -Sa0.05 -O -K -Gblack >> GPSArrays.ps
done
# ----------------------------
awk '{if (NR > 1) print $3, $2, 0.75}' SA_ZM_Aray_GPS.
   txt > GPS_AfricaArrayStatns.d
psxy -R -J GPS_AfricaArrayStatns.d -Skvolcano -O -K -
   Wthinnest -Gred >> GPSArrays.ps
# ----------------------------
# Label the stations by their names
awk '{if (NR > 1) print $3-0.1, $2, 14, 0, 1, "LB", $4}'
    SA_ZM_Aray_GPS.txt > GPS_StatineNames.txt
pstext -R -J -O -D0.1i/-0.15i -Gblue GPS_StatineNames.
   txt >> GPSArrays.ps
# ----------------------------
ps2raster GPSArrays.ps -A -Tef
```

The output.....


10˚
15˚
20˚
25˚
30˚
35˚
40˚
–15˚
–20˚
–25˚
–30˚
–35˚
–40˚
Lusaka
Erasras
Grahasmstown

More data processing...



More data processing...

For the kind of study..

- Need to process data from the other 2 stations

More data processing...

For the kind of study..

- Need to process data from the other 2 stations
- compute other parameters like the monthly median for each station

More data processing...

For the kind of study..

- Need to process data from the other 2 stations
- compute other parameters like the monthly median for each station
- include a plot of the Geomagnetic ideces

Computing monthly median values

need data files for the entire month

```
#-------------------------------------
# Loop through the files for first to last day of the
    month to process the monthly median values
#-------------------------------------
for (( i=$firstDay; i <= $lastDAy; i++ ))
do
doys=`echo $i | awk  '{printf ("%3.3d",$1)}'`
ddtte=`date -d "$yr -01-01 +$(( 10#$doys - 1 ))days" +%Y
    -%m-%d`
#
filename=$stn$doys'-'$ddtte.Cmn
echo 'processing file  ...' $filename
```

Computing monthly median values

need data files for the entire month

```
# This part computes the monthly median values:- that is
    , the median value of the tec at each epoch
awk '{a[$1] = (a[$1] == "") ? $2 : (a[$1] "|" $2)}
  END {for(x in a)
    {split(a[x], b, "|"); n=asort(b, c);
 if(n % 2 == 1)
   {m = c[int(n /2) + 1]}
 else
   {m = (c[int(n /2)] + c[int(n /2) + 1]) / 2};
 printf "%5.2f   %6.3f\n", x, m}}' Resample*.txt | sort
    -n -k1 > MonthlMedian.txt
```

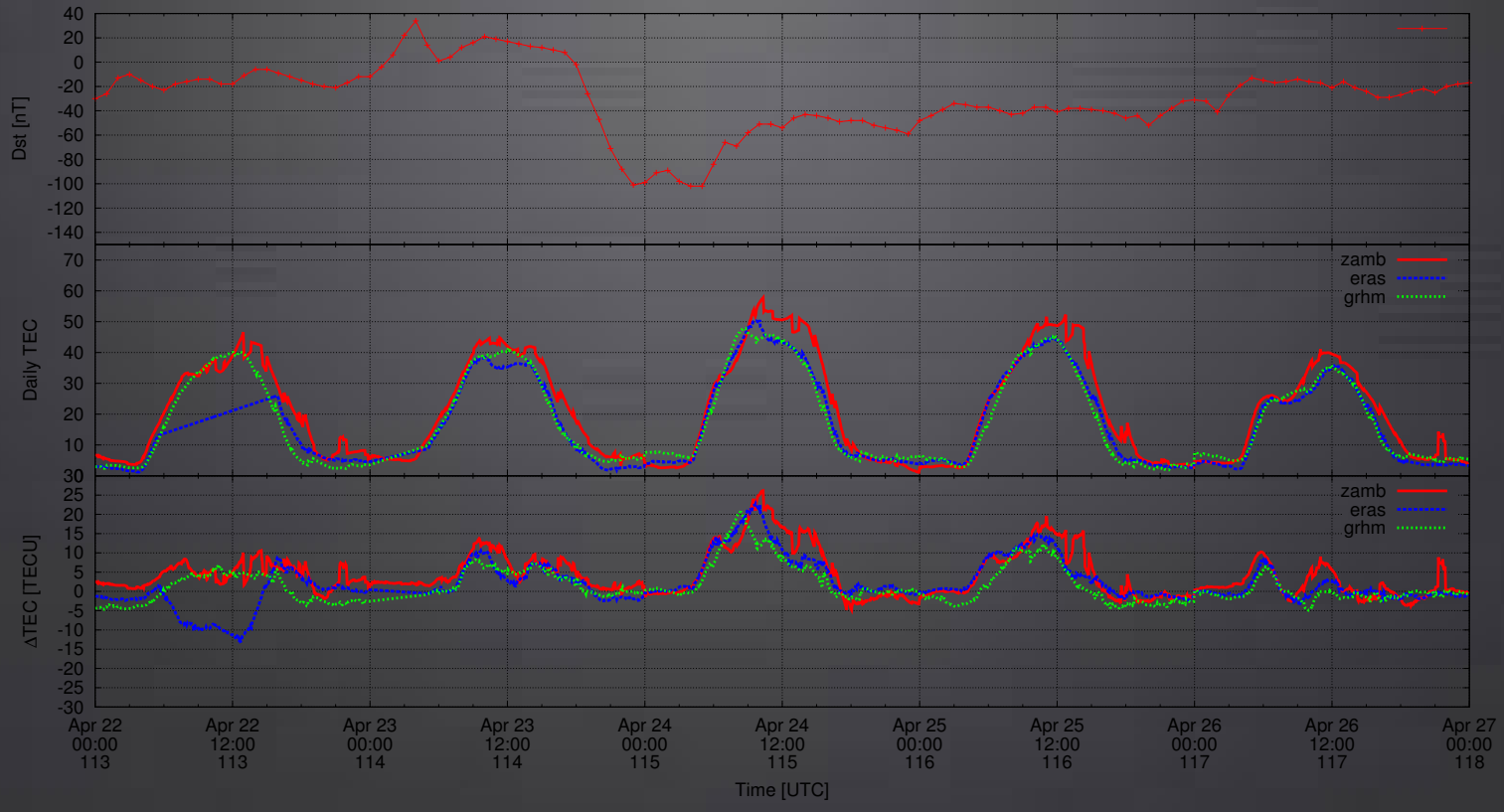
Select storm period data for plotting

```
# Take 2 days before and 2 after the storm period
for (( j=$strmBeg; j <= $strmEnd; j++ ))
do
dy=`date -d "$yr -01-01 +$(( 10#$j - 1 ))days" +%d`
my=`date -d "$yr -01-01 +$(( 10#$j - 1 ))days" +%m`
jj=`echo $j | awk  '{printf ("%3.3d",$1)}'`
# awk 'NR==FNR
#   {Ar[FNR]=$1FS$2;next}
#   {print Ar[FNR], $2}' Resample$j.txt MonthlMedian.txt
     >> StormData.txt
awk 'NR==FNR{a[NR]=$2; next}{hh=sprintf("%d\n",$1); min
   =($1-hh)*60;
  pival = sprintf("%4.4d-%2.2d-%2.2d_%2.2d:%2.2d:00", '$
     {yr}','${my}','${dy}',$1,min)}
  {printf "%19s   %5.2f   %6.3f   %6.3f   %6.3f\n",
  pival, $1, a[FNR], $2, a[FNR]-$2}' Resample$jj.txt
     MonthlMedian.txt >> StormData.txt
done
```

Process Dst data for the storm period

```
#-------------------------------------------
# Process dst Data for ploting
#-------------------------------------------
dstFile='../dstProcess/Dst2012.txt'
strtdte=`date -d "$yr -01-01 +$(( $strmBeg -1 ))days" +%
    Y-%m-%d`
enddate=`date -d "$yr -01-01 +$(( $strmEnd -1 ))days" +%
    Y-%m-%d`
# echo $strtdte $enddate
xr1=$strtdte"_00:00:00"
xr2=$enddate"_00:00:00"
strt=`awk '{if ($1 == "'$xr1'") print NR}' $dstFile`
endx=`awk '{if ($1 == "'$xr2'") print NR}' $dstFile`
echo "Date range is...:  "$xr1 $xr2
#
# Extract data for the given date range
awk '{ if (NR >= '$strt' && NR <= '$endx') { print $1,
    $2, $3}
      }' $dstFile > DstData.txt
```

The output.....


Dst [nT]
Daily TEC
ΔTEC [TECU]
zamb
eras
grhm
Apr 22 00:00 113
Apr 22 12:00 113
Apr 23 00:00 114
Apr 23 12:00 114
Apr 24 00:00 115
Apr 24 12:00 115
Apr 25 00:00 116
Apr 25 12:00 116
Apr 26 00:00 117
Apr 26 12:00 117
Apr 27 00:00 118
Time [UTC]

Thank you