ION GNSS+ 2018

The 31ST INTERNATIONAL TECHNICAL MEETING OF THE SATELLITE DIVISION OF THE INSTITUTE OF NAVIGATION

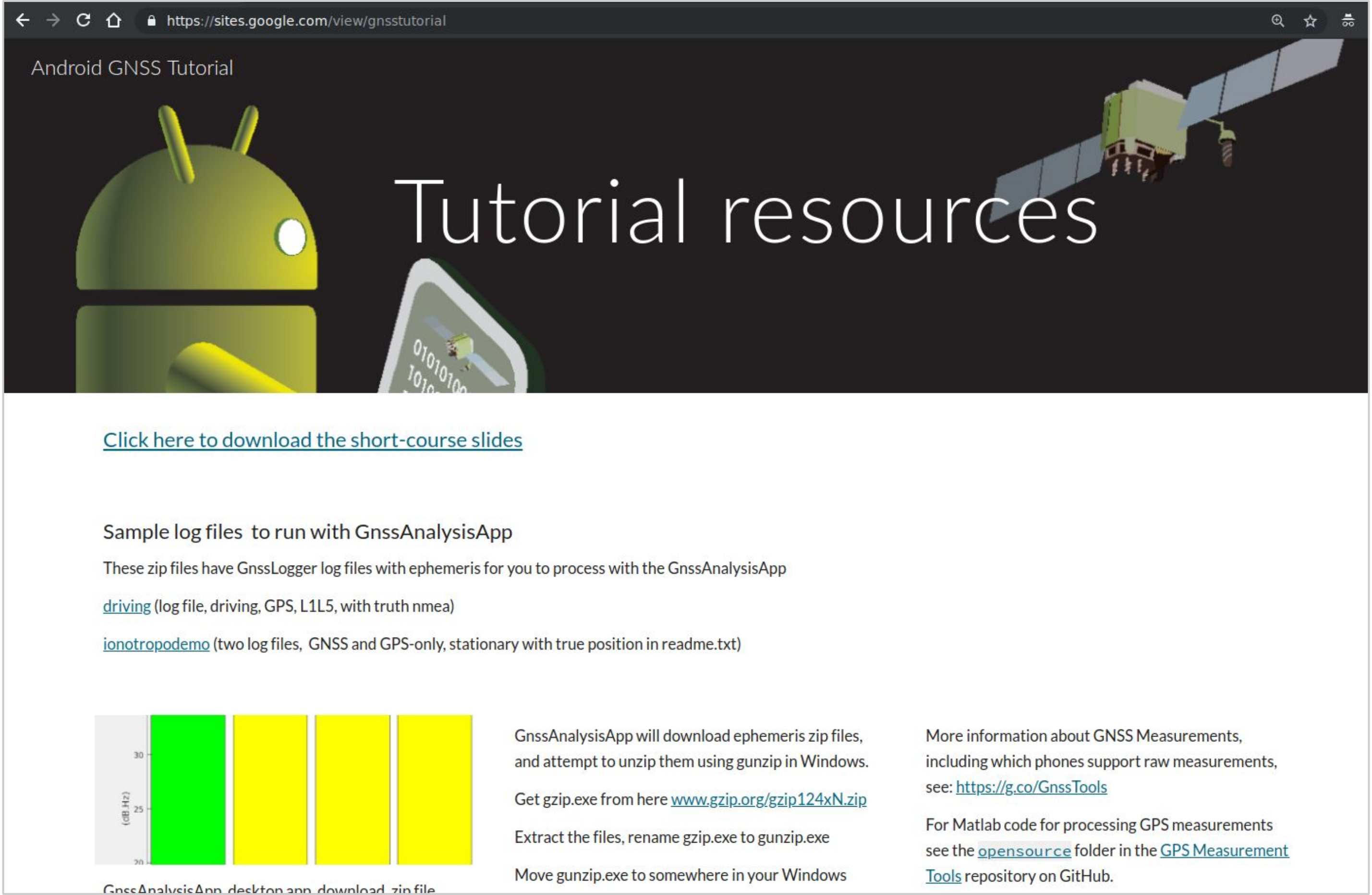
GNSS 102
Raw Measurements from Phones

ION Masters Course, Miami, 24 Sep 2018, v1.2

Frank van Diggelen

Google

This course has a support site: https://sites.google.com/view/gnsstutorial



Android GNSS Tutorial

Tutorial resources

Click here to download the short-course slides

Sample log files to run with GnssAnalysisApp

These zip files have GnssLogger log files with ephemeris for you to process with the GnssAnalysisApp

driving (log file, driving, GPS, L1L5, with truth nmea)

ionotropodemo (two log files, GNSS and GPS-only, stationary with true position in readme.txt)

GnssAnalysisApp desktop app download zip file

GnssAnalysisApp will download ephemeris zip files, and attempt to unzip them using gunzip in Windows.

Get gzip.exe from here www.gzip.org/gzip124xN.zip

Extract the files, rename gzip.exe to gunzip.exe

Move gunzip.exe to somewhere in your Windows

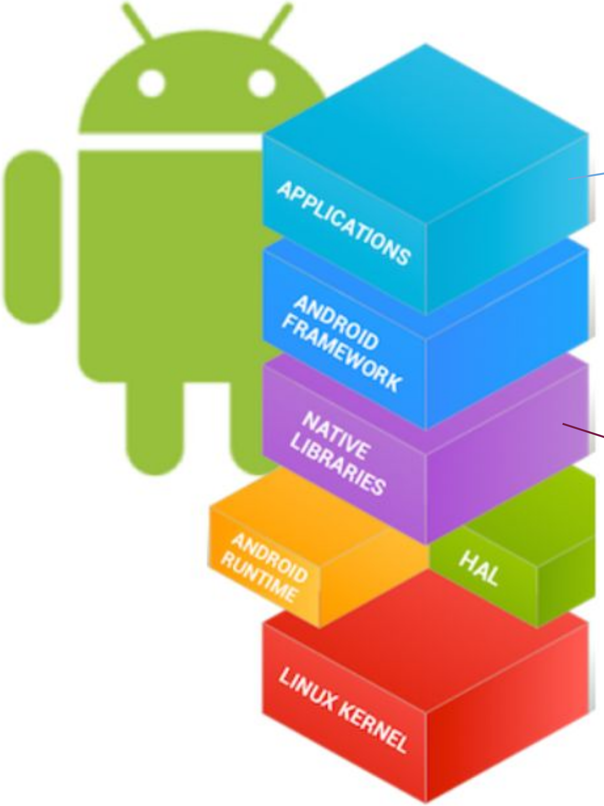
More information about GNSS Measurements, including which phones support raw measurements, see: https://g.co/GnssTools

For Matlab code for processing GPS measurements see the `opensource` folder in the GPS Measurement Tools repository on GitHub.

Overview

1. Raw GNSS Measurements
2. Logging Tools
3. How to get Pseudorange
4. Analysis Tools
5. Hands-on Exercises
6. Future: Apps and Research

Location APIs, Measurement APIs



aka *Google Play Services* aka *Google Mobile Service*
Most Android phones have this (not China)

Location APIs, android.gms.location

- Places
- Geofencing
- Fused Location Provider (FLP)
- Fit
- Activity Recognition
- Nearby

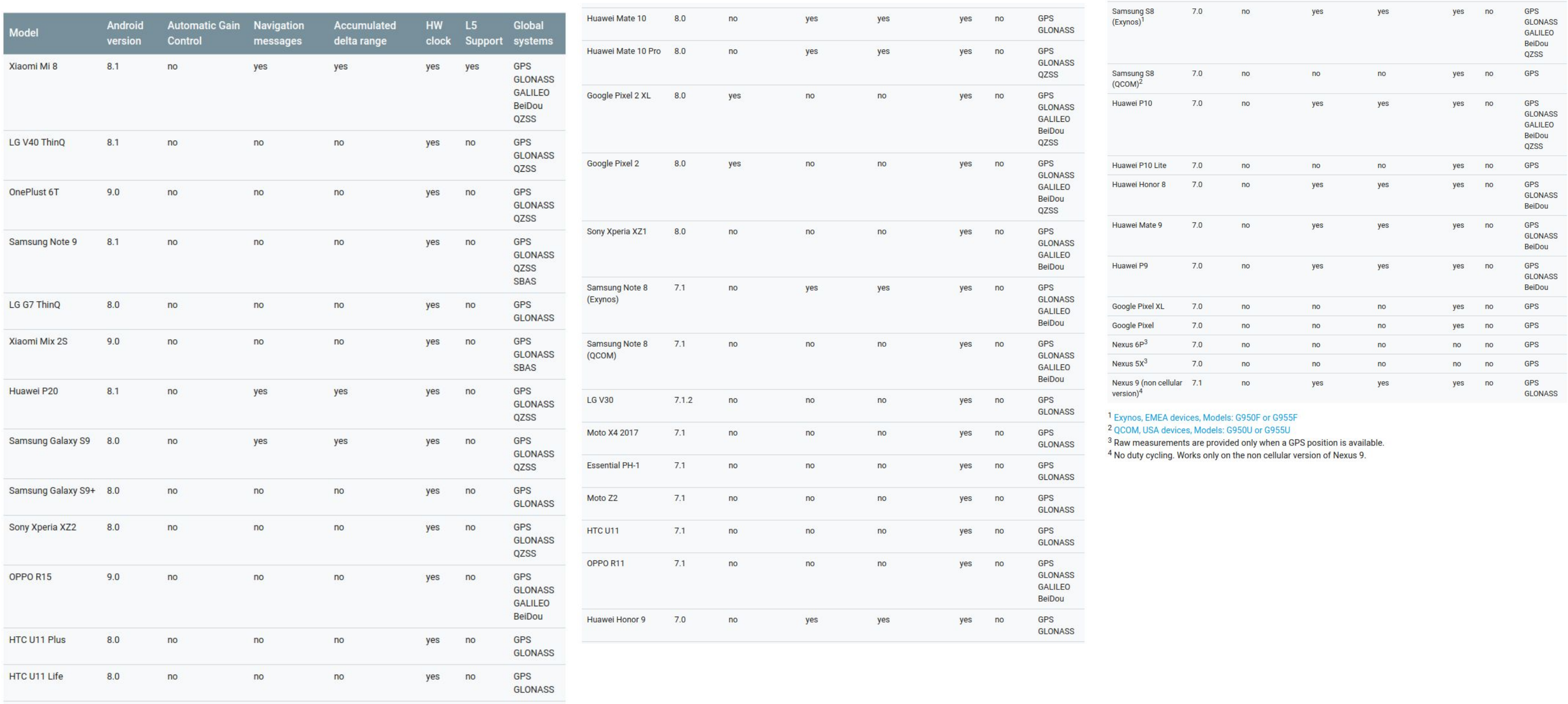
All Android phones have this

Measurement/Sensor APIs, in android.location

- Location
- GnssMeasurement
- GnssClock

GNSS Raw Measurements
All phones with:
GNSS chips build date ≥ 2016
OS ≥ Android N (Nougat)

https://g.co/GnssTools, which phones have GNSS Raw Measurements:



Model	Android version	Automatic Gain Control	Navigation messages	Accumulated delta range	HW clock	L5 Support	Global systems
Xiaomi Mi 8	8.1	no	yes	yes	yes	yes	GPS GLONASS GALILEO BeiDou QZSS
LG V40 ThinQ	8.1	no	no	no	yes	no	GPS GLONASS QZSS
OnePlust 6T	9.0	no	no	no	yes	no	GPS GLONASS QZSS
Samsung Note 9	8.1	no	no	no	yes	no	GPS GLONASS QZSS SBAS
LG G7 ThinQ	8.0	no	no	no	yes	no	GPS GLONASS
Xiaomi Mix 2S	9.0	no	no	no	yes	no	GPS GLONASS SBAS
Huawei P20	8.1	no	yes	yes	yes	no	GPS GLONASS QZSS
Samsung Galaxy S9	8.0	no	yes	yes	yes	no	GPS GLONASS QZSS
Samsung Galaxy S9+	8.0	no	no	no	yes	no	GPS GLONASS
Sony Xperia XZ2	8.0	no	no	no	yes	no	GPS GLONASS QZSS
OPPO R15	9.0	no	no	no	yes	no	GPS GLONASS GALILEO BeiDou
HTC U11 Plus	8.0	no	no	no	yes	no	GPS GLONASS
HTC U11 Life	8.0	no	no	no	yes	no	GPS GLONASS
Huawei Mate 10	8.0	no	yes	yes	yes	no	GPS GLONASS
Huawei Mate 10 Pro	8.0	no	yes	yes	yes	no	GPS GLONASS QZSS
Google Pixel 2 XL	8.0	yes	no	no	yes	no	GPS GLONASS GALILEO BeiDou QZSS
Google Pixel 2	8.0	yes	no	no	yes	no	GPS GLONASS GALILEO BeiDou QZSS
Sony Xperia XZ1	8.0	no	no	no	yes	no	GPS GLONASS GALILEO BeiDou
Samsung Note 8 (Exynos)	7.1	no	yes	yes	yes	no	GPS GLONASS GALILEO BeiDou
Samsung Note 8 (QCOM)	7.1	no	no	no	yes	no	GPS GLONASS GALILEO BeiDou
LG V30	7.1.2	no	no	no	yes	no	GPS GLONASS
Moto X4 2017	7.1	no	no	no	yes	no	GPS GLONASS
Essential PH-1	7.1	no	no	no	yes	no	GPS GLONASS
Moto Z2	7.1	no	no	no	yes	no	GPS GLONASS
HTC U11	7.1	no	no	no	yes	no	GPS GLONASS
OPPO R11	7.1	no	no	no	yes	no	GPS GLONASS GALILEO BeiDou
Huawei Honor 9	7.0	no	yes	yes	yes	no	GPS GLONASS
Samsung S8 (Exynos)[1]	7.0	no	yes	yes	yes	no	GPS GLONASS GALILEO BeiDou QZSS
Samsung S8 (QCOM)[2]	7.0	no	no	no	yes	no	GPS
Huawei P10	7.0	no	yes	yes	yes	no	GPS GLONASS GALILEO BeiDou QZSS
Huawei P10 Lite	7.0	no	no	no	yes	no	GPS
Huawei Honor 8	7.0	no	yes	yes	yes	no	GPS GLONASS BeiDou
Huawei Mate 9	7.0	no	yes	yes	yes	no	GPS GLONASS BeiDou
Huawei P9	7.0	no	yes	yes	yes	no	GPS GLONASS BeiDou
Google Pixel XL	7.0	no	no	no	yes	no	GPS
Google Pixel	7.0	no	no	no	yes	no	GPS
Nexus 6P[3]	7.0	no	no	no	no	no	GPS
Nexus 5X[3]	7.0	no	no	no	no	no	GPS
Nexus 9 (non cellular version)[4]	7.1	no	yes	yes	yes	no	GPS GLONASS

[1] Exynos, EMEA devices, Models: G950F or G955F

[2] QCOM, USA devices, Models: G950U or G955U

[3] Raw measurements are provided only when a GPS position is available.

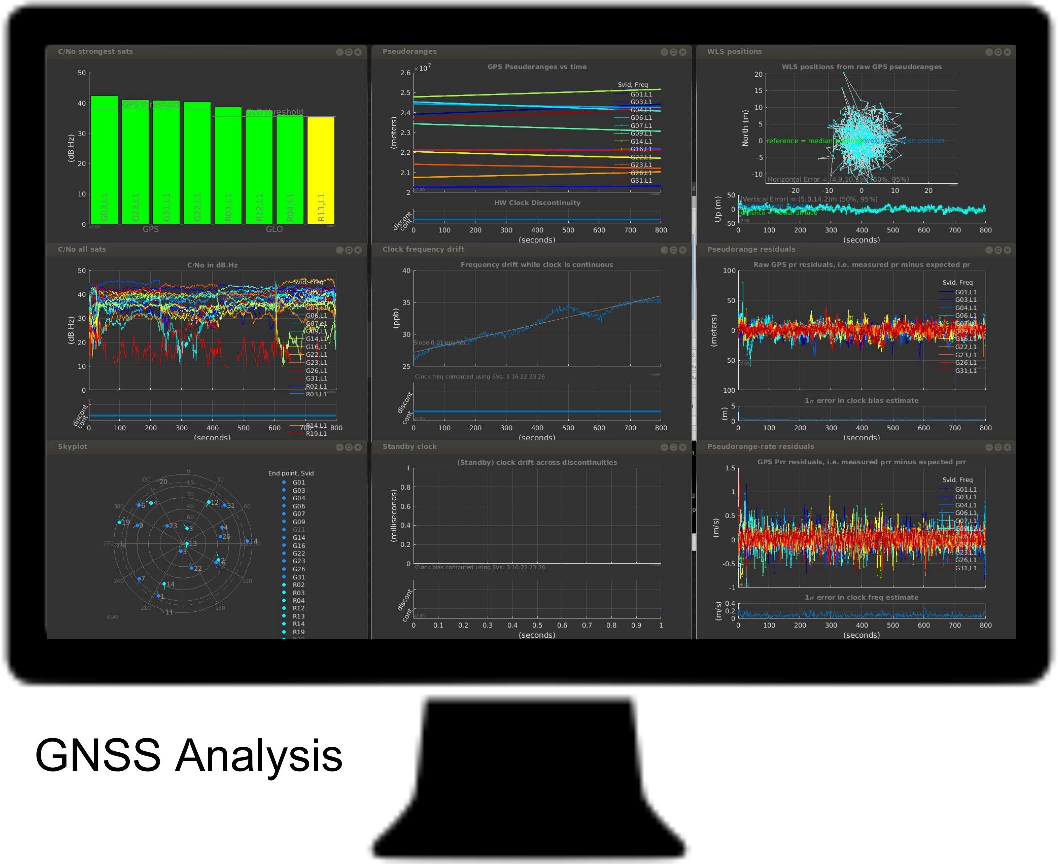
[4] No duty cycling. Works only on the non cellular version of Nexus 9.

If you know of a phone that supports raw measurements, and it is not in the online table please tell us by clicking the feedback link on this page: https://g.co/GnssTools

Overview

1. Raw GNSS Measurements
2. Logging Tools
3. How to get Pseudorange
4. Analysis Tools
5. Hands-on Exercises
6. Future: Apps and Research

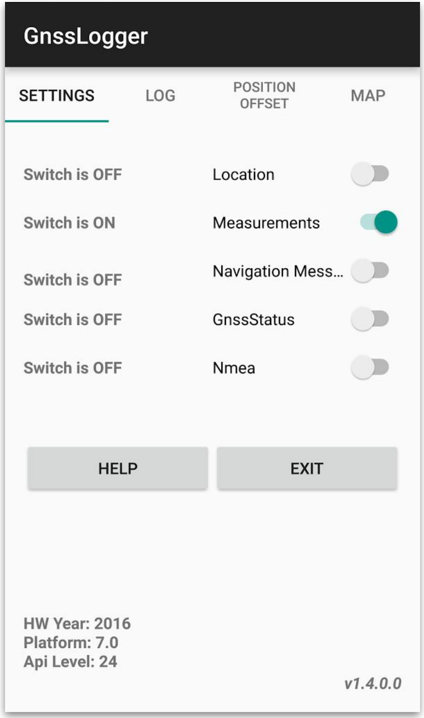
GNSS Logger


C/No strongest sats
Pseudoranges
GPS Pseudoranges vs time
HW Clock Discontinuity
WLS positions
WLS positions from raw GPS pseudoranges
C/No all sats
C/No in dB.Hz
Clock frequency drift
Frequency drift while clock is continuous
Pseudorange residuals
Raw GPS pr residuals, i.e. measured pr minus expected pr
1σ error in clock bias estimate
Skyplot
Standby clock
(Standby) clock drift across discontinuities
Pseudorange-rate residuals
GPS Prr residuals, i.e. measured prr minus expected prr
1σ error in clock freq estimate
(seconds)

GNSS Analysis

Logging the raw data on your phone:

1,



GnssLogger

SETTINGS LOG POSITION OFFSET MAP

Switch is OFF Location

Switch is ON Measurements

Switch is OFF Navigation Mess...

Switch is OFF GnssStatus

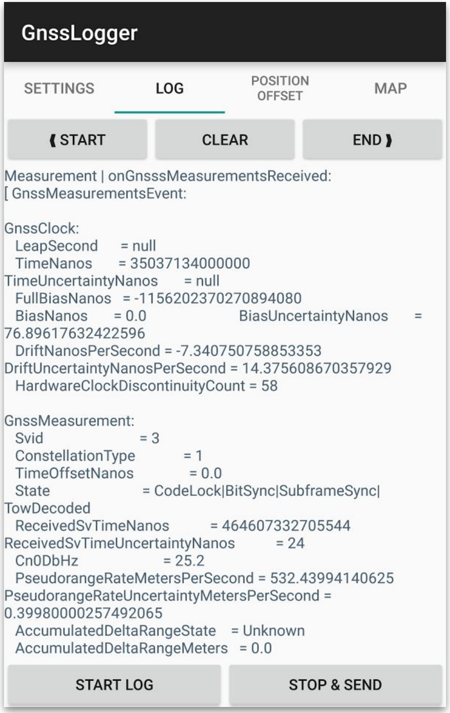
Switch is OFF Nmea

HELP EXIT

HW Year: 2016
Platform: 7.0
Api Level: 24

v1.4.0.0

2,



GnssLogger

SETTINGS LOG POSITION OFFSET MAP

❰ START CLEAR END ❱

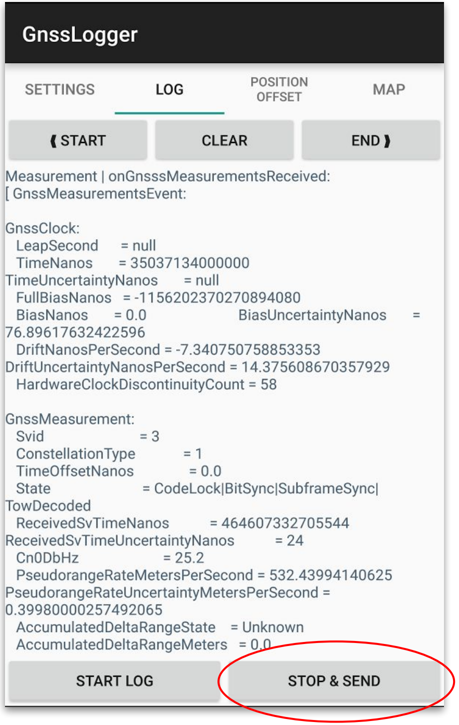
```
Measurement | onGnsssMeasurementsReceived:
[ GnssMeasurementsEvent:

GnssClock:
  LeapSecond     = null
  TimeNanos      = 35037134000000
TimeUncertaintyNanos      = null
  FullBiasNanos   = -1156202370270894080
  BiasNanos      = 0.0                  BiasUncertaintyNanos      =
76.89617632422596
  DriftNanosPerSecond = -7.340750758853353
DriftUncertaintyNanosPerSecond = 14.375608670357929
  HardwareClockDiscontinuityCount = 58

GnssMeasurement:
  Svid                        = 3
  ConstellationType           = 1
  TimeOffsetNanos             = 0.0
  State                       = CodeLock|BitSync|SubframeSync|
TowDecoded
  ReceivedSvTimeNanos         = 464607332705544
ReceivedSvTimeUncertaintyNanos         = 24
  Cn0DbHz                     = 25.2
  PseudorangeRateMetersPerSecond = 532.43994140625
PseudorangeRateUncertaintyMetersPerSecond =
0.39980000257492065
  AccumulatedDeltaRangeState    = Unknown
  AccumulatedDeltaRangeMeters   = 0.0
```

START LOG STOP & SEND

3,



GnssLogger

SETTINGS LOG POSITION OFFSET MAP

❰ START CLEAR END ❱

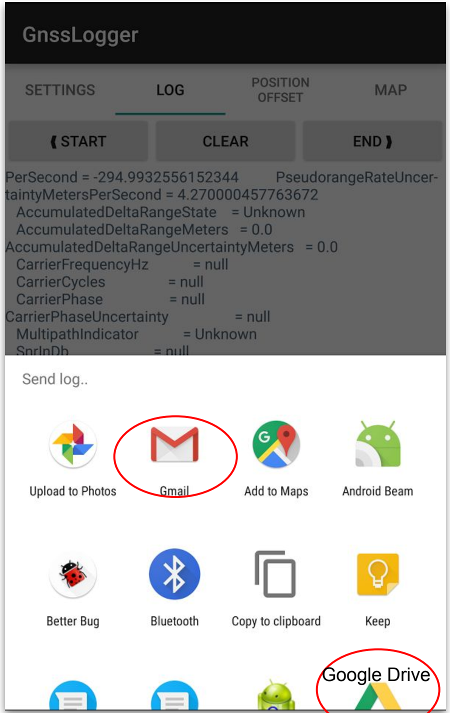
```
Measurement | onGnsssMeasurementsReceived:
[ GnssMeasurementsEvent:

GnssClock:
  LeapSecond     = null
  TimeNanos      = 35037134000000
TimeUncertaintyNanos      = null
  FullBiasNanos   = -1156202370270894080
  BiasNanos      = 0.0                  BiasUncertaintyNanos      =
76.89617632422596
  DriftNanosPerSecond = -7.340750758853353
DriftUncertaintyNanosPerSecond = 14.375608670357929
  HardwareClockDiscontinuityCount = 58

GnssMeasurement:
  Svid                        = 3
  ConstellationType           = 1
  TimeOffsetNanos             = 0.0
  State                       = CodeLock|BitSync|SubframeSync|
TowDecoded
  ReceivedSvTimeNanos         = 464607332705544
ReceivedSvTimeUncertaintyNanos         = 24
  Cn0DbHz                     = 25.2
  PseudorangeRateMetersPerSecond = 532.43994140625
PseudorangeRateUncertaintyMetersPerSecond =
0.39980000257492065
  AccumulatedDeltaRangeState    = Unknown
  AccumulatedDeltaRangeMeters   = 0.0
```

START LOG STOP & SEND

4.



GnssLogger

SETTINGS LOG POSITION OFFSET MAP

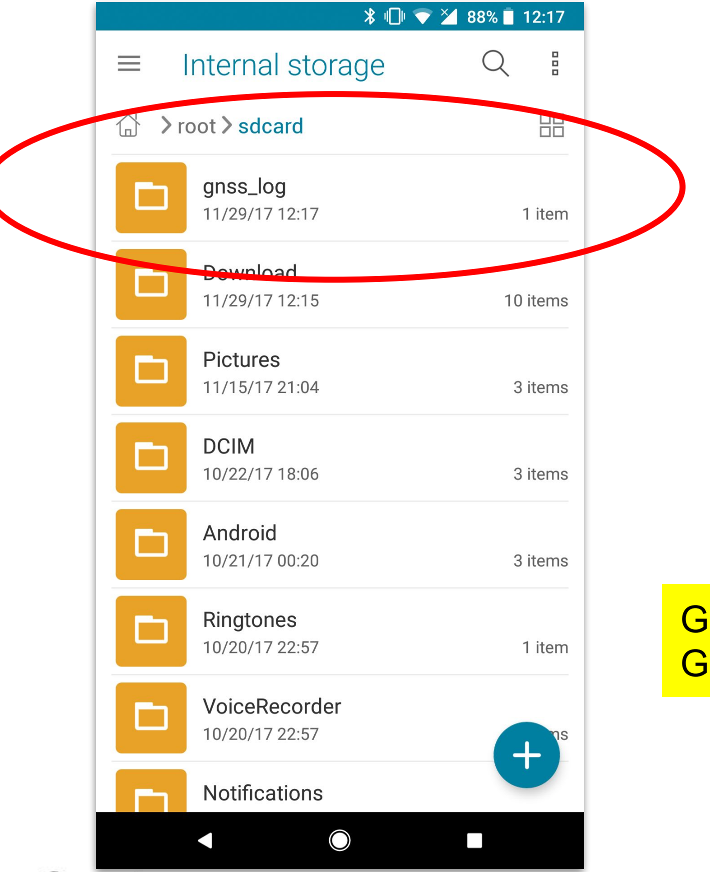
❰ START CLEAR END ❱

```
PerSecond = -294.9932556152344       PseudorangeRateUncer-
taintyMetersPerSecond = 4.270000457763672
  AccumulatedDeltaRangeState    = Unknown
  AccumulatedDeltaRangeMeters   = 0.0
AccumulatedDeltaRangeUncertaintyMeters   = 0.0
  CarrierFrequencyHz          = null
  CarrierCycles               = null
  CarrierPhase                = null
CarrierPhaseUncertainty                = null
  MultipathIndicator          = Unknown
  SnrInDb                     = null
```

Send log..

Google Drive

Logged Data is stored locally, on the phone:



88% 12:17

Internal storage

root > sdcard

gnss_log
11/29/17 12:17 1 item

Download
11/29/17 12:15 10 items

Pictures
11/15/17 21:04 3 items

DCIM
10/22/17 18:06 3 items

Android
10/21/17 00:20 3 items

Ringtones
10/20/17 22:57 1 item

VoiceRecorder
10/20/17 22:57

Notifications

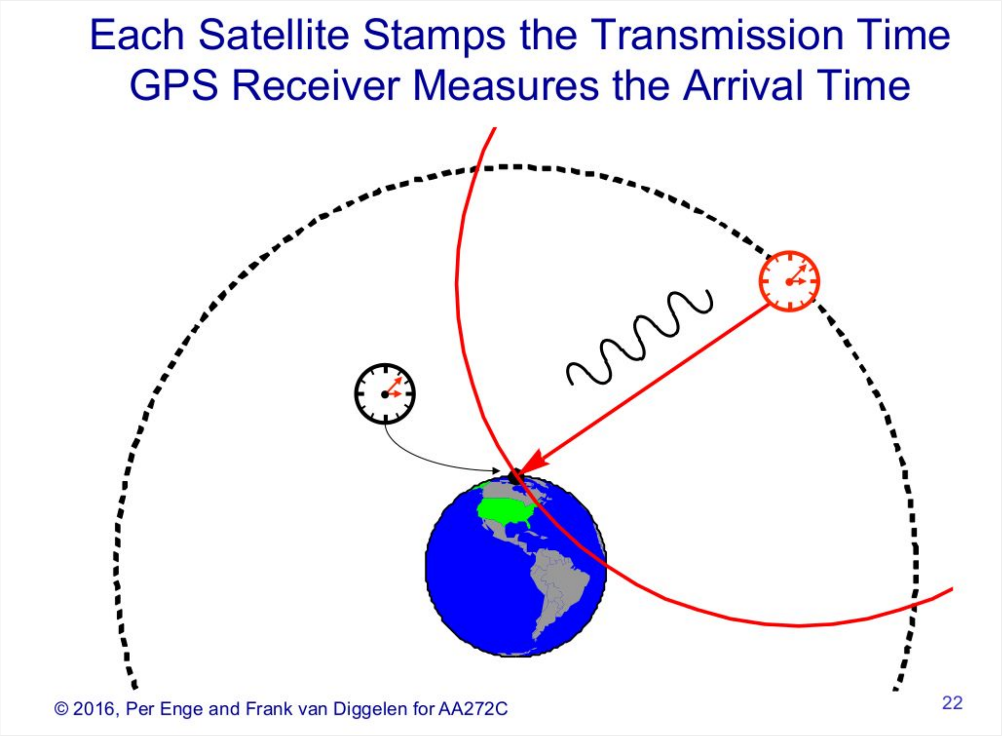
GNSS Logger data is stored locally on your phone.
Google doesn’t get any data from these log files.

Overview

1. Raw GNSS Measurements
2. Logging Tools
3. How to get Pseudorange
4. Analysis Tools
5. Hands-on Exercises
6. Future: Apps and Research

Most important takeaway:
Understand tRx and tTx

A reminder of what we mean by “pseudorange”



Each Satellite Stamps the Transmission Time
GPS Receiver Measures the Arrival Time

© 2016, Per Enge and Frank van Diggelen for AA272C

22

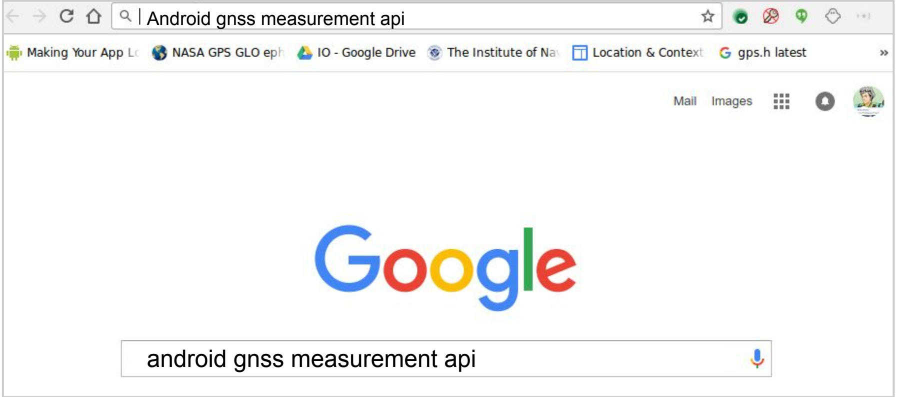
Source: Stanford University, Course AA272C “Introduction to GPS”, Per Enge & Frank van Diggelen

Pseudorange (in units of time)
= Arrival Time - Transmission Time

Pseudorange (in units of distance)
= (Arrival Time - Transmission Time)*c

To find the GNSS Measurement APIs ...

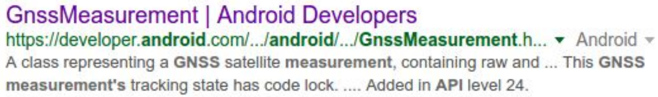
1)



Android gnss measurement api

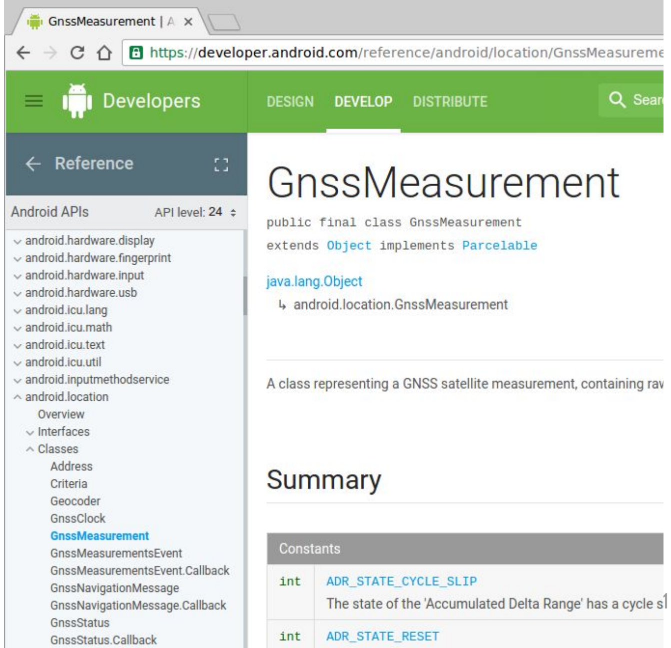
android gnss measurement api

2)



GnssMeasurement | Android Developers
https://developer.android.com/.../android/.../GnssMeasurement.h... ▾ Android ▾
A class representing a **GNSS** satellite **measurement**, containing raw and ... This **GNSS measurement's** tracking state has code lock. .... Added in **API** level 24.

3)



https://developer.android.com/reference/android/location/GnssMeasureme

GnssMeasurement

```
public final class GnssMeasurement
extends Object implements Parcelable
```

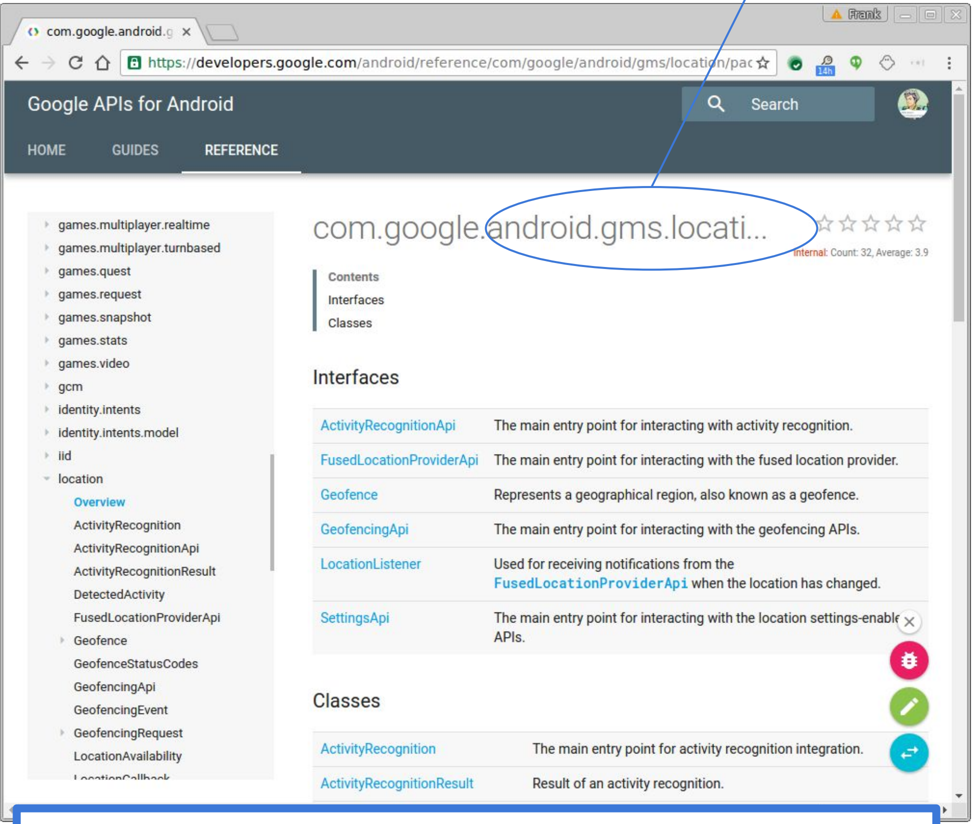
java.lang.Object
↳ android.location.GnssMeasurement

A class representing a GNSS satellite measurement, containing rav

Summary

Constants	
int	ADR_STATE_CYCLE_SLIP The state of the 'Accumulated Delta Range' has a cycle sl
int	ADR_STATE_RESET

In Section 1 we saw the difference between this, and this.



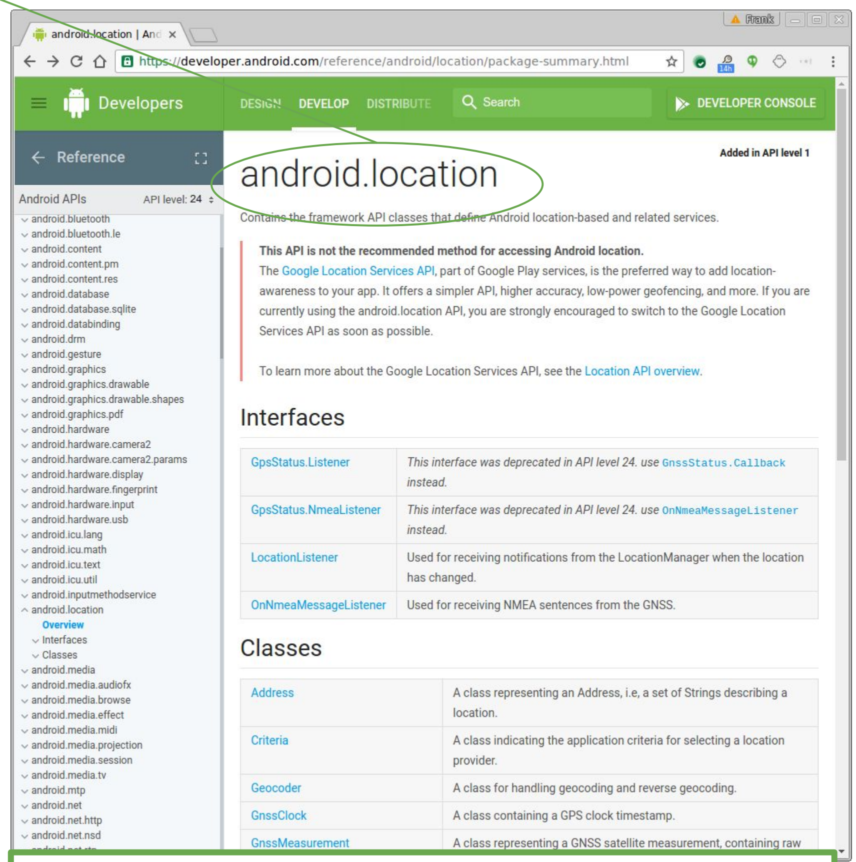
com.google.android.gms.locati...

Interfaces

ActivityRecognitionApi	The main entry point for interacting with activity recognition.
FusedLocationProviderApi	The main entry point for interacting with the fused location provider.
Geofence	Represents a geographical region, also known as a geofence.
GeofencingApi	The main entry point for interacting with the geofencing APIs.
LocationListener	Used for receiving notifications from the `FusedLocationProviderApi` when the location has changed.
SettingsApi	The main entry point for interacting with the location settings-enable APIs.

Classes

ActivityRecognition	The main entry point for activity recognition integration.
ActivityRecognitionResult	Result of an activity recognition.



android.location

Contains the framework API classes that define Android location-based and related services.

This API is not the recommended method for accessing Android location.
The Google Location Services API, part of Google Play services, is the preferred way to add location-awareness to your app. It offers a simpler API, higher accuracy, low-power geofencing, and more. If you are currently using the android.location API, you are strongly encouraged to switch to the Google Location Services API as soon as possible.

To learn more about the Google Location Services API, see the Location API overview.

Interfaces

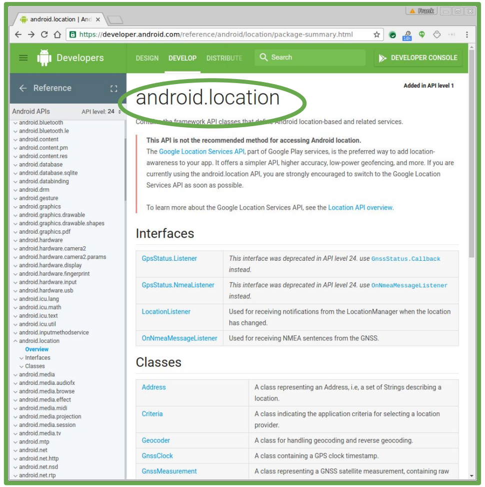
GpsStatus.Listener	*This interface was deprecated in API level 24. use* `GnssStatus.Callback` *instead.*
GpsStatus.NmeaListener	*This interface was deprecated in API level 24. use* `OnNmeaMessageListener` *instead.*
LocationListener	Used for receiving notifications from the LocationManager when the location has changed.
OnNmeaMessageListener	Used for receiving NMEA sentences from the GNSS.

Classes

Address	A class representing an Address, i.e, a set of Strings describing a location.
Criteria	A class indicating the application criteria for selecting a location provider.
Geocoder	A class for handling geocoding and reverse geocoding.
GnssClock	A class containing a GPS clock timestamp.
GnssMeasurement	A class representing a GNSS satellite measurement, containing raw

Google Play Services, higher abstractions of location

android.location, for Raw Measurements



Added in API level 1

android.location

Contains the framework API classes that define Android location-based and related services.

> **This API is not the recommended method for accessing Android location.**
> The Google Location Services API, part of Google Play services, is the preferred way to add location-awareness to your app. It offers a simpler API, higher accuracy, low-power geofencing, and more. If you are currently using the android.location API, you are strongly encouraged to switch to the Google Location Services API as soon as possible.
>
> To learn more about the Google Location Services API, see the Location API overview.

Interfaces

GpsStatus.Listener	*This interface was deprecated in API level 24. use* `GnssStatus.Callback` *instead.*
GpsStatus.NmeaListener	*This interface was deprecated in API level 24. use* `OnNmeaMessageListener` *instead.*
LocationListener	Used for receiving notifications from the LocationManager when the location has changed.
OnNmeaMessageListener	Used for receiving NMEA sentences from the GNSS.

Classes

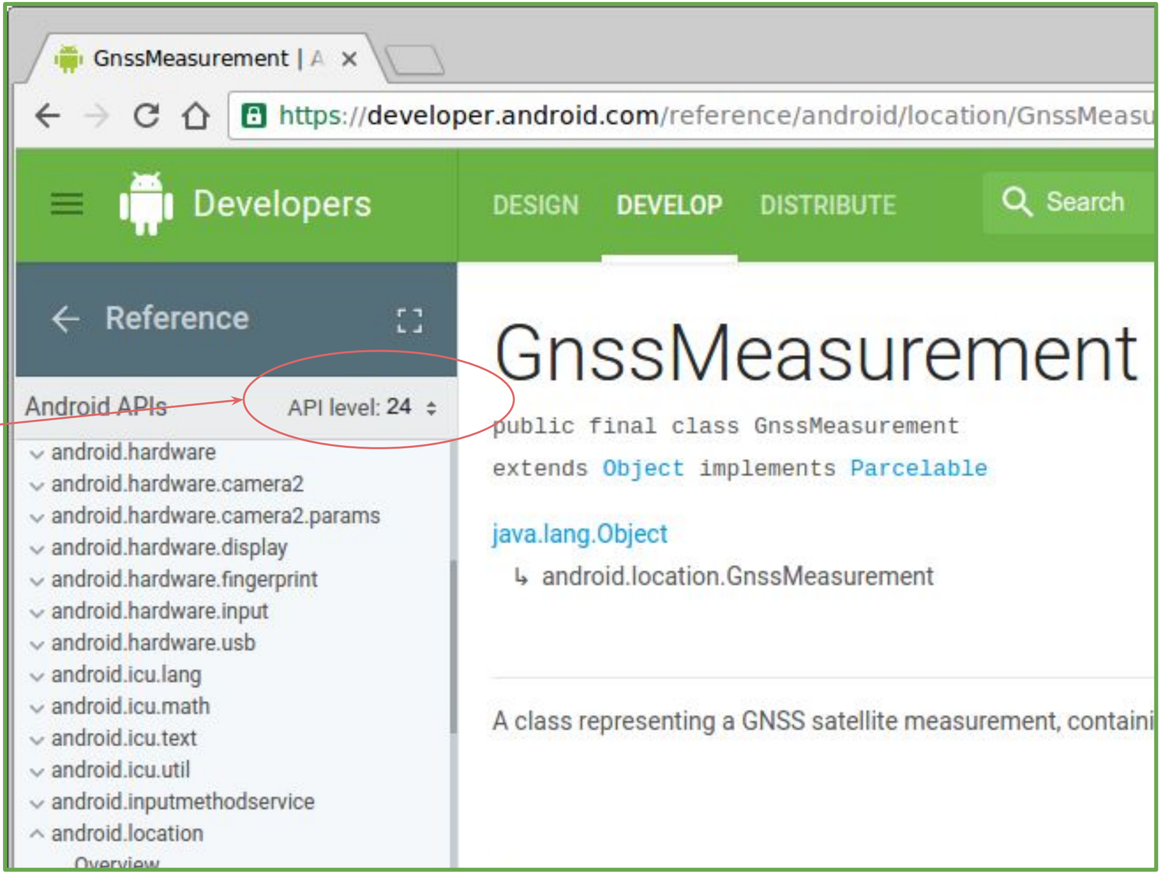
Address	A class representing an Address, i.e, a set of Strings describing a location.
Criteria	A class indicating the application criteria for selecting a location provider.
Geocoder	A class for handling geocoding and reverse geocoding.
GnssClock	A class containing a GPS clock timestamp.
GnssMeasurement	A class representing a GNSS satellite measurement, containing raw

Stylistic note for this section:

Green-outlined box,
means an extract from android.location APIs

How to make sure you're looking at Android release "N" or later

API level 24 = Release N



android.hardware
android.hardware.camera2
android.hardware.camera2.params
android.hardware.display
android.hardware.fingerprint
android.hardware.input
android.hardware.usb
android.icu.lang
android.icu.math
android.icu.text
android.icu.util
android.inputmethodservice
android.location
Overview

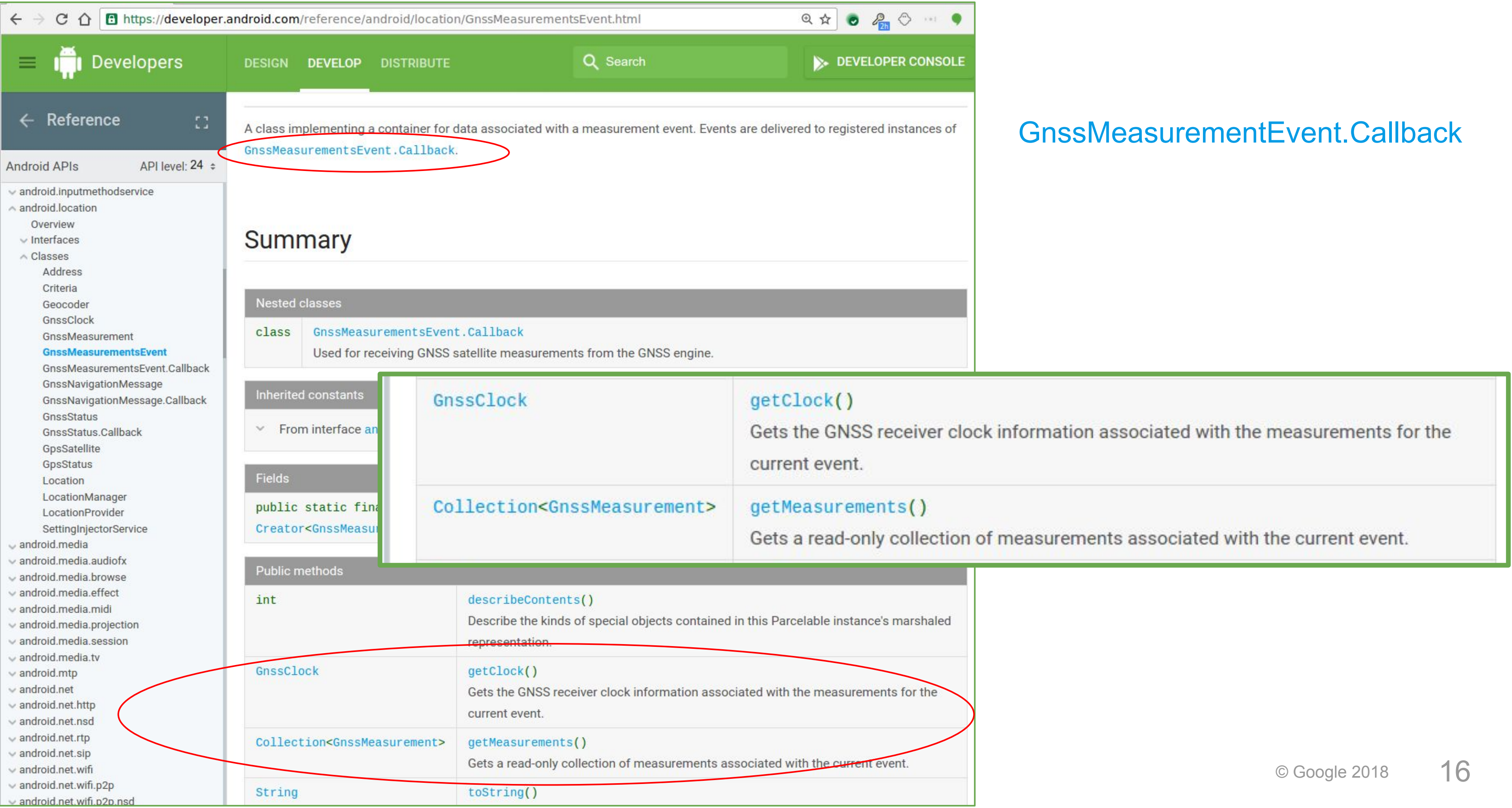
GnssMeasurement

```
public final class GnssMeasurement
extends Object implements Parcelable
```

java.lang.Object
↳ android.location.GnssMeasurement

A class representing a GNSS satellite measurement, containi

GnssMeasurementEvent.Callback



A class implementing a container for data associated with a measurement event. Events are delivered to registered instances of `GnssMeasurementsEvent.Callback`.

Summary

Nested classes	
class	`GnssMeasurementsEvent.Callback` Used for receiving GNSS satellite measurements from the GNSS engine.

Inherited constants

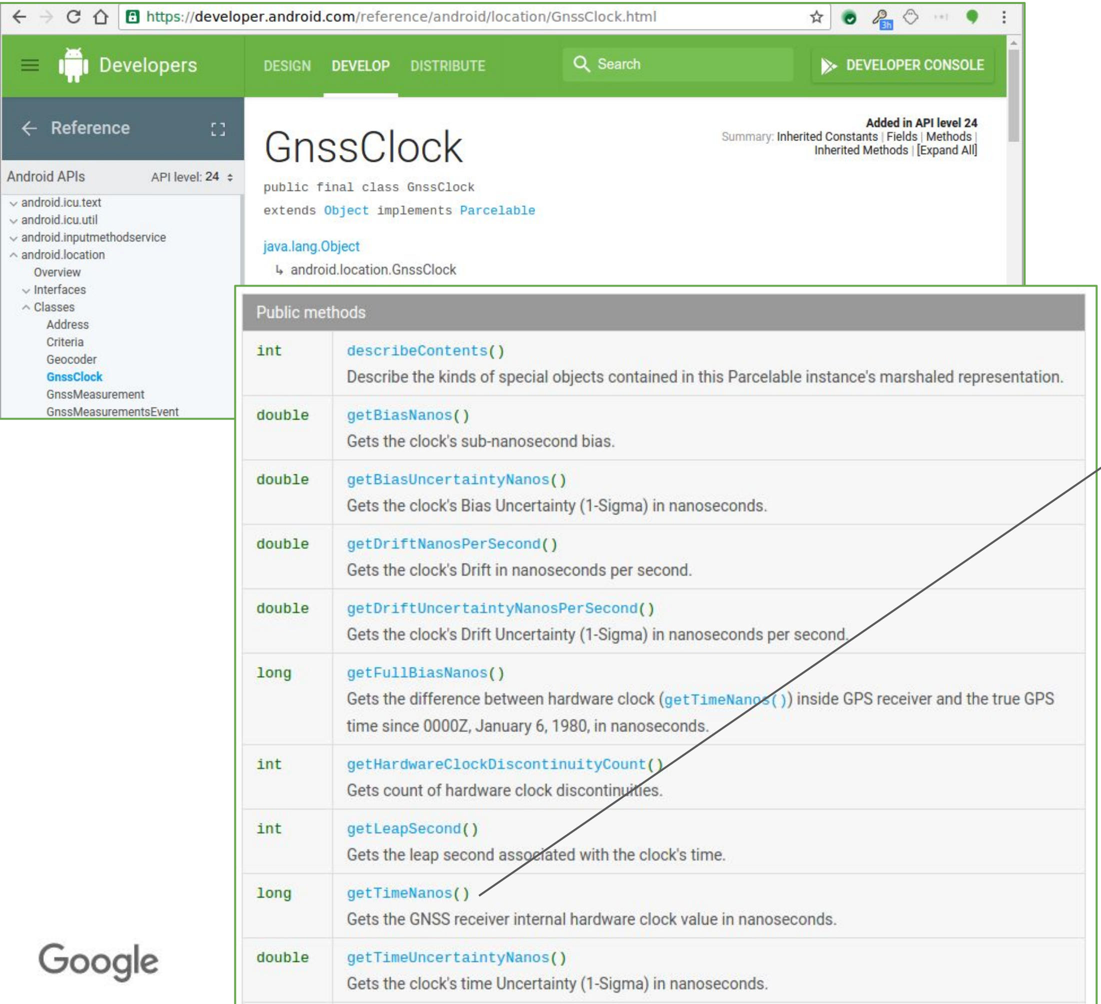
From interface an

Fields

```
public static fina
Creator<GnssMeasu
```

Public methods	
int	`describeContents()` Describe the kinds of special objects contained in this Parcelable instance's marshaled representation.
GnssClock	`getClock()` Gets the GNSS receiver clock information associated with the measurements for the current event.
Collection<GnssMeasurement>	`getMeasurements()` Gets a read-only collection of measurements associated with the current event.
String	`toString()`

GnssClock	`getClock()` Gets the GNSS receiver clock information associated with the measurements for the current event.
Collection<GnssMeasurement>	`getMeasurements()` Gets a read-only collection of measurements associated with the current event.



android.icu.text
android.icu.util
android.inputmethodservice
android.location
Overview
Interfaces
Classes
Address
Criteria
Geocoder
GnssClock
GnssMeasurement
GnssMeasurementsEvent

GnssClock

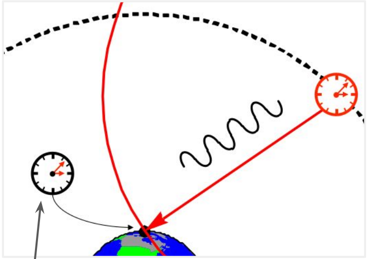
Added in API level 24

Summary: Inherited Constants | Fields | Methods | Inherited Methods | [Expand All]

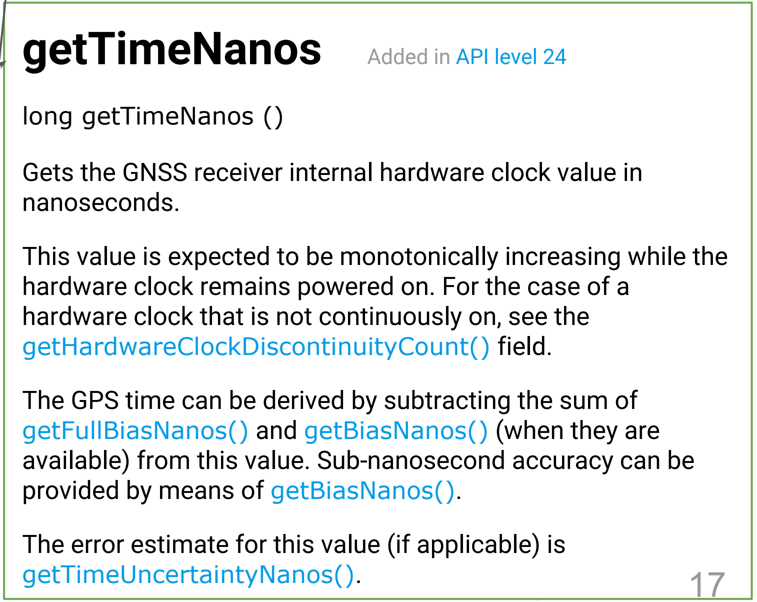
```
public final class GnssClock
extends Object implements Parcelable
```

java.lang.Object
↳ android.location.GnssClock

Public methods	
int	describeContents() Describe the kinds of special objects contained in this Parcelable instance's marshaled representation.
double	getBiasNanos() Gets the clock's sub-nanosecond bias.
double	getBiasUncertaintyNanos() Gets the clock's Bias Uncertainty (1-Sigma) in nanoseconds.
double	getDriftNanosPerSecond() Gets the clock's Drift in nanoseconds per second.
double	getDriftUncertaintyNanosPerSecond() Gets the clock's Drift Uncertainty (1-Sigma) in nanoseconds per second.
long	getFullBiasNanos() Gets the difference between hardware clock (getTimeNanos()) inside GPS receiver and the true GPS time since 0000Z, January 6, 1980, in nanoseconds.
int	getHardwareClockDiscontinuityCount() Gets count of hardware clock discontinuities.
int	getLeapSecond() Gets the leap second associated with the clock's time.
long	getTimeNanos() Gets the GNSS receiver internal hardware clock value in nanoseconds.
double	getTimeUncertaintyNanos() Gets the clock's time Uncertainty (1-Sigma) in nanoseconds.



This is the time tag of the GNSS Measurement



getTimeNanos

Added in API level 24

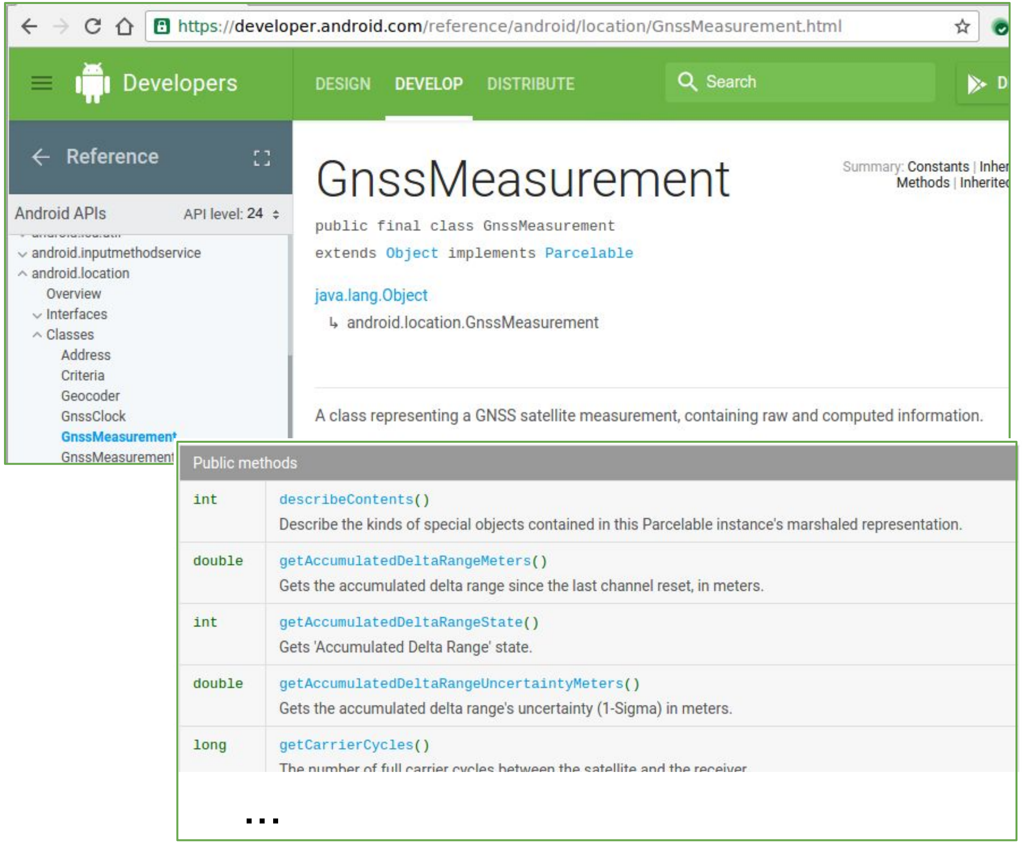
long getTimeNanos ()

Gets the GNSS receiver internal hardware clock value in nanoseconds.

This value is expected to be monotonically increasing while the hardware clock remains powered on. For the case of a hardware clock that is not continuously on, see the getHardwareClockDiscontinuityCount() field.

The GPS time can be derived by subtracting the sum of getFullBiasNanos() and getBiasNanos() (when they are available) from this value. Sub-nanosecond accuracy can be provided by means of getBiasNanos().

The error estimate for this value (if applicable) is getTimeUncertaintyNanos().



GnssMeasurement

```
public final class GnssMeasurement
extends Object implements Parcelable
```

java.lang.Object
↳ android.location.GnssMeasurement

A class representing a GNSS satellite measurement, containing raw and computed information.

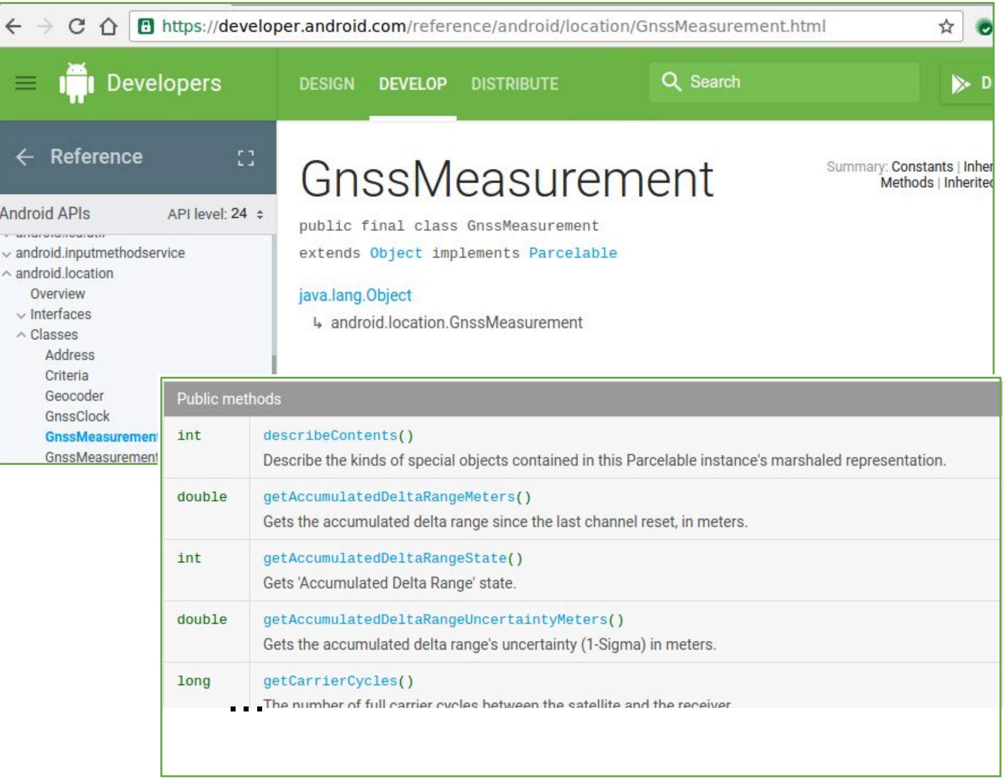
Public methods	
int	`describeContents()` Describe the kinds of special objects contained in this Parcelable instance's marshaled representation.
double	`getAccumulatedDeltaRangeMeters()` Gets the accumulated delta range since the last channel reset, in meters.
int	`getAccumulatedDeltaRangeState()` Gets 'Accumulated Delta Range' state.
double	`getAccumulatedDeltaRangeUncertaintyMeters()` Gets the accumulated delta range's uncertainty (1-Sigma) in meters.
long	`getCarrierCycles()` The number of full carrier cycles between the satellite and the receiver.

...

Public methods

- describeContents()
- getAccumulatedDeltaRangeMeters()
- getAccumulatedDeltaRangeState()
- getAccumulatedDeltaRangeUncertaintyMeters()
- getCarrierCycles()
- getCarrierFrequencyHz()
- getCarrierPhase()
- getCarrierPhaseUncertainty()
- getCn0DbHz()
- getConstellationType()
- getMultipathIndicator()
- getPseudorangeRateMetersPerSecond()
- getPseudorangeRateUncertaintyMetersPerSecond()
- getReceivedSvTimeNanos()
- getReceivedSvTimeUncertaintyNanos()
- getSnrInDb()
- getState()
- getSvid()
- getTimeOffsetNanos()
- hasCarrierCycles()
- hasCarrierFrequencyHz()
- hasCarrierPhase()
- hasCarrierPhaseUncertainty()
- hasSnrInDb()
- toString()
- writeToParcel(Parcel parcel, int flags)

So, how do you get pseudorange?



GnssMeasurement

```
public final class GnssMeasurement
extends Object implements Parcelable
```

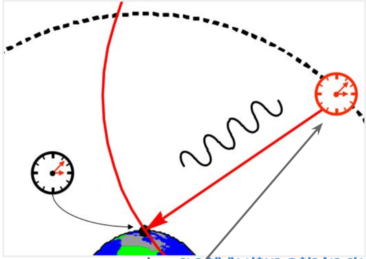
java.lang.Object
↳ android.location.GnssMeasurement

Public methods	
int	describeContents() Describe the kinds of special objects contained in this Parcelable instance's marshaled representation.
double	getAccumulatedDeltaRangeMeters() Gets the accumulated delta range since the last channel reset, in meters.
int	getAccumulatedDeltaRangeState() Gets 'Accumulated Delta Range' state.
double	getAccumulatedDeltaRangeUncertaintyMeters() Gets the accumulated delta range's uncertainty (1-Sigma) in meters.
long	getCarrierCycles() The number of full carrier cycles between the satellite and the receiver.

...

Public methods

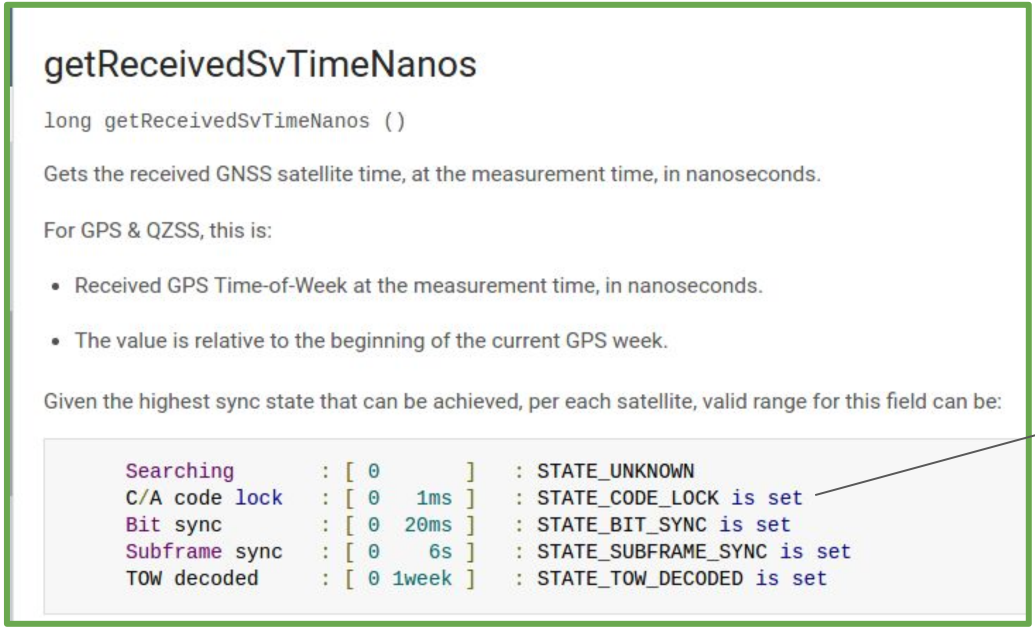
describeContents()
getAccumulatedDeltaRangeMeters()
getAccumulatedDeltaRangeState()
DeltaRangeUncertaintyMeters()
s()
encyHz()
e()
Uncertainty()
Type()
getMultipathIndicator()
getPseudorangeRateMetersPerSecond()
getPseudorangeRateUncertaintyMetersPerSecond()
getReceivedSvTimeNanos()
getReceivedSvTimeUncertaintyNanos()
getSnrInDb()
getState()
getSvid()
getTimeOffsetNanos()
hasCarrierCycles()
hasCarrierFrequencyHz()
hasCarrierPhase()
hasCarrierPhaseUncertainty()
hasSnrInDb()
toString()
writeToParcel(Parcel parcel, int flags)



You get ReceivedSvTimeNanos,
and from that you make the pseudorange

Why isn’t pseudorange provided explicitly?

Because about half of all GNSS location on smartphones happens *before* time (TOW) is fully known.



getReceivedSvTimeNanos

```
long getReceivedSvTimeNanos ()
```

Gets the received GNSS satellite time, at the measurement time, in nanoseconds.

For GPS & QZSS, this is:

- Received GPS Time-of-Week at the measurement time, in nanoseconds.
- The value is relative to the beginning of the current GPS week.

Given the highest sync state that can be achieved, per each satellite, valid range for this field can be:

```
Searching       : [ 0       ]   : STATE_UNKNOWN
C/A code lock   : [ 0   1ms ]   : STATE_CODE_LOCK is set
Bit sync        : [ 0  20ms ]   : STATE_BIT_SYNC is set
Subframe sync   : [ 0    6s ]   : STATE_SUBFRAME_SYNC is set
TOW decoded     : [ 0 1week ]   : STATE_TOW_DECODED is set
```

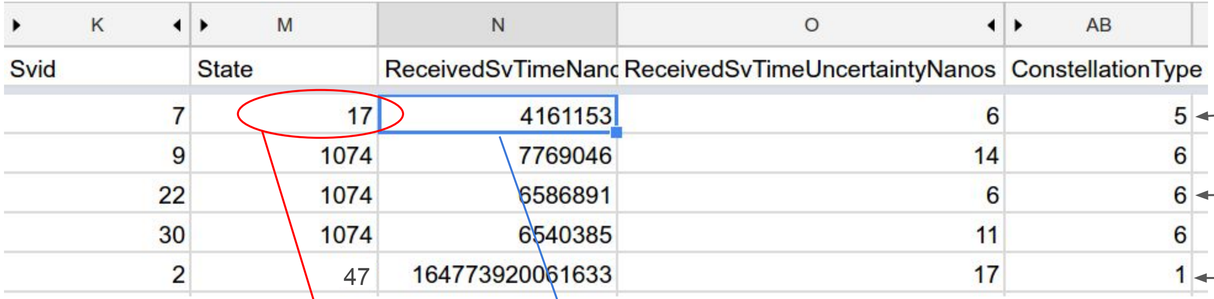
Smartphone GNSSes make extensive use of measurements long before TOW is decoded. That is how they get TTFF of 1 to 2 seconds [(1)]

These measurements are considered invalid in traditional GNSS. So you can use Android raw measurements to create RTCM and RINEX format log files, but not vice-versa without losing information.

For example: when only this bit is set, ReceivedSvTimeNanos is a value from zero to one millisecond.

[1] See “*A-GPS*” Book, van Diggelen, Chapter 4 “Coarse Time Navigation: Instant GPS”

Examples of SvTime < 20ms



K	M	N	O	AB
Svid	State	ReceivedSvTimeNanc	ReceivedSvTimeUncertaintyNanos	ConstellationType
7	17	4161153	6	5
9	1074	7769046	14	6
22	1074	6586891	6	6
30	1074	6540385	11	6
2	47	164773920061633	17	1

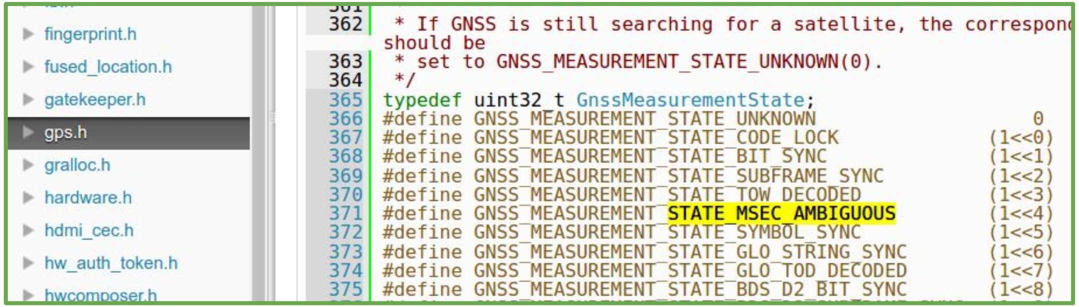
BeiDou

Galileo

GPS

4,161,153 ns = 4.1 ms

State = 17 = 0001 0001



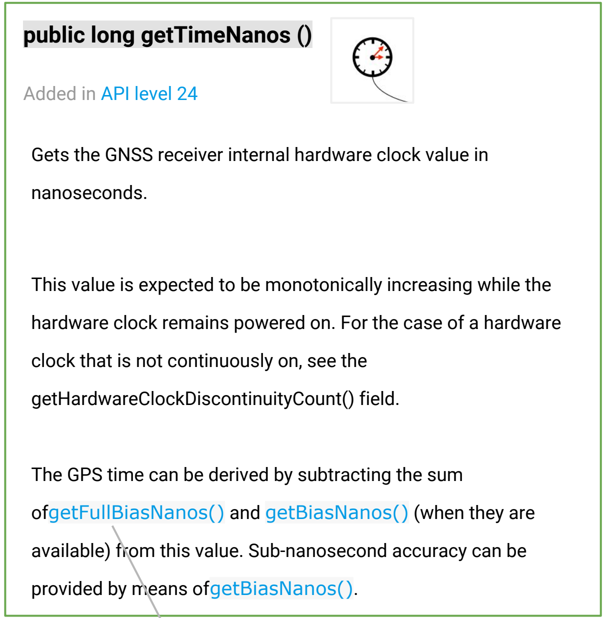
```
 * If GNSS is still searching for a satellite, the correspond
should be
 * set to GNSS_MEASUREMENT_STATE_UNKNOWN(0).
 */
typedef uint32_t GnssMeasurementState;
#define GNSS_MEASUREMENT_STATE_UNKNOWN                  0
#define GNSS_MEASUREMENT_STATE_CODE_LOCK             (1<<0)
#define GNSS_MEASUREMENT_STATE_BIT_SYNC              (1<<1)
#define GNSS_MEASUREMENT_STATE_SUBFRAME_SYNC         (1<<2)
#define GNSS_MEASUREMENT_STATE_TOW_DECODED           (1<<3)
#define GNSS_MEASUREMENT_STATE_MSEC_AMBIGUOUS        (1<<4)
#define GNSS_MEASUREMENT_STATE_SYMBOL_SYNC           (1<<5)
#define GNSS_MEASUREMENT_STATE_GLO_STRING_SYNC       (1<<6)
#define GNSS_MEASUREMENT_STATE_GLO_TOD_DECODED       (1<<7)
#define GNSS_MEASUREMENT_STATE_BDS_D2_BIT_SYNC       (1<<8)
```

For the rest of this short-course, we'll focus on GPS measurements where TOW is known.

Google

How to get GPS pseudorange (1):

From GnssClock



public long getTimeNanos ()

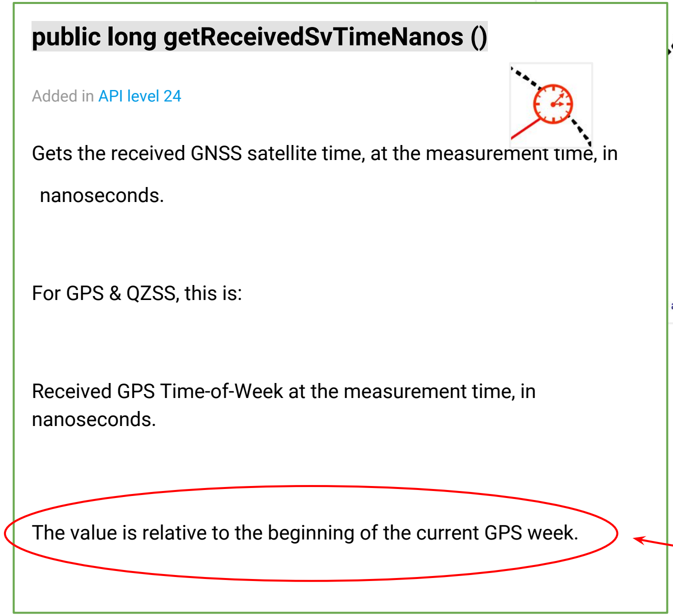
Added in API level 24

Gets the GNSS receiver internal hardware clock value in nanoseconds.

This value is expected to be monotonically increasing while the hardware clock remains powered on. For the case of a hardware clock that is not continuously on, see the getHardwareClockDiscontinuityCount() field.

The GPS time can be derived by subtracting the sum ofgetFullBiasNanos() and getBiasNanos() (when they are available) from this value. Sub-nanosecond accuracy can be provided by means ofgetBiasNanos().

From GnssMeasurement



public long getReceivedSvTimeNanos ()

Added in API level 24

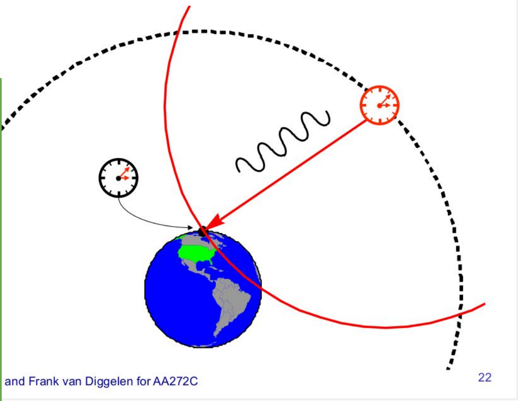
Gets the received GNSS satellite time, at the measurement time, in nanoseconds.

For GPS & QZSS, this is:

Received GPS Time-of-Week at the measurement time, in nanoseconds.

The value is relative to the beginning of the current GPS week.

Each Satellite Stamps the Transmission Time
GPS Receiver Measures the Arrival Time

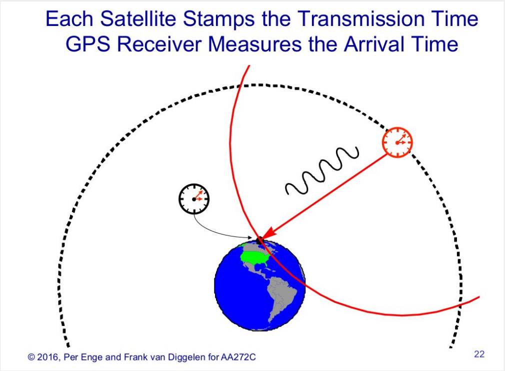


getFullBiasNanos()
Gets the difference between hardware clock (getTimeNanos()) inside GPS receiver and the true GPS time since 0000Z, January 6, 1980, in nanoseconds.

You must adjust the GnssClock value to the same time reference as GnssMeasurement

Google

How to get GPS pseudorange (2):



Each Satellite Stamps the Transmission Time
GPS Receiver Measures the Arrival Time

© 2016, Per Enge and Frank van Diggelen for AA272C

22

From GnssClock

The GPS time can be derived by subtracting the sum of getFullBiasNanos() and getBiasNanos()

$$\text{tRxNanos} = \underbrace{\underbrace{\underbrace{\text{TimeNanos}}_{\text{Referenced to start time}} - (\text{FullBiasNanos} + \text{BiasNanos})}_{\text{Referenced to GPS Epoch}} - \text{WeekNumberNanos}}_{\text{Referenced to GPS Week}}$$

From GnssMeasurement

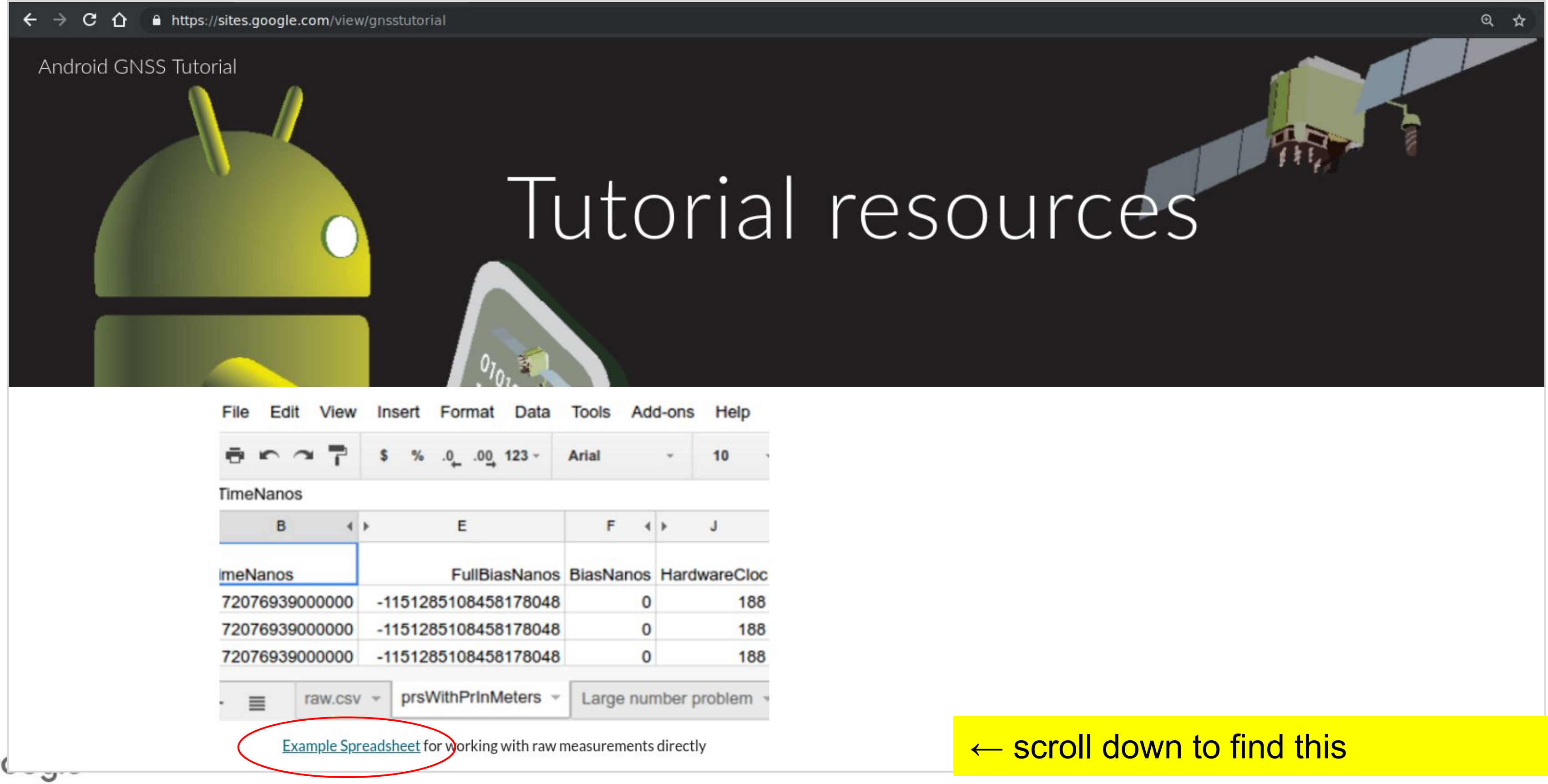
$$\text{tTxNanos} = \underbrace{\text{ReceivedSvTimeNanos}}_{\text{Referenced to GPS Week}}$$

getReceivedSvTimeNanos ()

For GPS & QZSS, this is:

- Received GPS Time-of-Week at the measurement time, in nanoseconds.
- The value is relative to the beginning of the current GPS week

Access the spreadsheet for the pseudorange exercise: https://sites.google.com/view/gnsstutorial



TimeNanos

B	E	F	J
imeNanos	FullBiasNanos	BiasNanos	HardwareCloc
72076939000000	-1151285108458178048	0	188
72076939000000	-1151285108458178048	0	188
72076939000000	-1151285108458178048	0	188

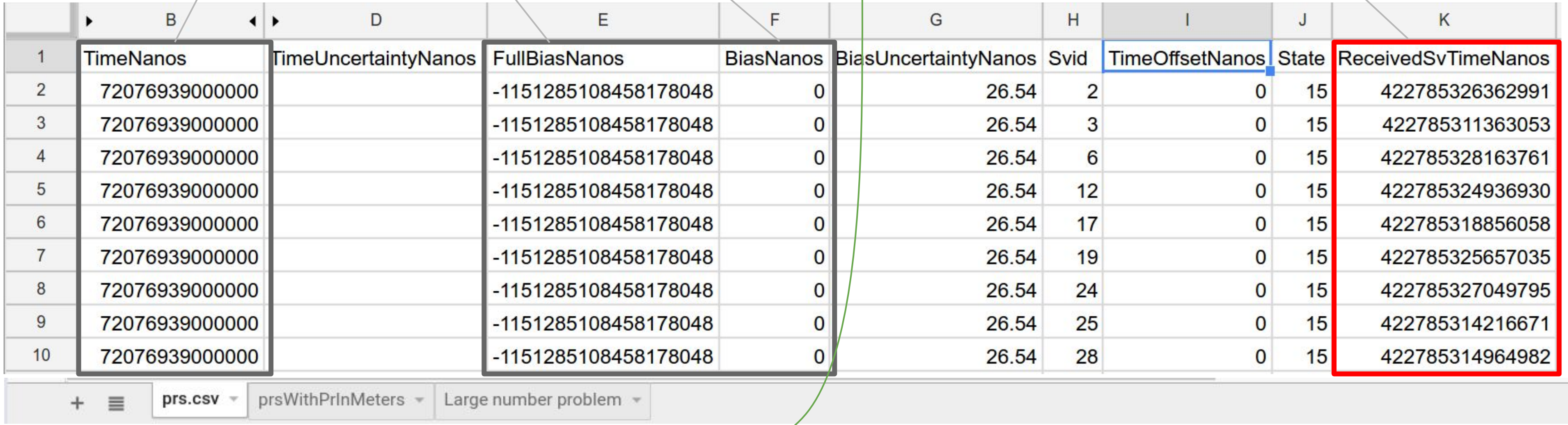
Example Spreadsheet for working with raw measurements directly

← scroll down to find this

Google Android App: GnssLogger, all these values are read to a log file.

tRxNanos = TimeNanos - (FullBiasNanos + BiasNanos) - WeekNumberNanos

tTxNanos = ReceivedSvTimeNanos



	B	D	E	F	G	H	I	J	K
1	TimeNanos	TimeUncertaintyNanos	FullBiasNanos	BiasNanos	BiasUncertaintyNanos	Svid	TimeOffsetNanos	State	ReceivedSvTimeNanos
2	72076939000000		-1151285108458178048	0	26.54	2	0	15	422785326362991
3	72076939000000		-1151285108458178048	0	26.54	3	0	15	422785311363053
4	72076939000000		-1151285108458178048	0	26.54	6	0	15	422785328163761
5	72076939000000		-1151285108458178048	0	26.54	12	0	15	422785324936930
6	72076939000000		-1151285108458178048	0	26.54	17	0	15	422785318856058
7	72076939000000		-1151285108458178048	0	26.54	19	0	15	422785325657035
8	72076939000000		-1151285108458178048	0	26.54	24	0	15	422785327049795
9	72076939000000		-1151285108458178048	0	26.54	25	0	15	422785314216671
10	72076939000000		-1151285108458178048	0	26.54	28	0	15	422785314964982

%GPS Week number:

```
weekNumber = floor(-double(gnssRaw.FullBiasNanos)*1e-9/GpsConstants.WEEKSEC);
```

Code snippet from MATLAB/gpstools/opensource/ProcessGnssMeas.m

Google Android App: GnssLogger, and one more thing: TimeOffsetNanos

double getTimeOffsetNanos ()

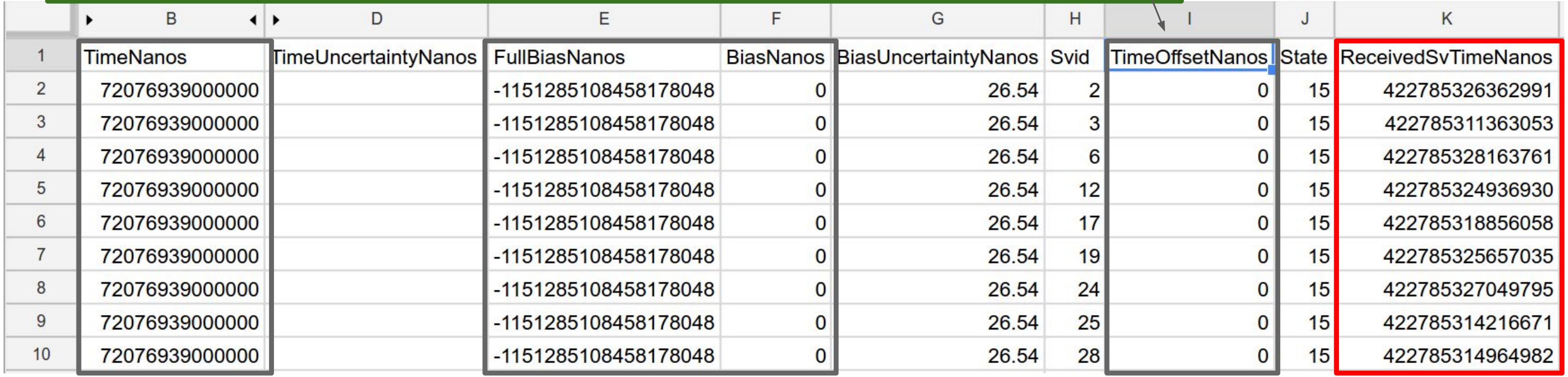
Gets the time offset at which the measurement was taken in nanoseconds.

The reference receiver's time from which this is offset is specified by getTimeNanos().

The sign of this value is given by the following equation:

measurement time = TimeNanos + TimeOffsetNanos

The value provides an individual time-stamp for the measurement, and allows sub-nanosecond accuracy.



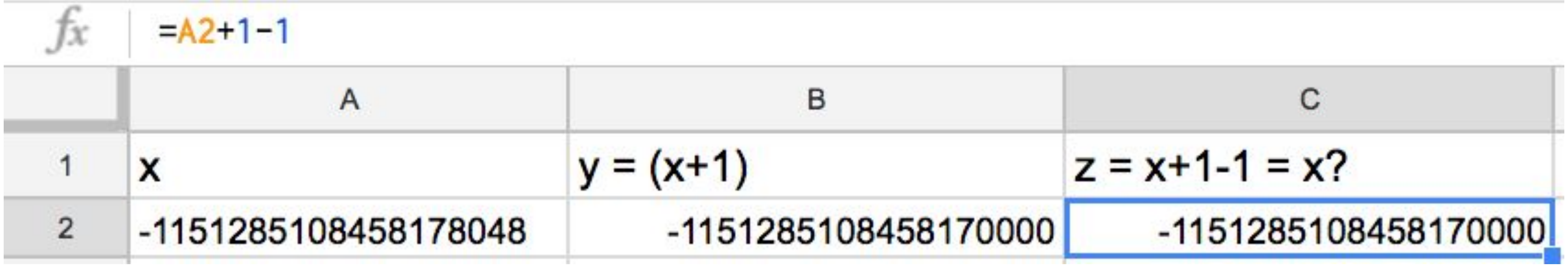
	B	D	E	F	G	H	I	J	K
1	TimeNanos	TimeUncertaintyNanos	FullBiasNanos	BiasNanos	BiasUncertaintyNanos	Svid	TimeOffsetNanos	State	ReceivedSvTimeNanos
2	72076939000000		-1151285108458178048	0	26.54	2	0	15	422785326362991
3	72076939000000		-1151285108458178048	0	26.54	3	0	15	422785311363053
4	72076939000000		-1151285108458178048	0	26.54	6	0	15	422785328163761
5	72076939000000		-1151285108458178048	0	26.54	12	0	15	422785324936930
6	72076939000000		-1151285108458178048	0	26.54	17	0	15	422785318856058
7	72076939000000		-1151285108458178048	0	26.54	19	0	15	422785325657035
8	72076939000000		-1151285108458178048	0	26.54	24	0	15	422785327049795
9	72076939000000		-1151285108458178048	0	26.54	25	0	15	422785314216671
10	72076939000000		-1151285108458178048	0	26.54	28	0	15	422785314964982

tRxNanos = TimeNanos - (FullBiasNanos + BiasNanos) - WeekNumberNanos *from previous slide*

= (TimeNanos + TimeOffsetNanos) - (FullBiasNanos + BiasNanos) - WeekNumberNanos

Making pseudoranges in the worksheet.

ARITHMETIC HEALTH WARNING:
We use worksheets for *illustration only* - because you will get precision errors of the order of 1000 ns because of the large numbers, especially FullBiasNanos



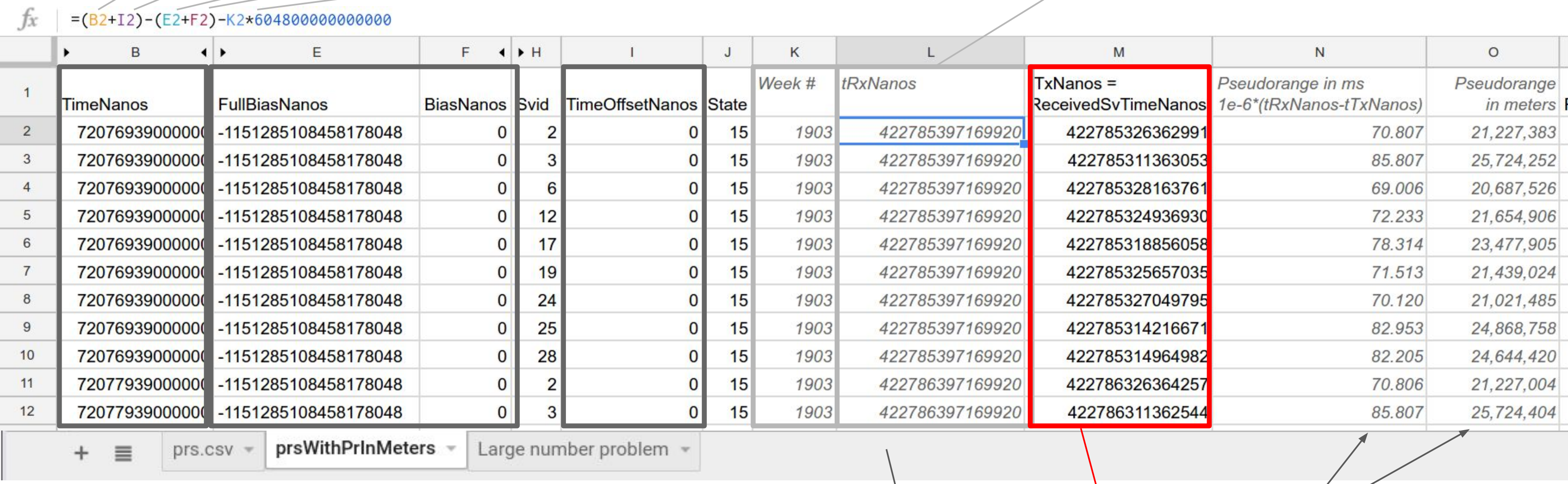
fx =A2+1-1

	A	B	C
1	x	y = (x+1)	z = x+1-1 = x?
2	-1151285108458178048	-1151285108458170000	-1151285108458170000

See sample code in later slide for how to do this without losing resolution

Making GPS pseudoranges in the worksheet *(for illustration only)*

tRxNanos = (TimeNanos + TimeOffsetNanos) - (FullBiasNanos + BiasNanos) - *WeekNumberNanos* = *tRxNanos*



```
fx  =(B2+I2)-(E2+F2)-K2*604800000000000
```

	B	E	F	H	I	J	K	L	M	N	O
1	TimeNanos	FullBiasNanos	BiasNanos	Svid	TimeOffsetNanos	State	*Week #*	*tRxNanos*	TxNanos = ReceivedSvTimeNanos	*Pseudorange in ms 1e-6*(tRxNanos-tTxNanos)*	*Pseudorange in meters*
2	72076939000000	-1151285108458178048	0	2	0	15	*1903*	*422785397169920*	422785326362991	*70.807*	*21,227,383*
3	72076939000000	-1151285108458178048	0	3	0	15	*1903*	*422785397169920*	422785311363053	*85.807*	*25,724,252*
4	72076939000000	-1151285108458178048	0	6	0	15	*1903*	*422785397169920*	422785328163761	*69.006*	*20,687,526*
5	72076939000000	-1151285108458178048	0	12	0	15	*1903*	*422785397169920*	422785324936930	*72.233*	*21,654,906*
6	72076939000000	-1151285108458178048	0	17	0	15	*1903*	*422785397169920*	422785318856058	*78.314*	*23,477,905*
7	72076939000000	-1151285108458178048	0	19	0	15	*1903*	*422785397169920*	422785325657035	*71.513*	*21,439,024*
8	72076939000000	-1151285108458178048	0	24	0	15	*1903*	*422785397169920*	422785327049795	*70.120*	*21,021,485*
9	72076939000000	-1151285108458178048	0	25	0	15	*1903*	*422785397169920*	422785314216671	*82.953*	*24,868,758*
10	72076939000000	-1151285108458178048	0	28	0	15	*1903*	*422785397169920*	422785314964982	*82.205*	*24,644,420*
11	72077939000000	-1151285108458178048	0	2	0	15	*1903*	*422786397169920*	422786326364257	*70.806*	*21,227,004*
12	72077939000000	-1151285108458178048	0	3	0	15	*1903*	*422786397169920*	422786311362544	*85.807*	*25,724,404*

tRx - tTx = *pseudorange*

Key:
Italics = derived values
Normal = read from GnssLogger

Making GPS pseudoranges in Matlab

First we get gnssRaw from: [gnssRaw] = ReadGnssLogger(dirName,prFileName);

```
%GPS Week number:
weekNumber = floor(-double(gnssRaw.FullBiasNanos)*1e-9/GpsConstants.WEEKSEC);

%compute time of measurement relative to start of week
%subtract big longs (i.e. time from 1980) before casting time of week as double
WEEKNANOS = int64(GpsConstants.WEEKSEC*1e9);
weekNumberNanos = int64(weekNumber)*int64(GpsConstants.WEEKSEC*1e9);
tRxNanos = gnssRaw.TimeNanos -gnssRaw.FullBiasNanos - weekNumberNanos;
%tRxNanos is now since the beginning of the week

%subtract the fractional offsets TimeOffsetNanos and BiasNanos:
tRxSeconds  = (double(tRxNanos)-gnssRaw.TimeOffsetNanos-gnssRaw.BiasNanos)*1e-9;
tTxSeconds  = double(gnssRaw.ReceivedSvTimeNanos)*1e-9;

prSeconds  = tRxSeconds - tTxSeconds;

PrM            = prSeconds*GpsConstants.LIGHTSPEED;
```

Note: when we deal with these numbers, subtract large integers first.

Code snippet from MATLAB/gpstools/opensource/ProcessGnssMeas.m

Open-source code

GNSS Logger apk (app) and GNSS Analysis source code is available on GitHub

- See links on https://g.co/GnssTools
- GNSS Logger is Java code
- GNSS Analysis is Matlab code
- In both cases only GPS analysis code is available as open-source
- You can submit contributions (e.g. add other GNSS open-source code).

The compiled version of GNSS Logger and GNSS Analysis are fully GNSS compatible (i.e. GPS, GLONASS, BeiDou Galileo, QZSS).

Overview

1. Raw GNSS Measurements
2. Logging Tools
3. How to get Pseudorange
4. Analysis Tools
5. Hands-on Exercises
6. Future: Apps and Research

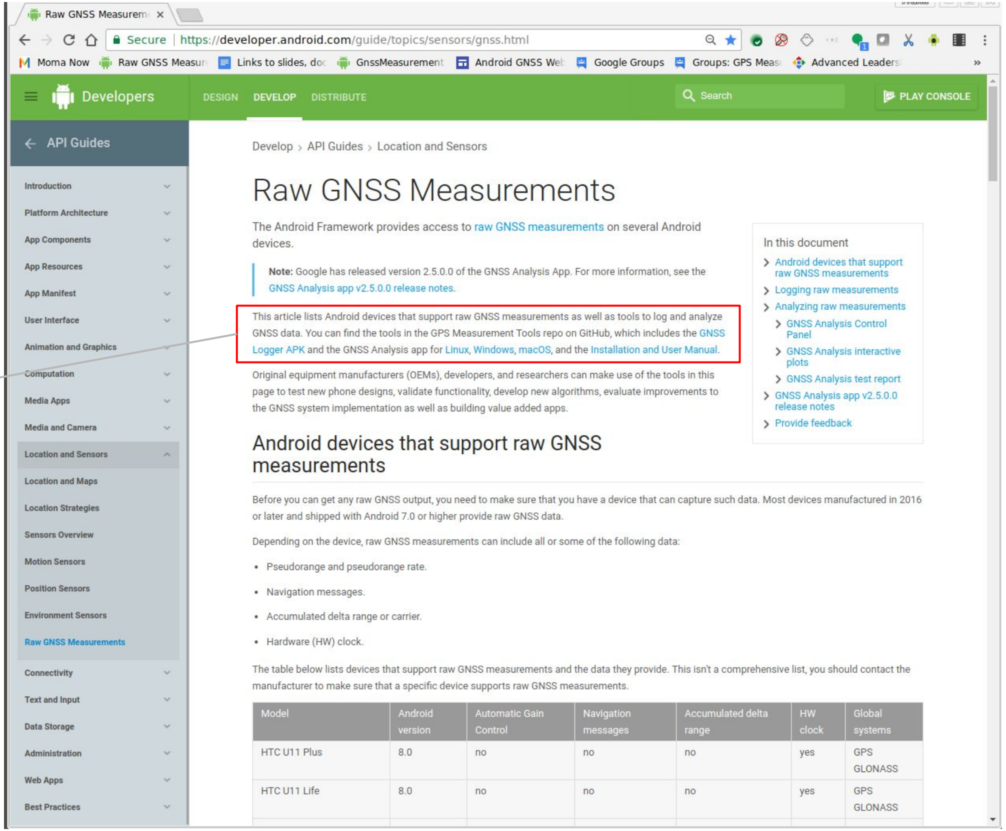
https://g.co/GnssTools

Links to tools:

... find the tools in the GPS Measurement Tools repo on GitHub, which includes the GNSS Logger APK and the GNSS Analysis app for

Linux,
Windows,
macOS,

and the Installation and User Manual.



Develop › API Guides › Location and Sensors

Raw GNSS Measurements

The Android Framework provides access to raw GNSS measurements on several Android devices.

> **Note:** Google has released version 2.5.0.0 of the GNSS Analysis App. For more information, see the GNSS Analysis app v2.5.0.0 release notes.

This article lists Android devices that support raw GNSS measurements as well as tools to log and analyze GNSS data. You can find the tools in the GPS Measurement Tools repo on GitHub, which includes the GNSS Logger APK and the GNSS Analysis app for Linux, Windows, macOS, and the Installation and User Manual.

Original equipment manufacturers (OEMs), developers, and researchers can make use of the tools in this page to test new phone designs, validate functionality, develop new algorithms, evaluate improvements to the GNSS system implementation as well as building value added apps.

Android devices that support raw GNSS measurements

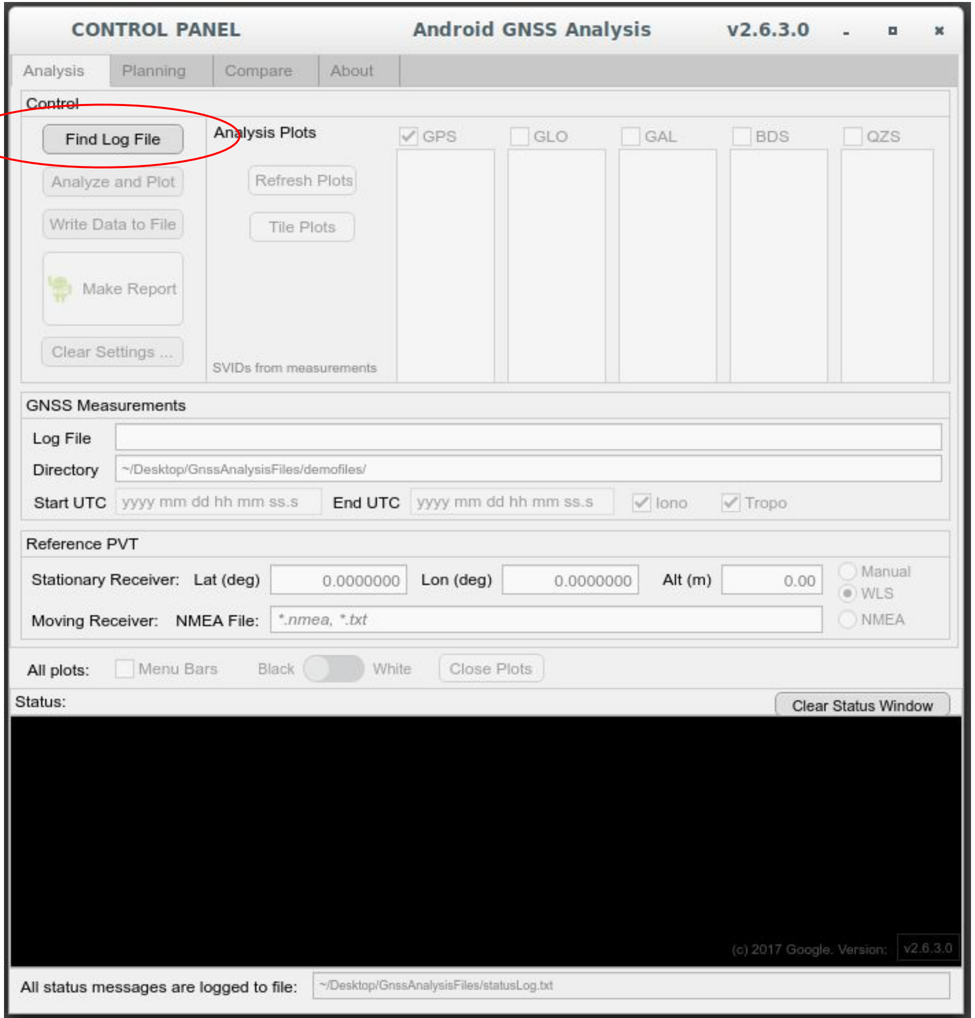
Before you can get any raw GNSS output, you need to make sure that you have a device that can capture such data. Most devices manufactured in 2016 or later and shipped with Android 7.0 or higher provide raw GNSS data.

Depending on the device, raw GNSS measurements can include all or some of the following data:

- Pseudorange and pseudorange rate.
- Navigation messages.
- Accumulated delta range or carrier.
- Hardware (HW) clock.

The table below lists devices that support raw GNSS measurements and the data they provide. This isn't a comprehensive list, you should contact the manufacturer to make sure that a specific device supports raw GNSS measurements.

Model	Android version	Automatic Gain Control	Navigation messages	Accumulated delta range	HW clock	Global systems
HTC U11 Plus	8.0	no	no	no	yes	GPS GLONASS
HTC U11 Life	8.0	no	no	no	yes	GPS GLONASS



CONTROL PANEL Android GNSS Analysis v2.6.3.0

Control

Analysis Plots ☑ GPS ☐ GLO ☐ GAL ☐ BDS ☐ QZS

SVIDs from measurements

GNSS Measurements

Log File:

Directory: ~/Desktop/GnssAnalysisFiles/demofiles/

Start UTC: yyyy mm dd hh mm ss.s End UTC: yyyy mm dd hh mm ss.s ☑ Iono ☑ Tropo

Reference PVT

Stationary Receiver: Lat (deg) 0.0000000 Lon (deg) 0.0000000 Alt (m) 0.00 ○ Manual ● WLS

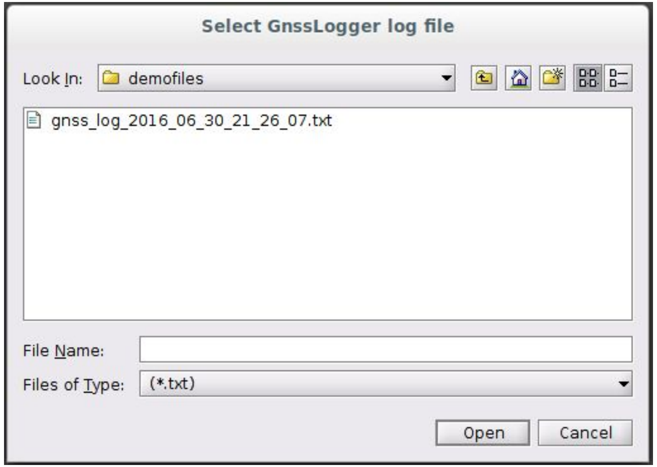
Moving Receiver: NMEA File: *.nmea, *.txt ○ NMEA

All plots: ☐ Menu Bars Black White

Status:

(c) 2017 Google. Version: v2.6.3.0

All status messages are logged to file: ~/Desktop/GnssAnalysisFiles/statusLog.txt



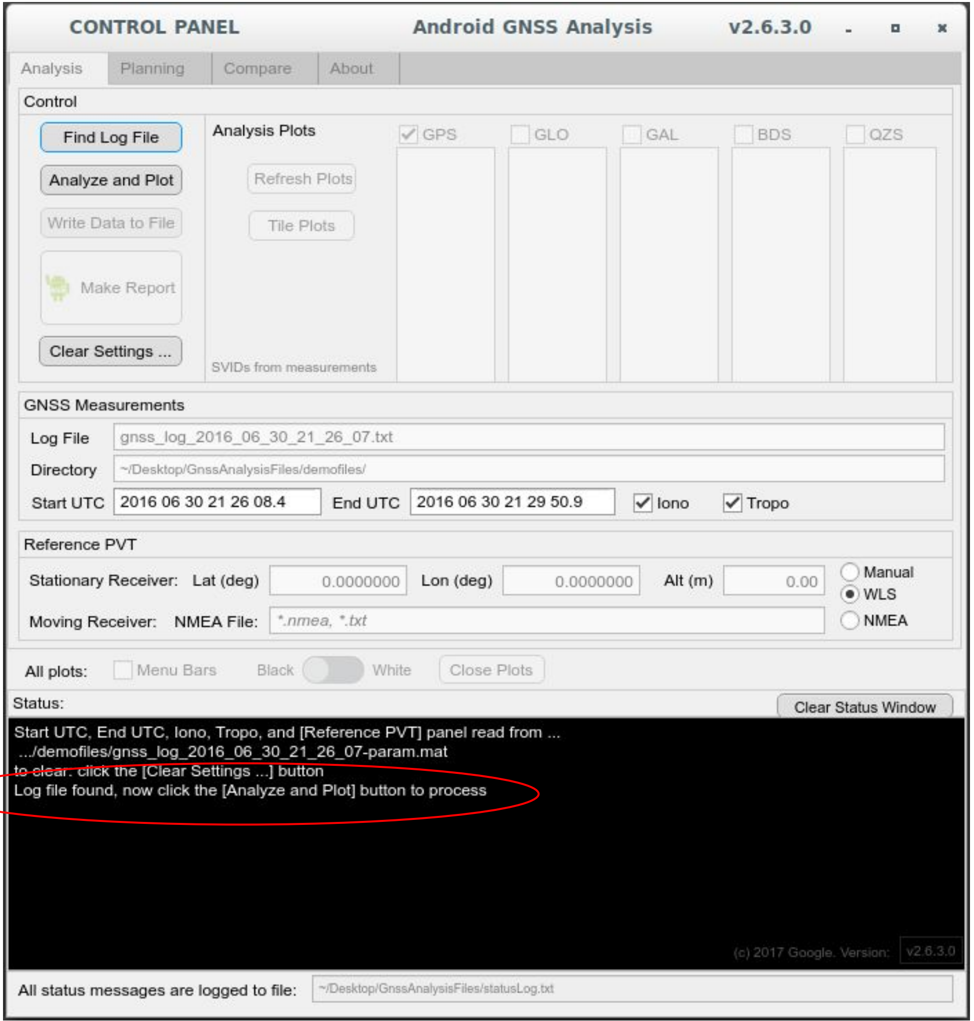
Select GnssLogger log file

Look In: demofiles

gnss_log_2016_06_30_21_26_07.txt

File Name:

Files of Type: (*.txt)



Log File: gnss_log_2016_06_30_21_26_07.txt

Directory: ~/Desktop/GnssAnalysisFiles/demofiles/

Start UTC: 2016 06 30 21 26 08.4 End UTC: 2016 06 30 21 29 50.9 ☑ Iono ☑ Tropo

Stationary Receiver: Lat (deg) 0.0000000 Lon (deg) 0.0000000 Alt (m) 0.00

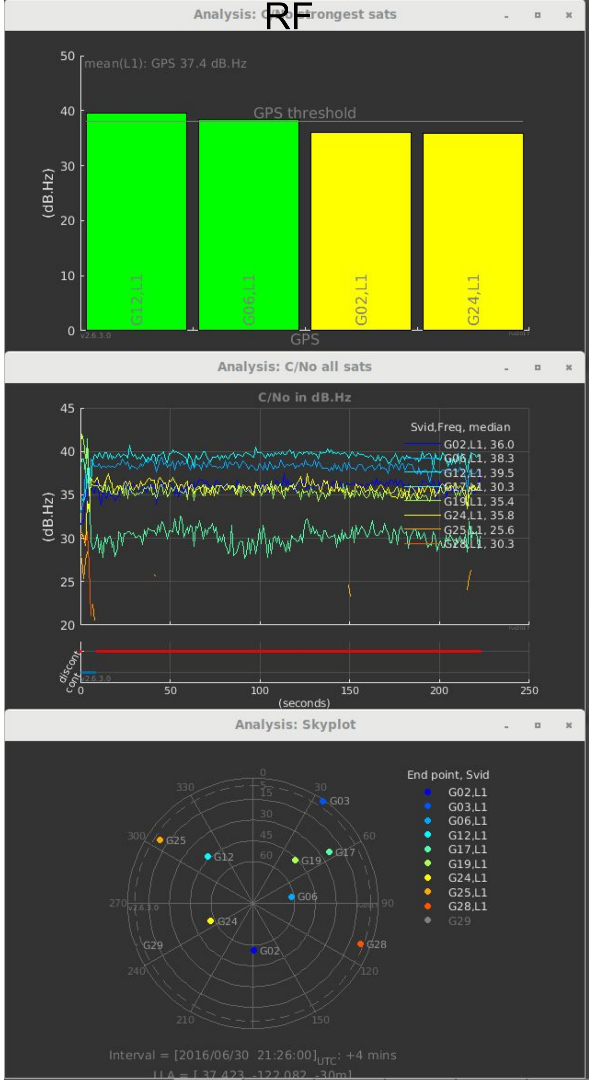
Moving Receiver: NMEA File: *.nmea, *.txt

○ Manual
● WLS
○ NMEA

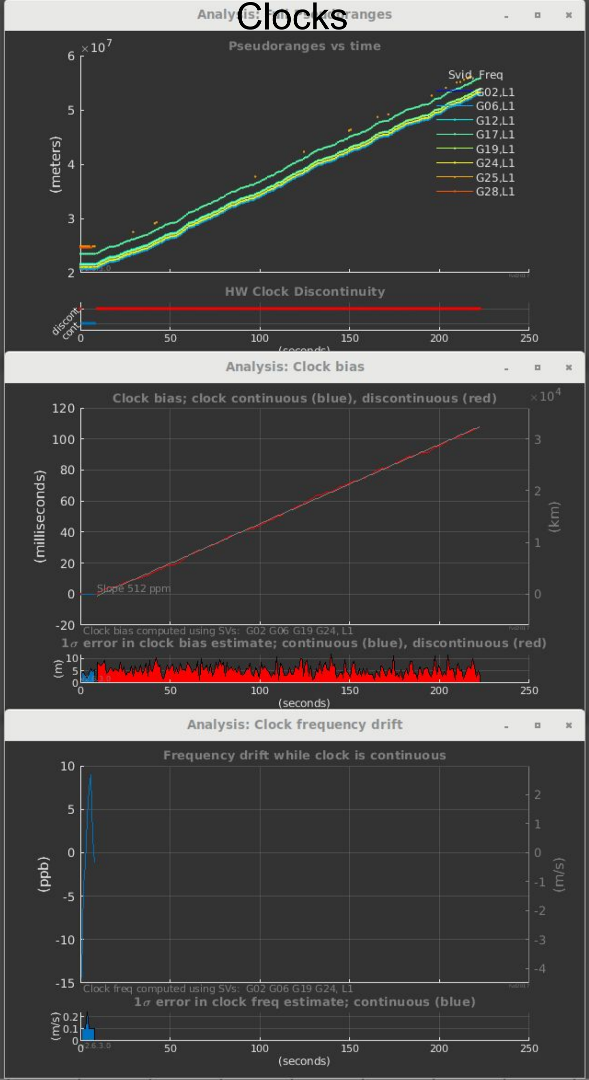
```
Start UTC, End UTC, Iono, Tropo, and [Reference PVT] panel read from ...
 .../demofiles/gnss_log_2016_06_30_21_26_07-param.mat
to clear: click the [Clear Settings ...] button
Log file found, now click the [Analyze and Plot] button to process
```

All status messages are logged to file: ~/Desktop/GnssAnalysisFiles/statusLog.txt

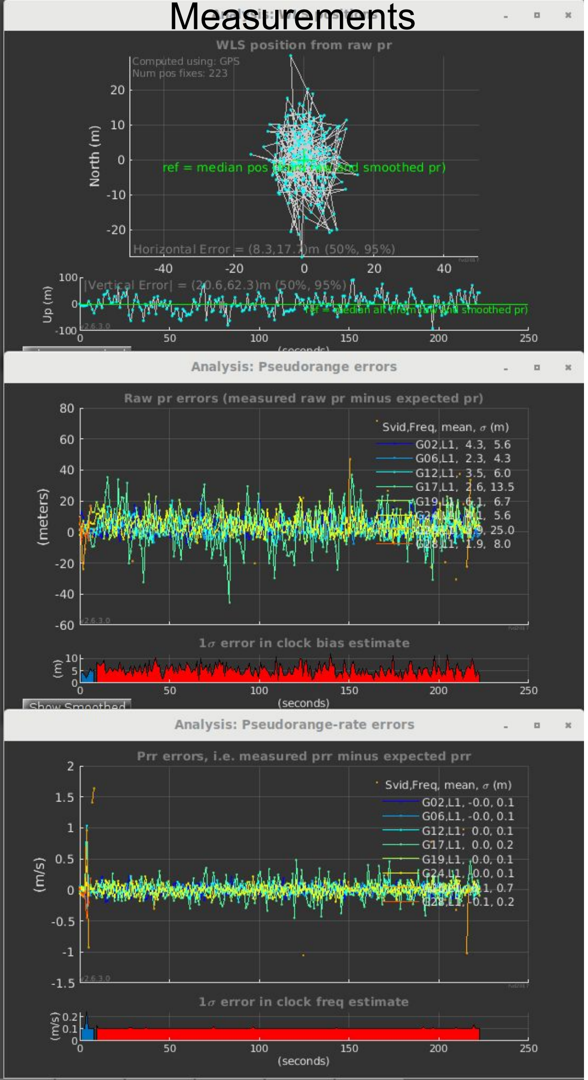
RF

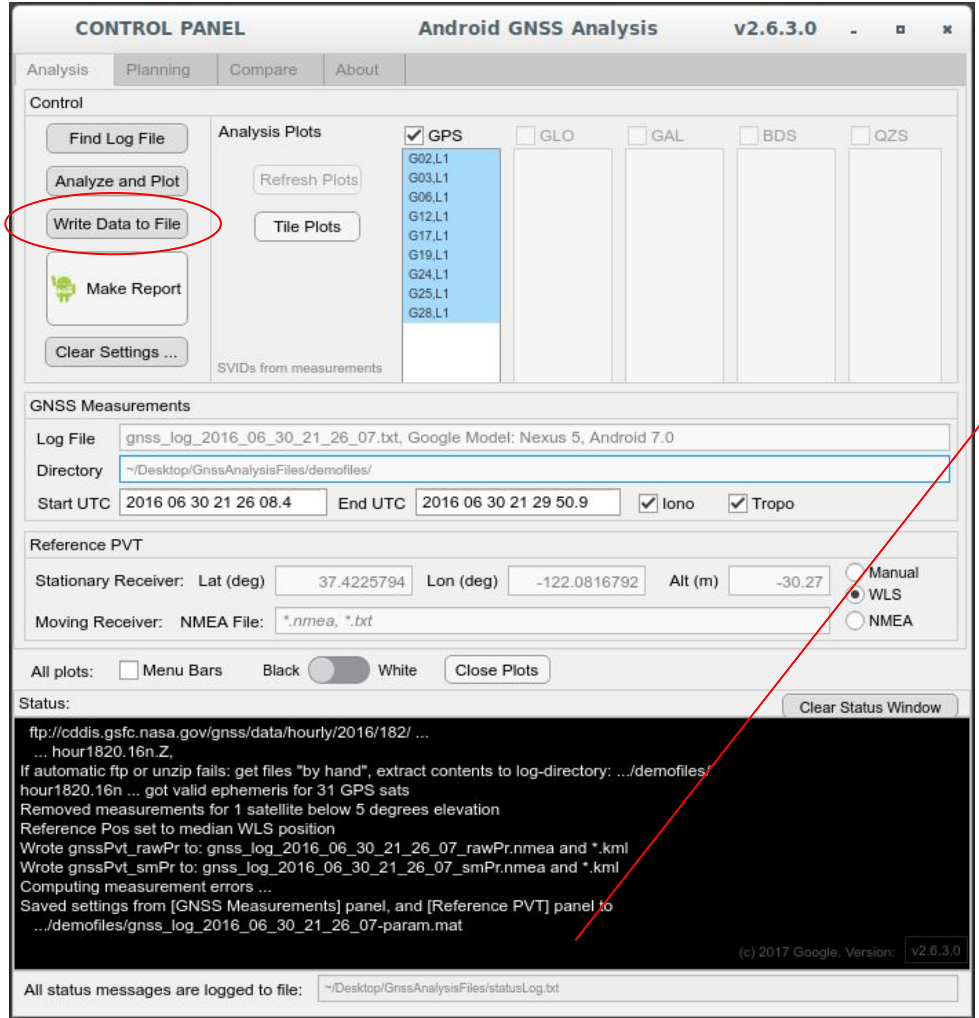


Clocks


Analysis: Raw pseudoranges
Pseudoranges vs time
×10^7
(meters)
Svid, Freq
G02,L1
G06,L1
G12,L1
G17,L1
G19,L1
G24,L1
G25,L1
G28,L1
HW Clock Discontinuity
discont
cont
(seconds)
Analysis: Clock bias
Clock bias; clock continuous (blue), discontinuous (red)
×10^4
(milliseconds)
(km)
Slope 512 ppm
Clock bias computed using SVs: G02 G06 G19 G24, L1
1σ error in clock bias estimate; continuous (blue), discontinuous (red)
(m)
(seconds)
Analysis: Clock frequency drift
Frequency drift while clock is continuous
(ppb)
(m/s)
Clock freq computed using SVs: G02 G06 G19 G24, L1
1σ error in clock freq estimate; continuous (blue)
(m/s)
(seconds)

Measurements





☑ GPS
G02,L1
G03,L1
G06,L1
G12,L1
G17,L1
G19,L1
G24,L1
G25,L1
G28,L1

Log File: gnss_log_2016_06_30_21_26_07.txt, Google Model: Nexus 5, Android 7.0

Directory: ~/Desktop/GnssAnalysisFiles/demofiles/

Start UTC: 2016 06 30 21 26 08.4 End UTC: 2016 06 30 21 29 50.9 ☑ Iono ☑ Tropo

Stationary Receiver: Lat (deg) 37.4225794 Lon (deg) -122.0816792 Alt (m) -30.27

○ Manual
● WLS
○ NMEA

Moving Receiver: NMEA File: *.nmea, *.txt

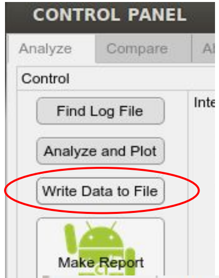
All plots: □ Menu Bars

```
ftp://cddis.gsfc.nasa.gov/gnss/data/hourly/2016/182/ ...
  ... hour1820.16n.Z,
If automatic ftp or unzip fails: get files "by hand", extract contents to log-directory: .../demofiles/
hour1820.16n ... got valid ephemeris for 31 GPS sats
Removed measurements for 1 satellite below 5 degrees elevation
Reference Pos set to median WLS position
Wrote gnssPvt_rawPr to: gnss_log_2016_06_30_21_26_07_rawPr.nmea and *.kml
Wrote gnssPvt_smPr to: gnss_log_2016_06_30_21_26_07_smPr.nmea and *.kml
Computing measurement errors ...
Saved settings from [GNSS Measurements] panel, and [Reference PVT] panel to
  .../demofiles/gnss_log_2016_06_30_21_26_07-param.mat
```

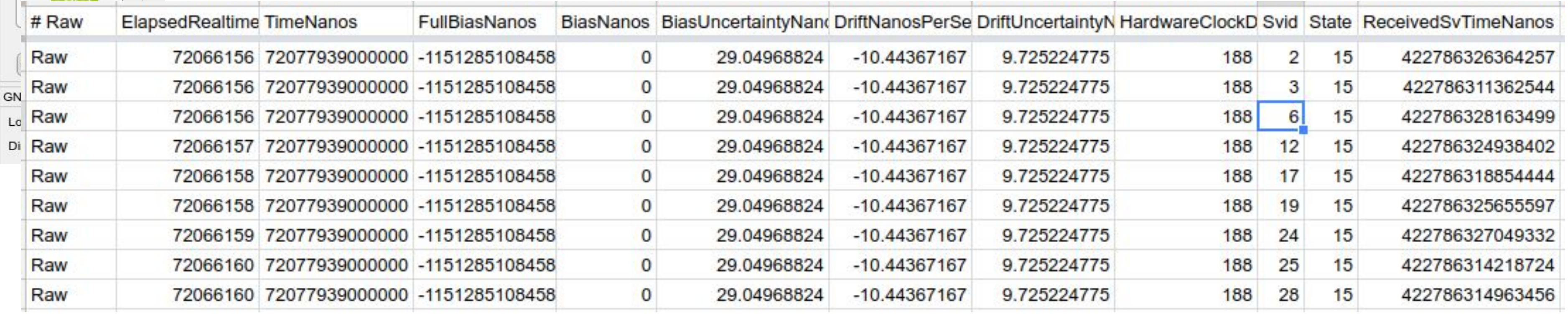
(c) 2017 Google. Version: v2.6.3.0

All status messages are logged to file: ~/Desktop/GnssAnalysisFiles/statusLog.txt

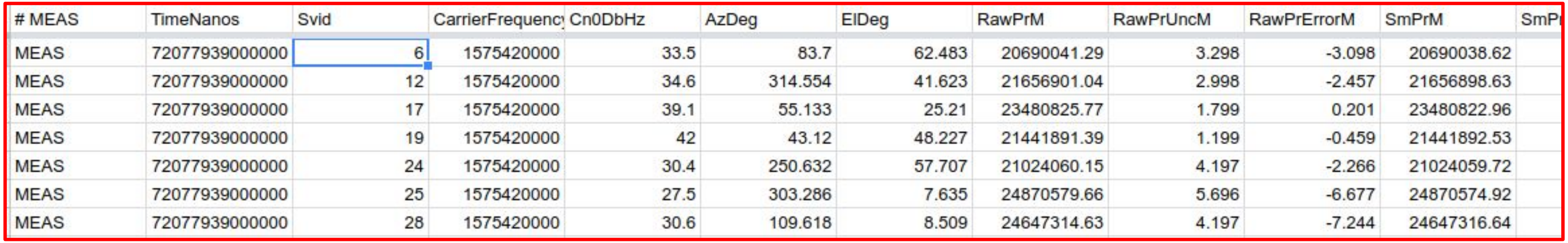
```
Writing derived data to a file ...
Wrote derived data to .../demofiles/gnss_log_2016_06_30_21_26_07.derived
```



Log file of derived data

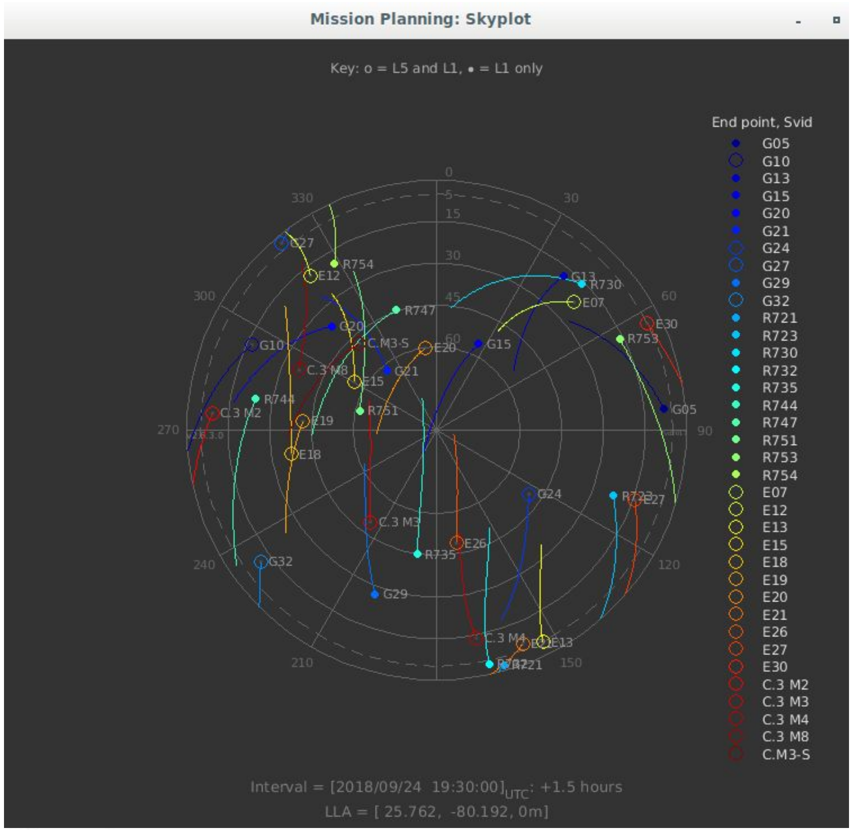


# Raw	ElapsedRealtime	TimeNanos	FullBiasNanos	BiasNanos	BiasUncertaintyNan	DriftNanosPerSe	DriftUncertaintyN	HardwareClockD	Svid	State	ReceivedSvTimeNanos
Raw	72066156	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	2	15	422786326364257
Raw	72066156	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	3	15	422786311362544
Raw	72066156	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	6	15	422786328163499
Raw	72066157	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	12	15	422786324938402
Raw	72066158	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	17	15	422786318854444
Raw	72066158	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	19	15	422786325655597
Raw	72066159	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	24	15	422786327049332
Raw	72066160	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	25	15	422786314218724
Raw	72066160	72077939000000	-1151285108458	0	29.04968824	-10.44367167	9.725224775	188	28	15	422786314963456



# MEAS	TimeNanos	Svid	CarrierFrequenc	Cn0DbHz	AzDeg	ElDeg	RawPrM	RawPrUncM	RawPrErrorM	SmPrM	SmPr
MEAS	72077939000000	6	1575420000	33.5	83.7	62.483	20690041.29	3.298	-3.098	20690038.62	
MEAS	72077939000000	12	1575420000	34.6	314.554	41.623	21656901.04	2.998	-2.457	21656898.63	
MEAS	72077939000000	17	1575420000	39.1	55.133	25.21	23480825.77	1.799	0.201	23480822.96	
MEAS	72077939000000	19	1575420000	42	43.12	48.227	21441891.39	1.199	-0.459	21441892.53	
MEAS	72077939000000	24	1575420000	30.4	250.632	57.707	21024060.15	4.197	-2.266	21024059.72	
MEAS	72077939000000	25	1575420000	27.5	303.286	7.635	24870579.66	5.696	-6.677	24870574.92	
MEAS	72077939000000	28	1575420000	30.6	109.618	8.509	24647314.63	4.197	-7.244	24647316.64	

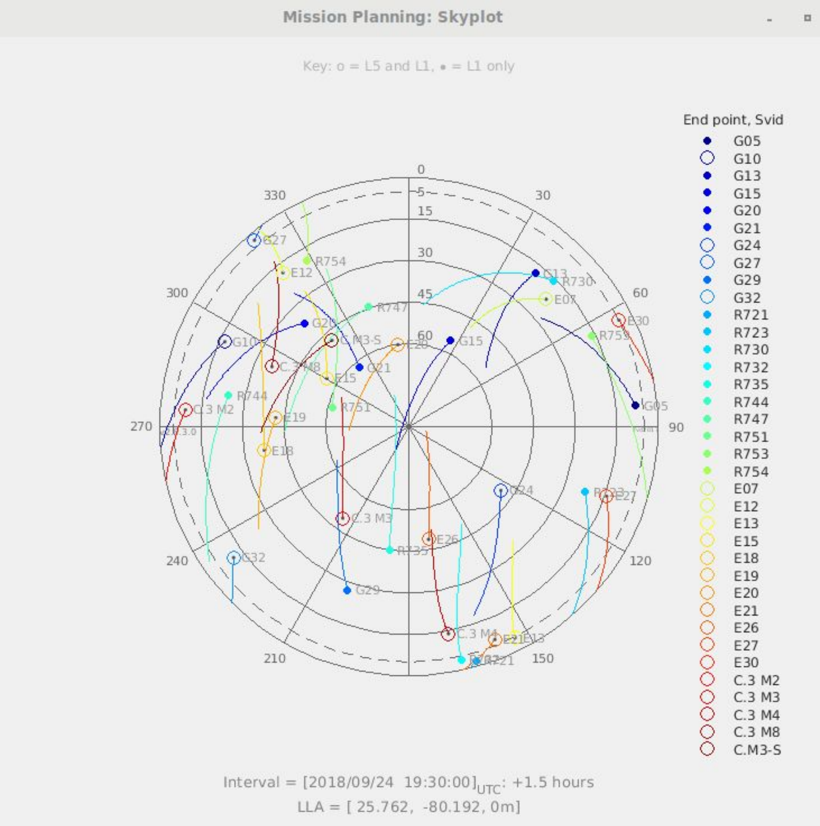
Other useful features of the tools: Mission Planner



Mission Planning: Skyplot

Key: o = L5 and L1, • = L1 only

Interval = [2018/09/24 19:30:00]$_{UTC}$: +1.5 hours
LLA = [25.762, -80.192, 0m]

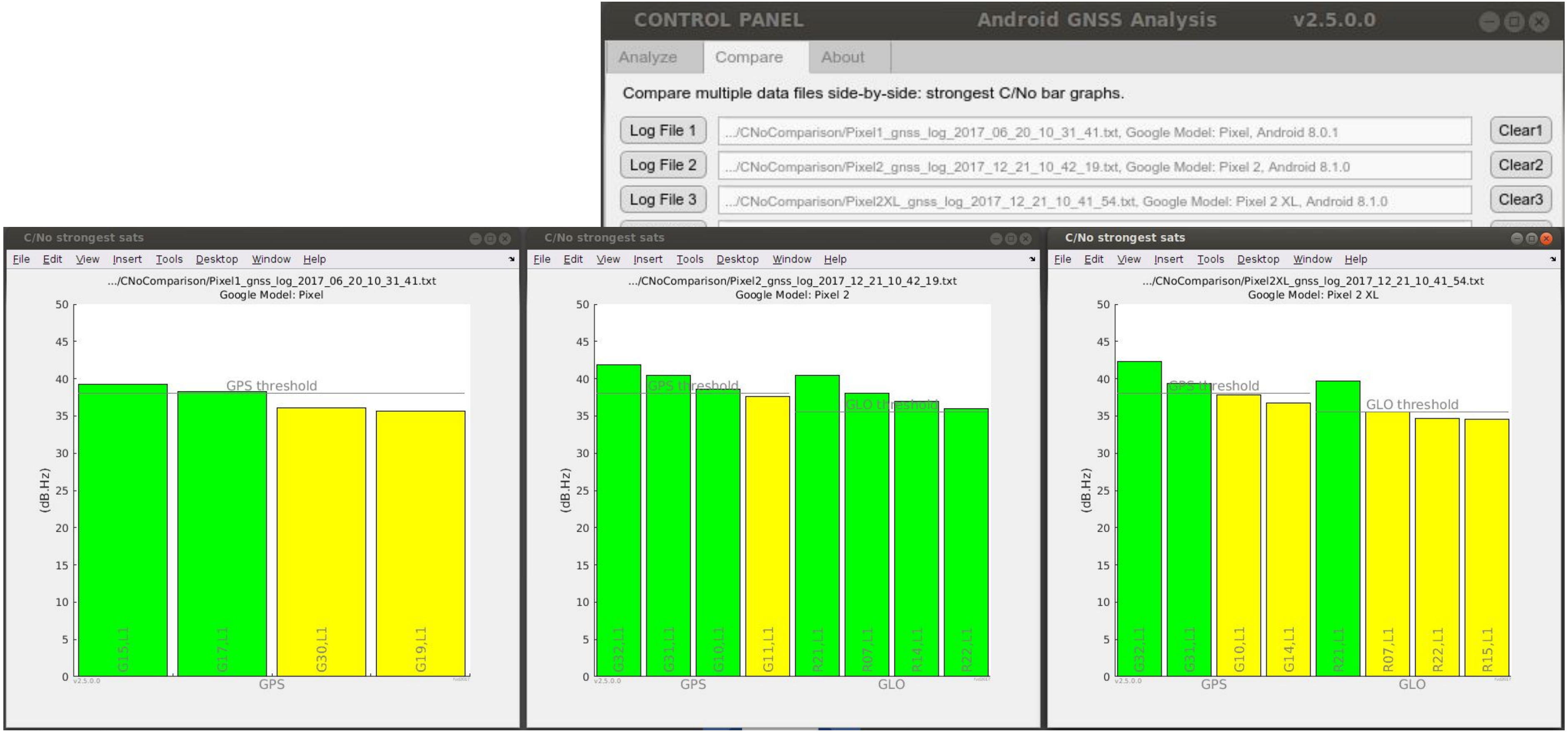


Mission Planning: Skyplot

Key: o = L5 and L1, • = L1 only

Interval = [2018/09/24 19:30:00]$_{UTC}$: +1.5 hours
LLA = [25.762, -80.192, 0m]

Other useful features of the tools: Receiver C/No comparison



CONTROL PANEL Android GNSS Analysis v2.5.0.0

Analyze | Compare | About

Compare multiple data files side-by-side: strongest C/No bar graphs.

Log File 1 ../CNoComparison/Pixel1_gnss_log_2017_06_20_10_31_41.txt, Google Model: Pixel, Android 8.0.1 Clear1

Log File 2 ../CNoComparison/Pixel2_gnss_log_2017_12_21_10_42_19.txt, Google Model: Pixel 2, Android 8.1.0 Clear2

Log File 3 ../CNoComparison/Pixel2XL_gnss_log_2017_12_21_10_41_54.txt, Google Model: Pixel 2 XL, Android 8.1.0 Clear3

Other useful features of the tools:

Illustrated by hands-on exercises ...

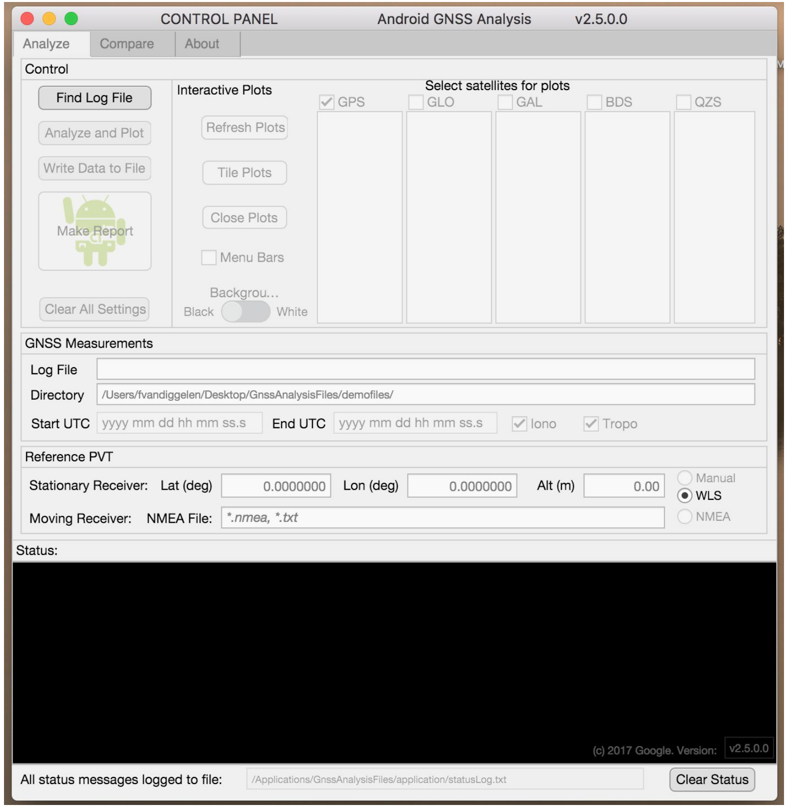
Overview

1. Raw GNSS Measurements
2. Logging Tools
3. How to get Pseudorange
4. Analysis Tools
5. Hands-on Exercises
6. Future: Apps and Research

Hands-on exercises

1. .../GnssAnalysisFiles/demofiles/
   - The demo log file you downloaded with the desktop app
   - Learn the basic capabilities of the analysis tools
2. .../GnssAnalysisFiles/driving/
   - GPS dual-frequency log file with ground-truth nmea
   - Analyze reflections in urban canyons
3. .../GnssAnalysisFiles/ionotropodemo/
   - GNSS log file, stationary, at a known position, open sky
   - Analyze iono and tropo errors.

Exercise #1 .../GnssAnalysisFiles/demofiles/



CONTROL PANEL Android GNSS Analysis v2.5.0.0

Analyze | Compare | About

Control

Find Log File

Analyze and Plot

Write Data to File

Make Report

Clear All Settings

Interactive Plots

Refresh Plots

Tile Plots

Close Plots

□ Menu Bars

Backgrou...

Black White

Select satellites for plots

☑ GPS □ GLO □ GAL □ BDS □ QZS

GNSS Measurements

Log File:

Directory: /Users/fvandiggelen/Desktop/GnssAnalysisFiles/demofiles/

Start UTC: yyyy mm dd hh mm ss.s End UTC: yyyy mm dd hh mm ss.s ☑ Iono ☑ Tropo

Reference PVT

Stationary Receiver: Lat (deg) 0.0000000 Lon (deg) 0.0000000 Alt (m) 0.00

○ Manual
● WLS
○ NMEA

Moving Receiver: NMEA File: *.nmea, *.txt

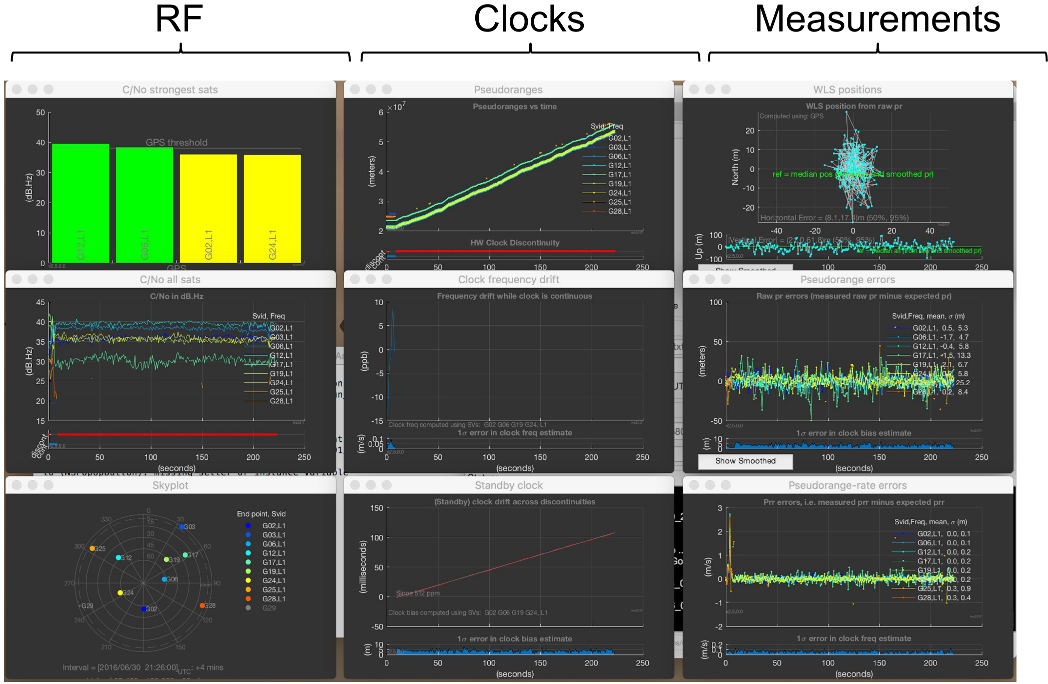
Status:

(c) 2017 Google. Version: v2.5.0.0

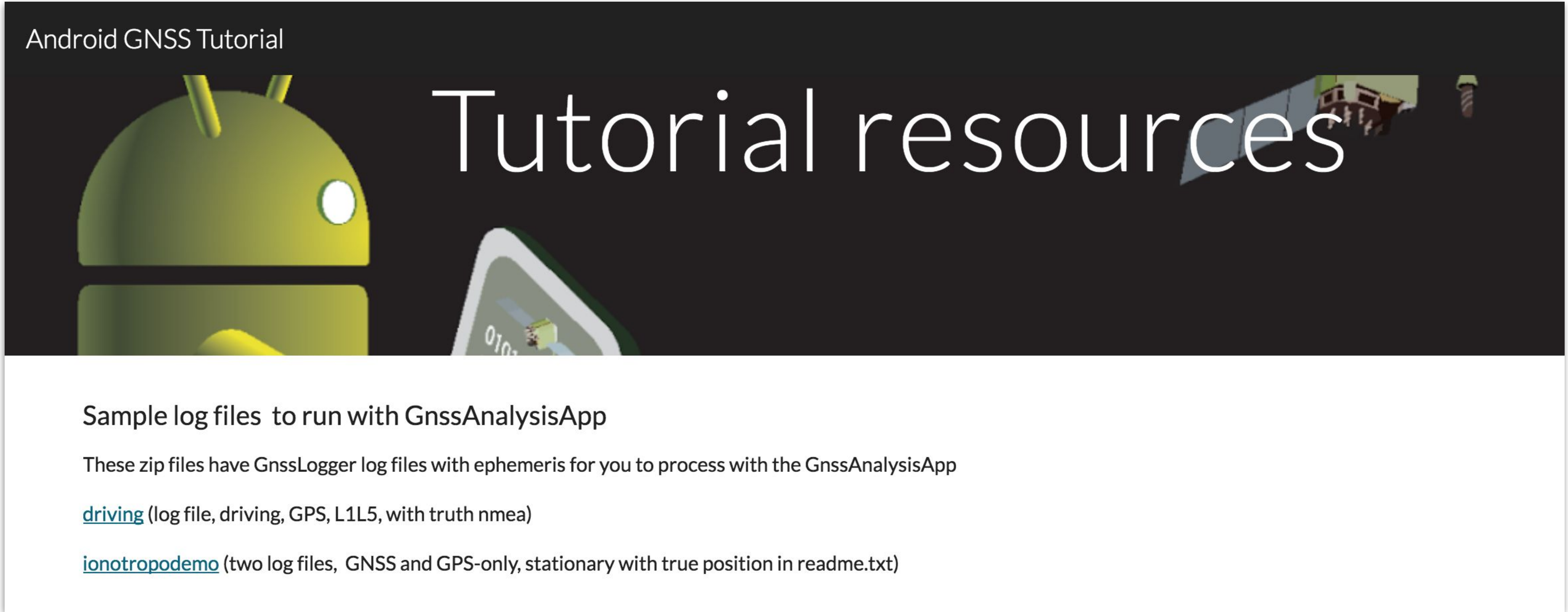
All status messages logged to file: /Applications/GnssAnalysisFiles/application/statusLog.txt

Clear Status

RF Clocks Measurements



Download log files for the following exercises https://sites.google.com/view/gnsstutorial



Android GNSS Tutorial

Tutorial resources

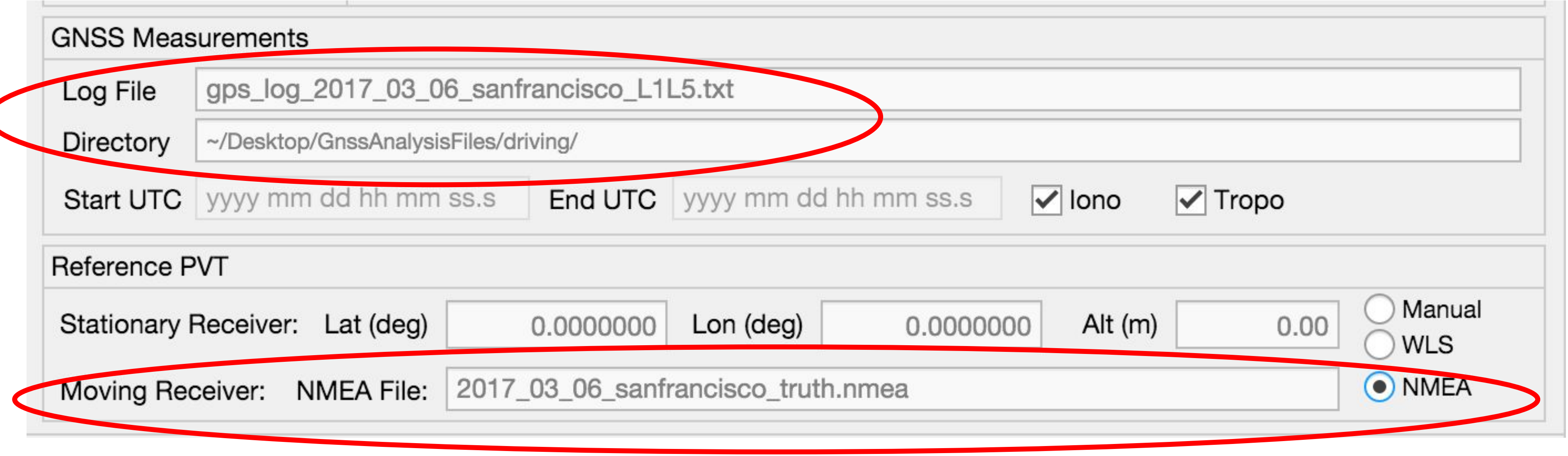
Sample log files to run with GnssAnalysisApp

These zip files have GnssLogger log files with ephemeris for you to process with the GnssAnalysisApp

driving (log file, driving, GPS, L1L5, with truth nmea)

ionotropodemo (two log files, GNSS and GPS-only, stationary with true position in readme.txt)

Exercise #2 .../GnssAnalysisFiles/driving/



GNSS Measurements

Log File: gps_log_2017_03_06_sanfrancisco_L1L5.txt

Directory: ~/Desktop/GnssAnalysisFiles/driving/

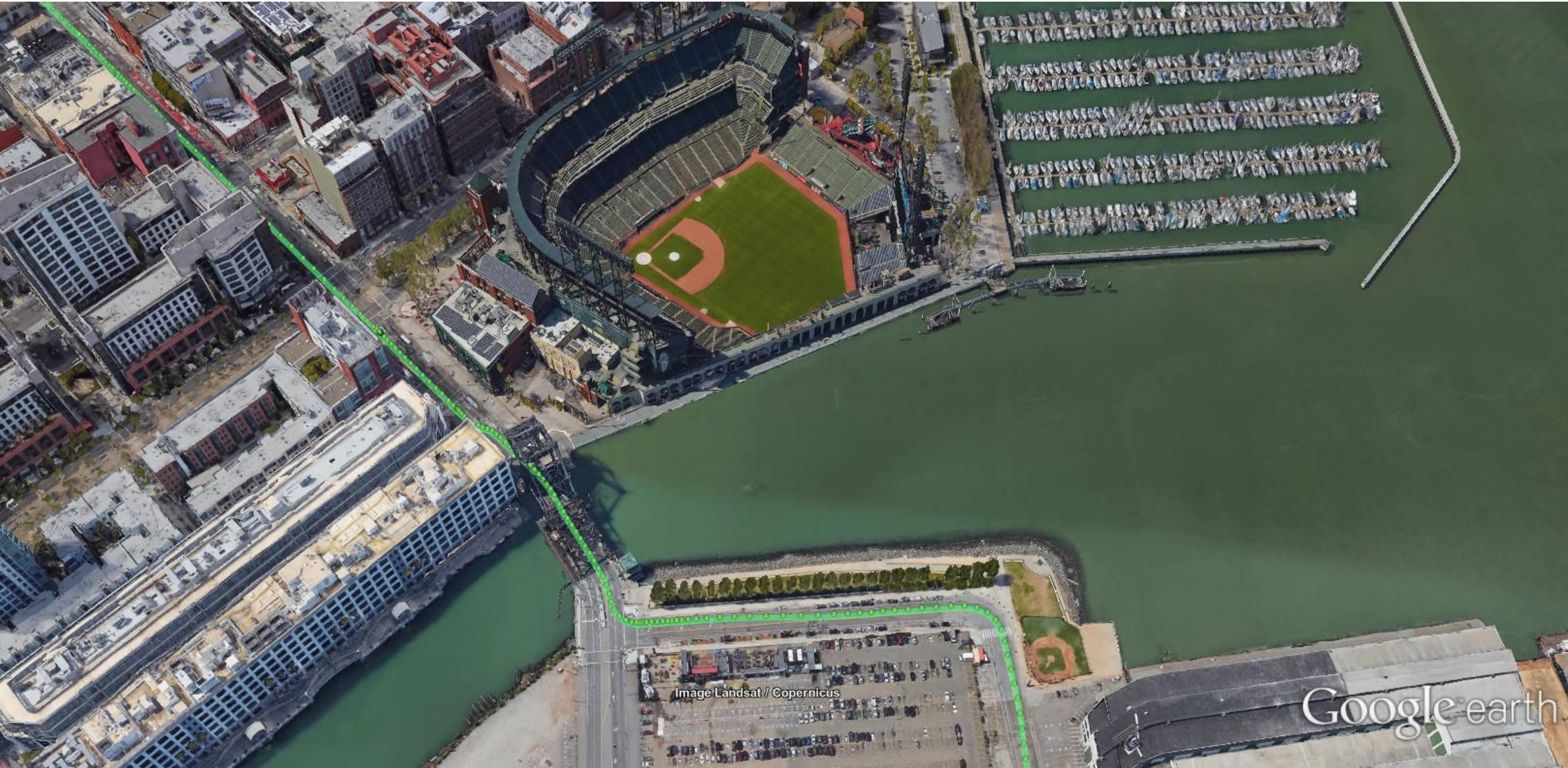
Start UTC: yyyy mm dd hh mm ss.s End UTC: yyyy mm dd hh mm ss.s ☑ Iono ☑ Tropo

Reference PVT

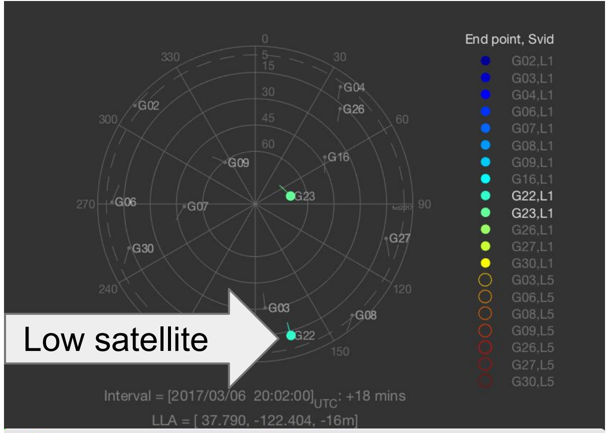
Stationary Receiver: Lat (deg) 0.0000000 Lon (deg) 0.0000000 Alt (m) 0.00 ○ Manual ○ WLS

Moving Receiver: NMEA File: 2017_03_06_sanfrancisco_truth.nmea ● NMEA

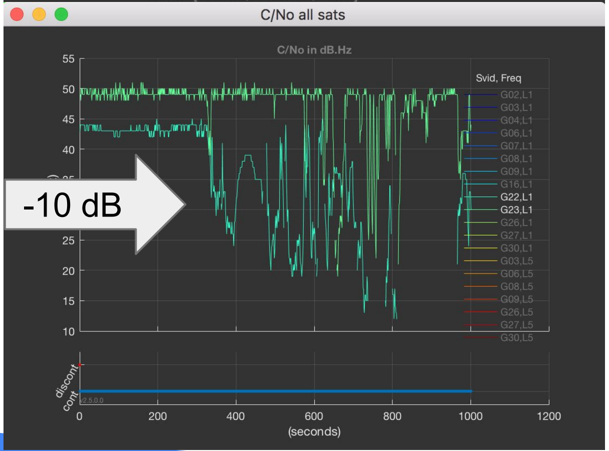
Analysis example, driving into San Francisco:



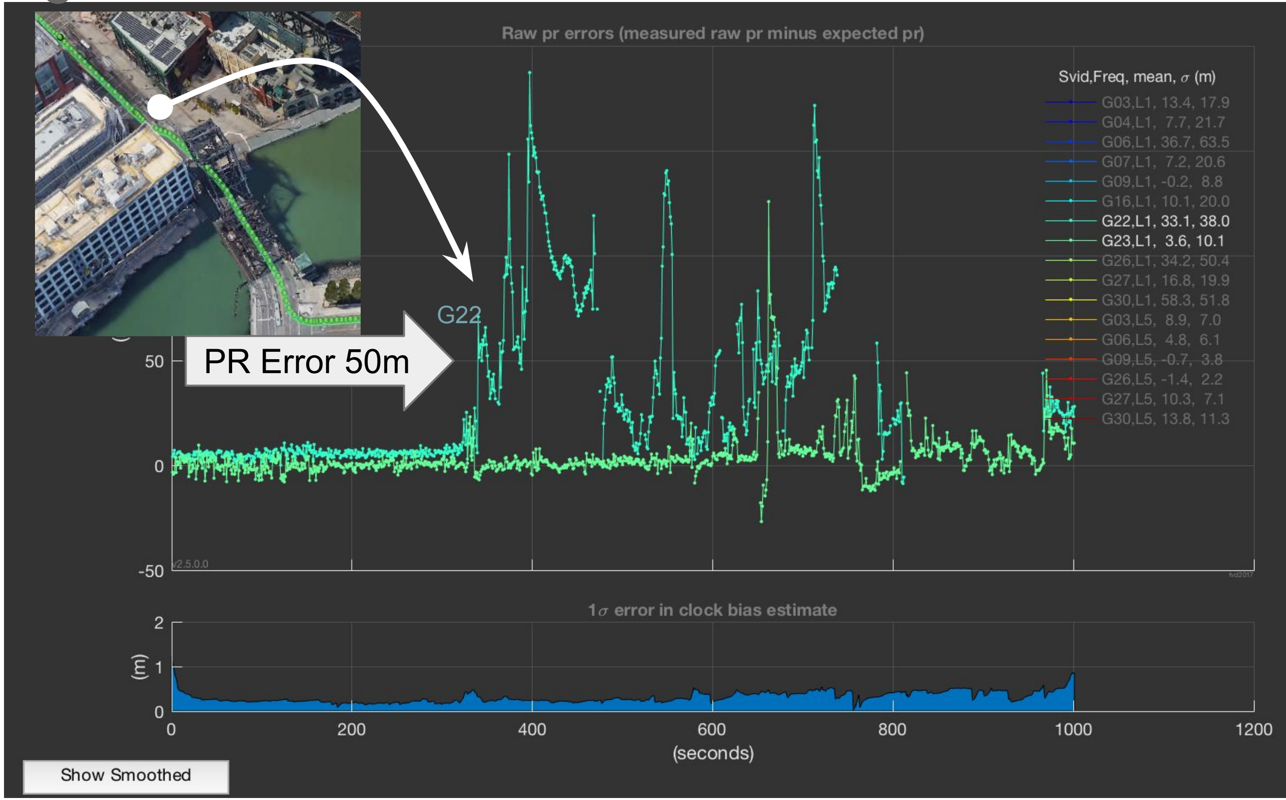
Analysis example, driving into San Francisco:



Low satellite



-10 dB



PR Error 50m

What happened with satellite G22?

Exercise #3 .../GnssAnalysisFiles/ionotropodemo/

1. Use true position for Reference PVT
2. Select highest satelltes to use for clock bias computation
   CustomParam.txt
3. Remove iono and tropo model from analysis

Then error plot will show all errors relative to the highest satellites.

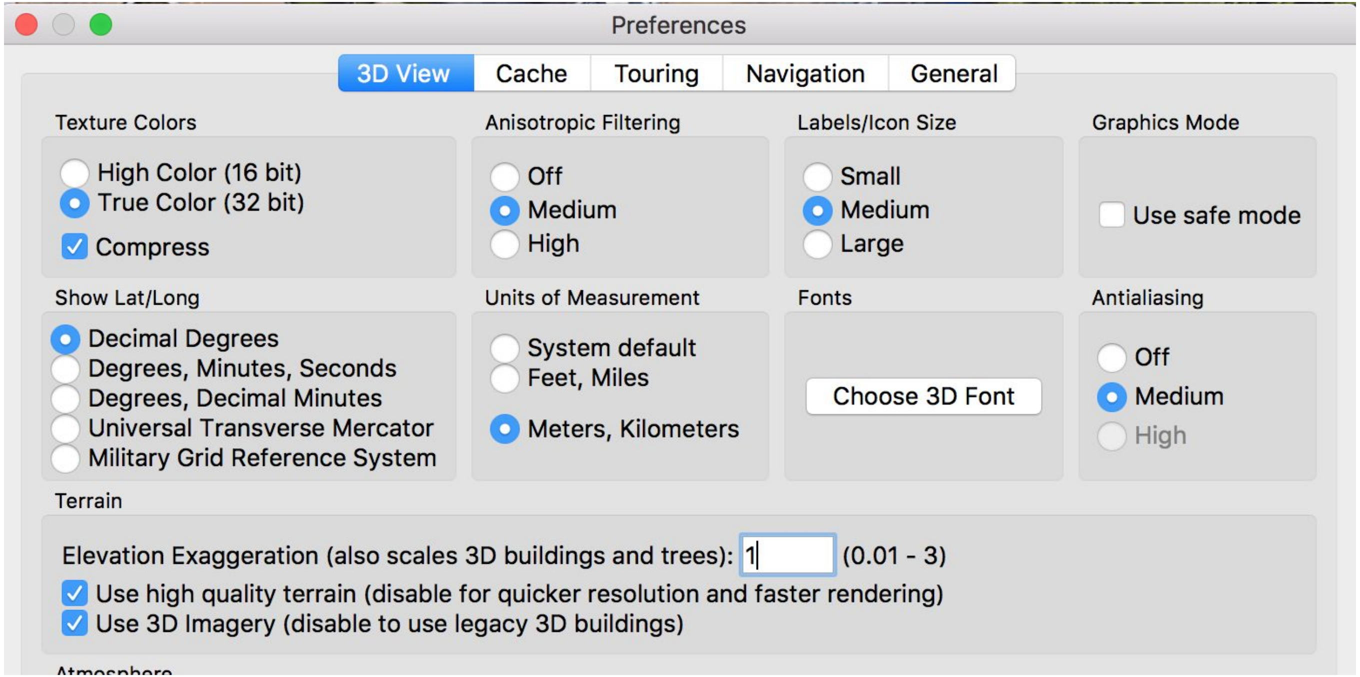
How to get true position from Google Earth (1)

Preferences …

Show Lat/Long
  Decimal Degrees

Units of Measurement
  Meters,Kilometers

Terrain
  Use high quality terrain
  Use 3D imagery



Texture Colors
○ High Color (16 bit)
● True Color (32 bit)
☑ Compress

Anisotropic Filtering
○ Off
● Medium
○ High

Labels/Icon Size
○ Small
● Medium
○ Large

Graphics Mode
□ Use safe mode

Show Lat/Long
● Decimal Degrees
○ Degrees, Minutes, Seconds
○ Degrees, Decimal Minutes
○ Universal Transverse Mercator
○ Military Grid Reference System

Units of Measurement
○ System default
○ Feet, Miles
● Meters, Kilometers

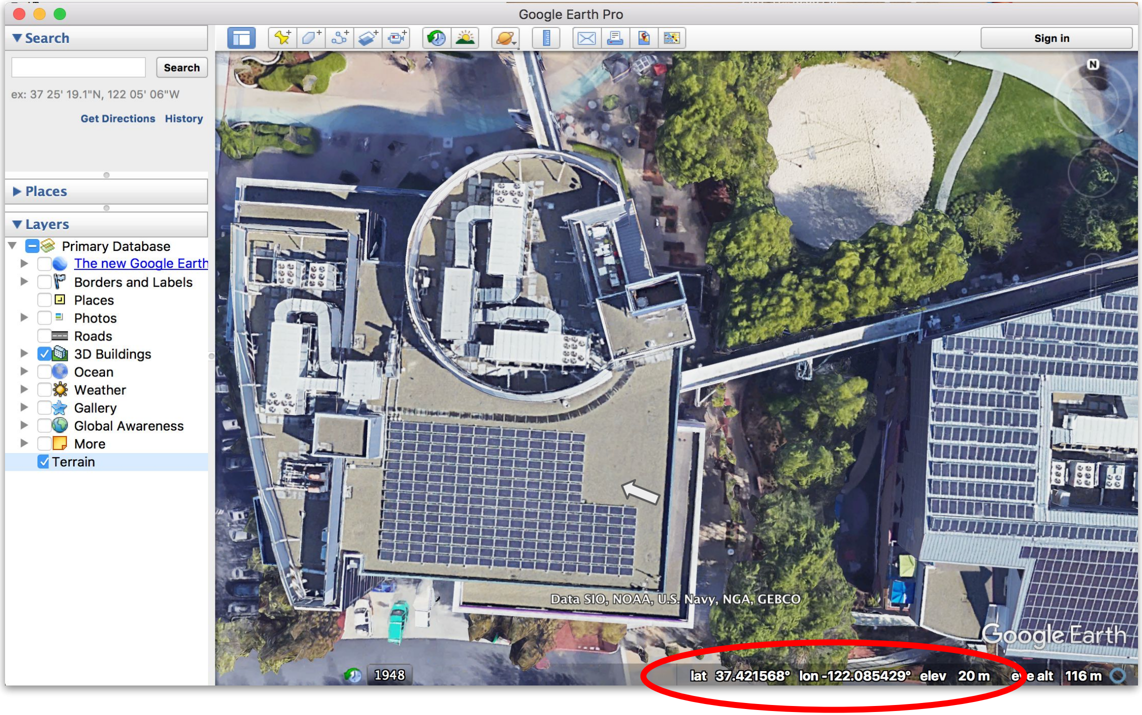
Fonts

Antialiasing
○ Off
● Medium
○ High

Terrain
Elevation Exaggeration (also scales 3D buildings and trees): 1 (0.01 - 3)
☑ Use high quality terrain (disable for quicker resolution and faster rendering)
☑ Use 3D Imagery (disable to use legacy 3D buildings)

Atmosphere

How to get true position from Google Earth (2)



lat 37.421568° lon -122.085429° elev 20 m eye alt 116 m

hG = height above Geoid,
from Google Earth 3D Buildings, 20m
hS = height of stand = 1m
dE = -32, Ellipsoid - Geoid

hE = hG+hS+dE = 20+1-32 = -11 m.

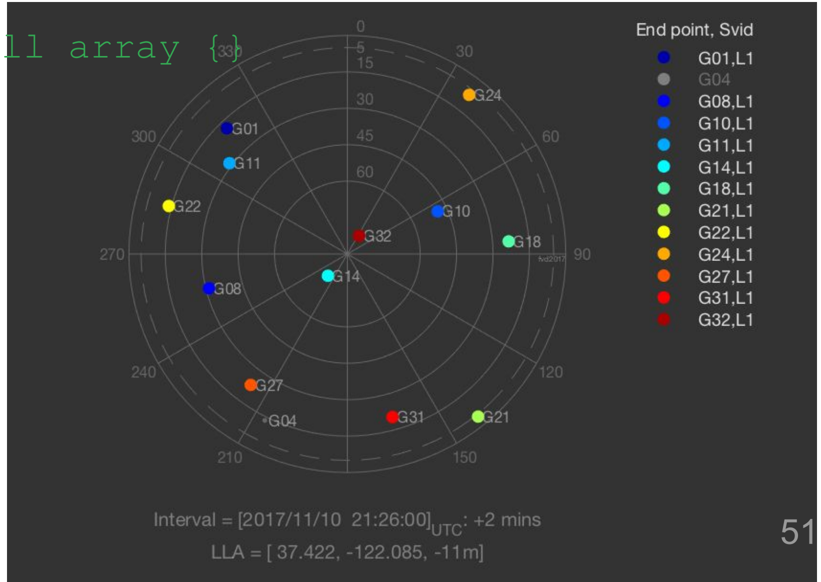
Rooftop true position: 37.421568, -122.085429, -11m

CustomParam.txt

```
%Currently supported:
%param.losSvid = list of svid to use for computing clock (Bc and BcDot)
%template for losSvid.Svid: must have .FreqBand, .Constellation, .Id
GpsL1Svid.Id=0;
GpsL1Svid.Constellation=GnssConstants.GNSS_CONSTELLATION_GPS;
GpsL1Svid.FreqBand=GnssConstants.L1_BAND;  %generic GPS L1 struct
Svid(1)=GpsL1Svid; Svid(1).Id = 32;
%Svid(2)=GpsL1Svid; Svid(2).Id = 14;
param.losSvid.Svids = {Svid};  %pack in a cell array {}
```

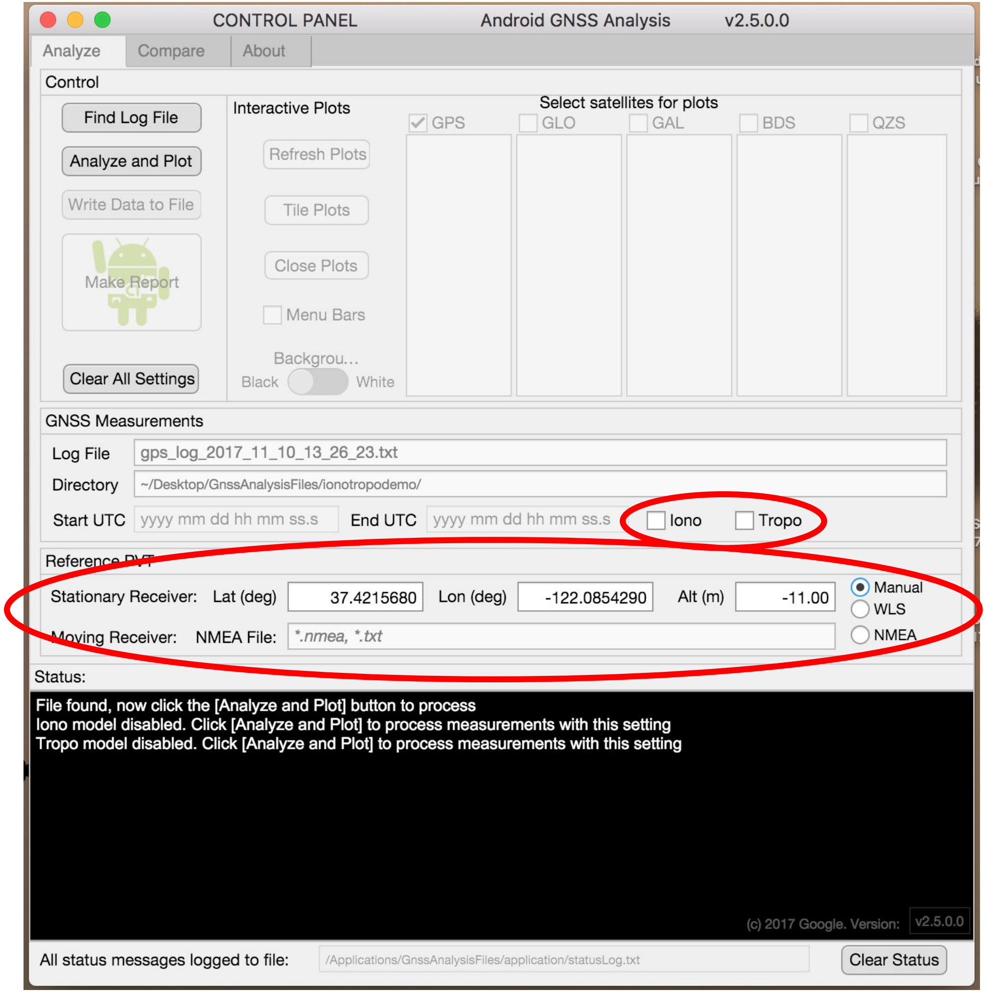
You edit these lines to choose the reference satellite(s) you want.

And place this txt file in the same directory as your log file.



Analyzing, errors:

iono + tropo + SIS[1]



Select satellites for plots

☑ GPS □ GLO □ GAL □ BDS □ QZS

□ Menu Bars

GNSS Measurements

Log File: gps_log_2017_11_10_13_26_23.txt

Directory: ~/Desktop/GnssAnalysisFiles/ionotropodemo/

Start UTC: yyyy mm dd hh mm ss.s End UTC: yyyy mm dd hh mm ss.s □ Iono □ Tropo

Reference PVT

Stationary Receiver: Lat (deg) 37.4215680 Lon (deg) -122.0854290 Alt (m) -11.00

● Manual ○ WLS ○ NMEA

Moving Receiver: NMEA File: *.nmea, *.txt

Status:

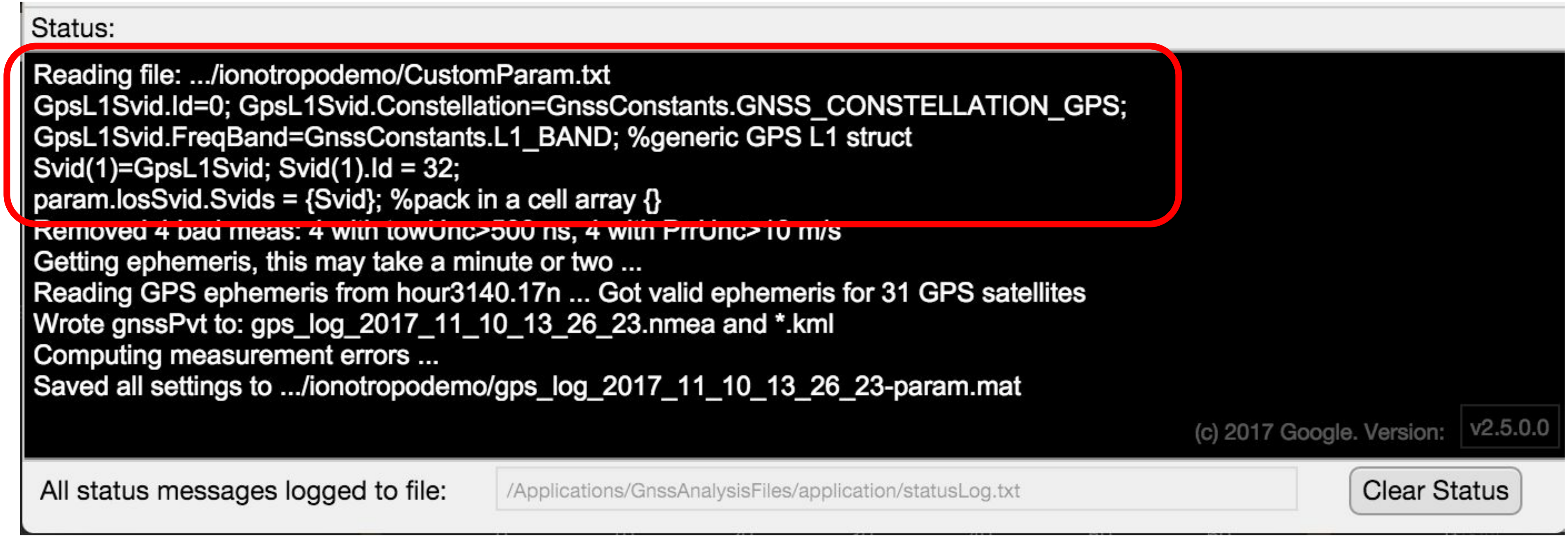
```
File found, now click the [Analyze and Plot] button to process
Iono model disabled. Click [Analyze and Plot] to process measurements with this setting
Tropo model disabled. Click [Analyze and Plot] to process measurements with this setting
```

(c) 2017 Google. Version: v2.5.0.0

All status messages logged to file: /Applications/GnssAnalysisFiles/application/statusLog.txt

[1] SIS errors = Signal In Space errors = errors after applying broadcast ephemeris and clock values

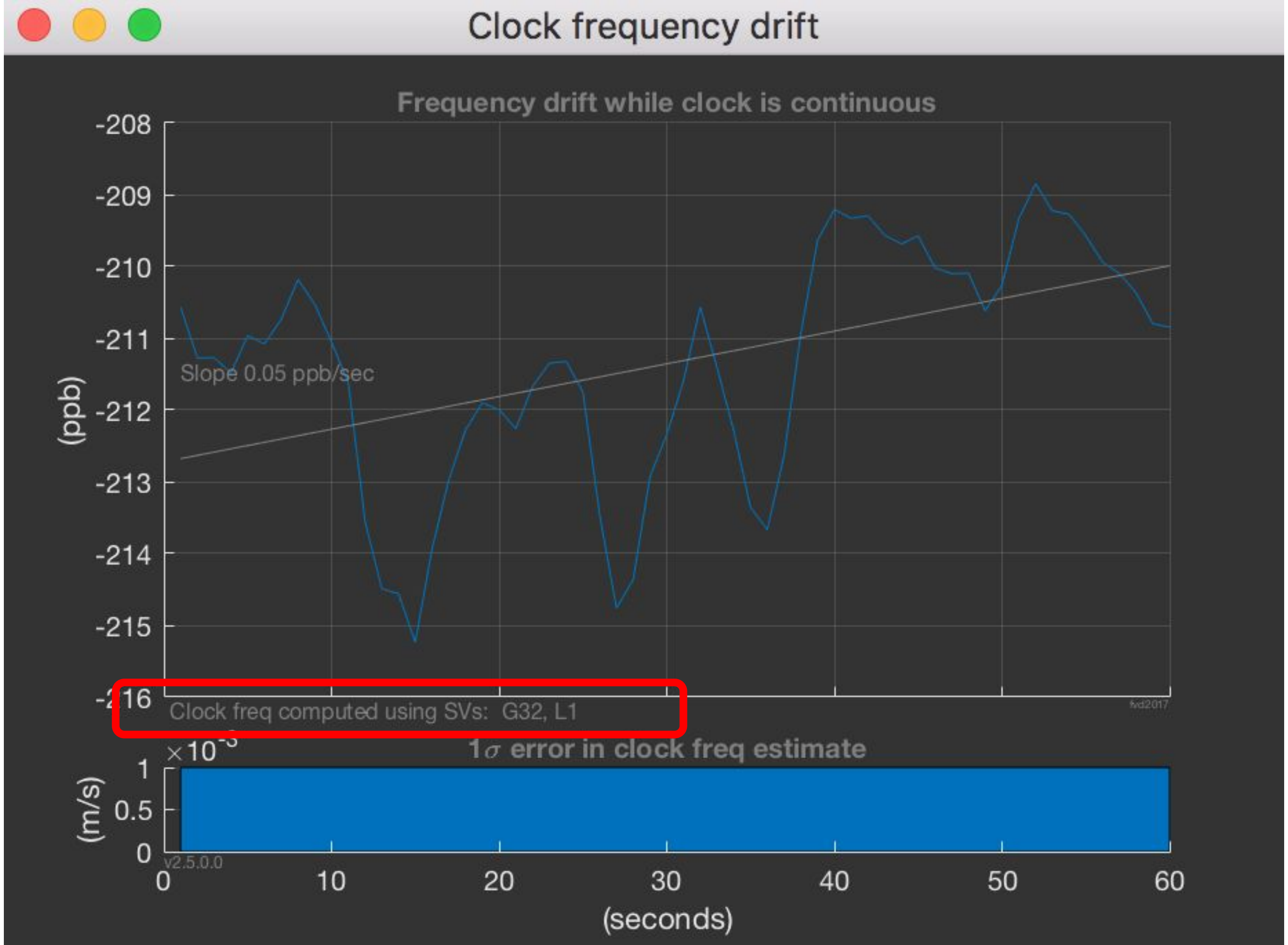
Notice the CustomParam.txt values being applied:



```
Reading file: .../ionotropodemo/CustomParam.txt
GpsL1Svid.Id=0; GpsL1Svid.Constellation=GnssConstants.GNSS_CONSTELLATION_GPS;
GpsL1Svid.FreqBand=GnssConstants.L1_BAND; %generic GPS L1 struct
Svid(1)=GpsL1Svid; Svid(1).Id = 32;
param.losSvid.Svids = {Svid}; %pack in a cell array {}
Removed 4 bad meas: 4 with towUnc>500 ns, 4 with PrrUnc>10 m/s
Getting ephemeris, this may take a minute or two ...
Reading GPS ephemeris from hour3140.17n ... Got valid ephemeris for 31 GPS satellites
Wrote gnssPvt to: gps_log_2017_11_10_13_26_23.nmea and *.kml
Computing measurement errors ...
Saved all settings to .../ionotropodemo/gps_log_2017_11_10_13_26_23-param.mat
```

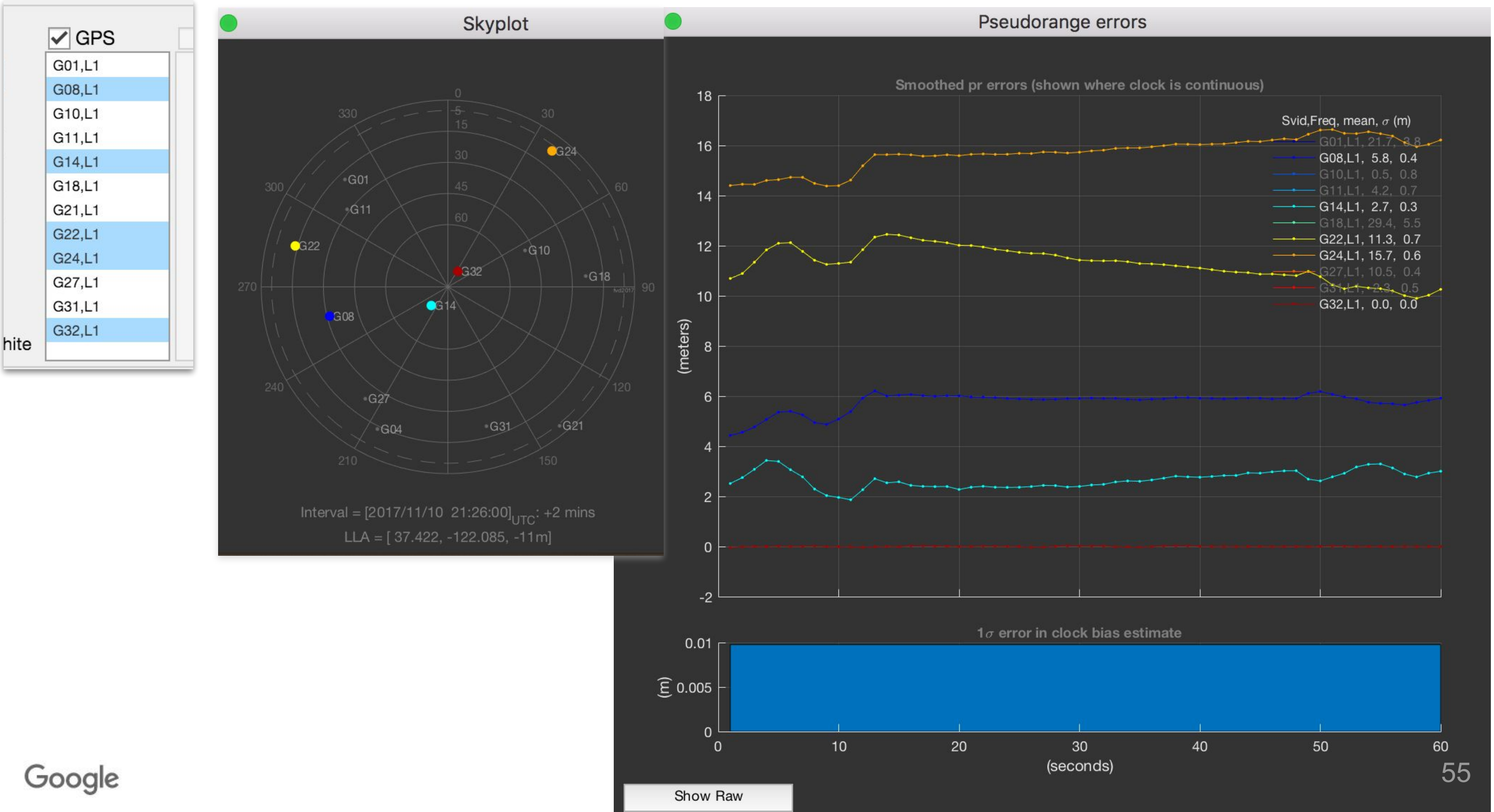
(c) 2017 Google. Version: v2.5.0.0

All status messages logged to file: /Applications/GnssAnalysisFiles/application/statusLog.txt



Clock frequency drift

Clock freq computed using SVs: G32, L1



☑ GPS

- G01,L1
- G08,L1
- G10,L1
- G11,L1
- G14,L1
- G18,L1
- G21,L1
- G22,L1
- G24,L1
- G27,L1
- G31,L1
- G32,L1

hite

Skyplot

Interval = [2017/11/10 21:26:00]$_{UTC}$: +2 mins

LLA = [37.422, -122.085, -11m]

Pseudorange errors

Smoothed pr errors (shown where clock is continuous)

Svid,Freq, mean, σ (m)

G01,L1, 21.7, 3.8
G08,L1, 5.8, 0.4
G10,L1, 0.5, 0.8
G11,L1, 4.2, 0.7
G14,L1, 2.7, 0.3
G18,L1, 29.4, 5.5
G22,L1, 11.3, 0.7
G24,L1, 15.7, 0.6
G27,L1, 10.5, 0.4
G31,L1, 2.3, 0.5
G32,L1, 0.0, 0.0

1σ error in clock bias estimate

Show Raw

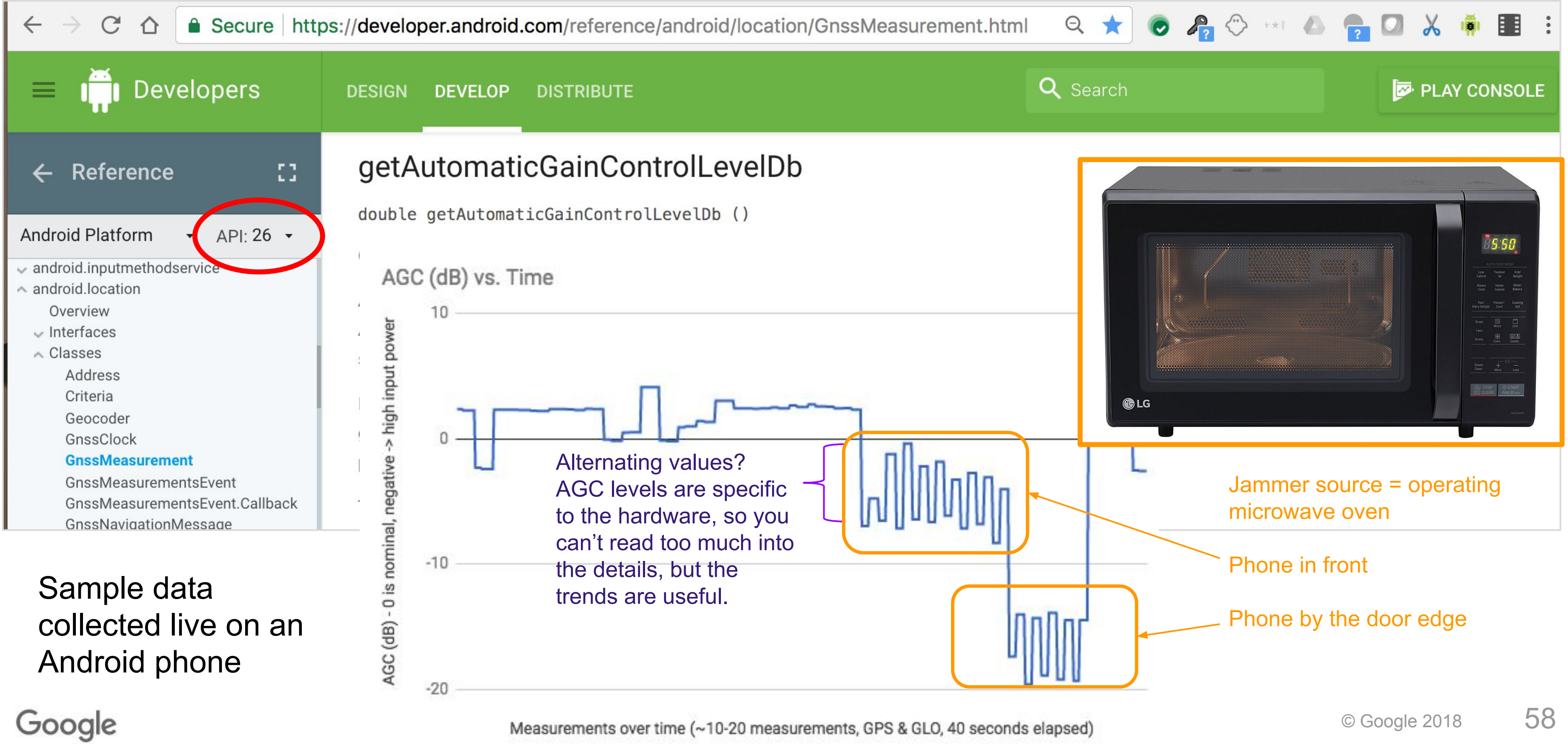
Overview

1. Raw GNSS Measurements
2. Logging Tools
3. How to get Pseudorange
4. Analysis Tools
5. Hands-on Exercises
6. Future: Apps and Research

Future: examples of apps and research

1. Jamming detection
2. Carrier-phase PVT
3. GNSS system monitor
4. Signal analysis (iono, tropo, SIS, multipath, radio noise)

1) Jamming detection


Secure
https://developer.android.com/reference/android/location/GnssMeasurement.html
Developers
DESIGN
DEVELOP
DISTRIBUTE
Search
PLAY CONSOLE
Reference
Android Platform
API: 26
android.inputmethodservice
android.location
Overview
Interfaces
Classes
Address
Criteria
Geocoder
GnssClock
GnssMeasurement
GnssMeasurementsEvent
GnssMeasurementsEvent.Callback
GnssNavigationMessage
AGC (dB) vs. Time
10
0
-10
-20
AGC (dB) - 0 is nominal, negative -> high input power
Measurements over time (~10-20 measurements, GPS & GLO, 40 seconds elapsed)
LG
5:50

getAutomaticGainControlLevelDb

```
double getAutomaticGainControlLevelDb ()
```

Alternating values? AGC levels are specific to the hardware, so you can't read too much into the details, but the trends are useful.

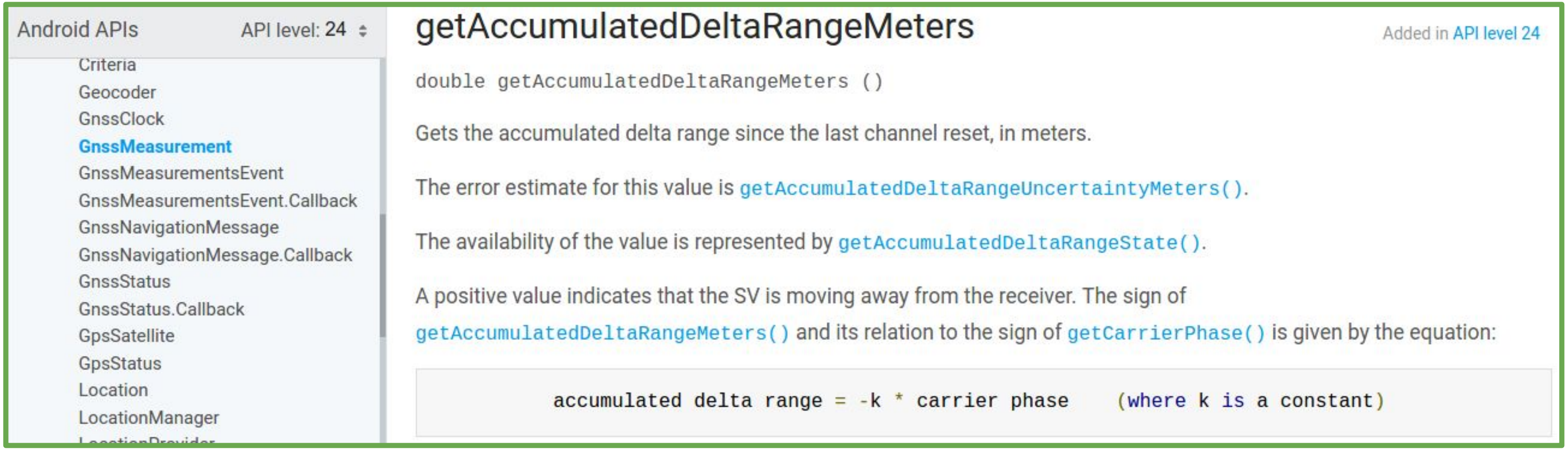
Jammer source = operating microwave oven

Phone in front

Phone by the door edge

Sample data collected live on an Android phone

2) Carrier phase = AccumulatedDeltaRange



getAccumulatedDeltaRangeMeters

Added in API level 24

```
double getAccumulatedDeltaRangeMeters ()
```

Gets the accumulated delta range since the last channel reset, in meters.

The error estimate for this value is `getAccumulatedDeltaRangeUncertaintyMeters()`.

The availability of the value is represented by `getAccumulatedDeltaRangeState()`.

A positive value indicates that the SV is moving away from the receiver. The sign of `getAccumulatedDeltaRangeMeters()` and its relation to the sign of `getCarrierPhase()` is given by the equation:

```
accumulated delta range = -k * carrier phase    (where k is a constant)
```

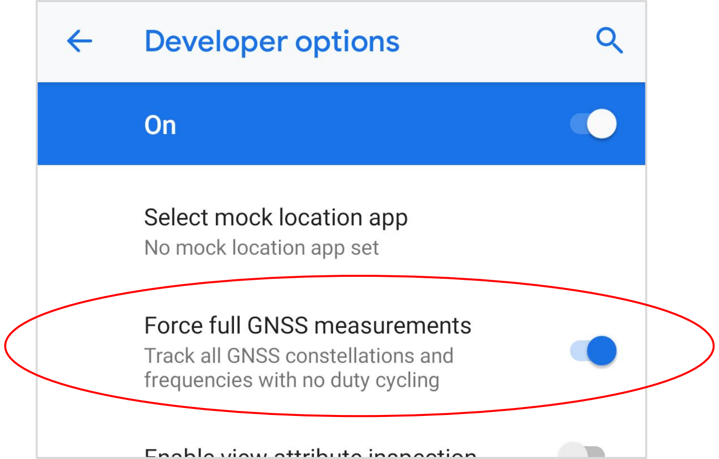
ADR is continuous only when clock is continuous, and there is no duty cycling

Carrier-phase PVT

Enable / Disable Duty Cycling:

In Android P, Google added a Developer option to enable or disable GNSS Duty Cycling

- When selected: The GNSS chipset will not duty cycle and will run at full power - keeping a continuous clock so one can receive continuous carrier phase measurements.
- Look for more information at g.co/GnssTools



Developer options

On

Select mock location app
No mock location app set

Force full GNSS measurements
Track all GNSS constellations and frequencies with no duty cycling

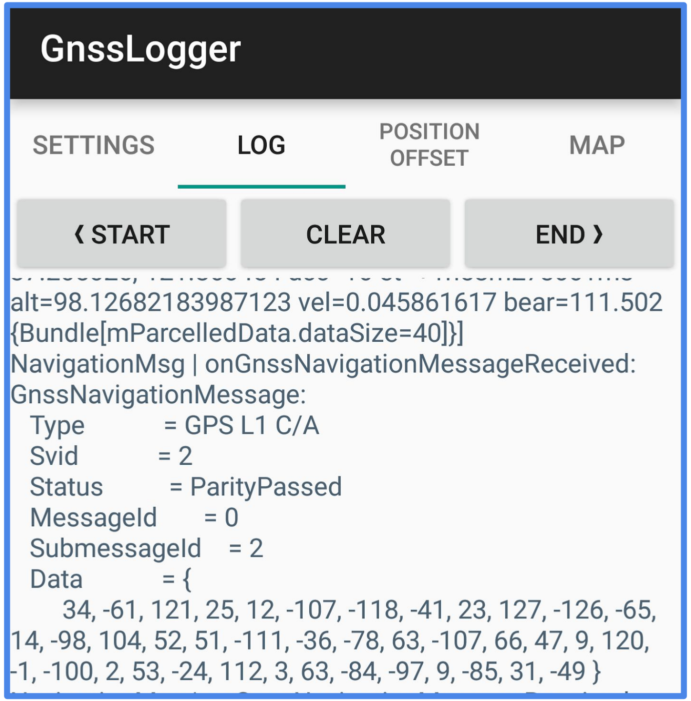
Enable view attribute inspection

Becoming an Android Developer:

1. Go to the settings menu, and scroll to, or search for "About phone." Tap it.
2. Scroll to the bottom, where you see "Build number."
3. Tap it seven (7) times

Also: some phones disable duty cycling automatically when you request raw measurements

3) Decoded Nav data, in GnssLogger:



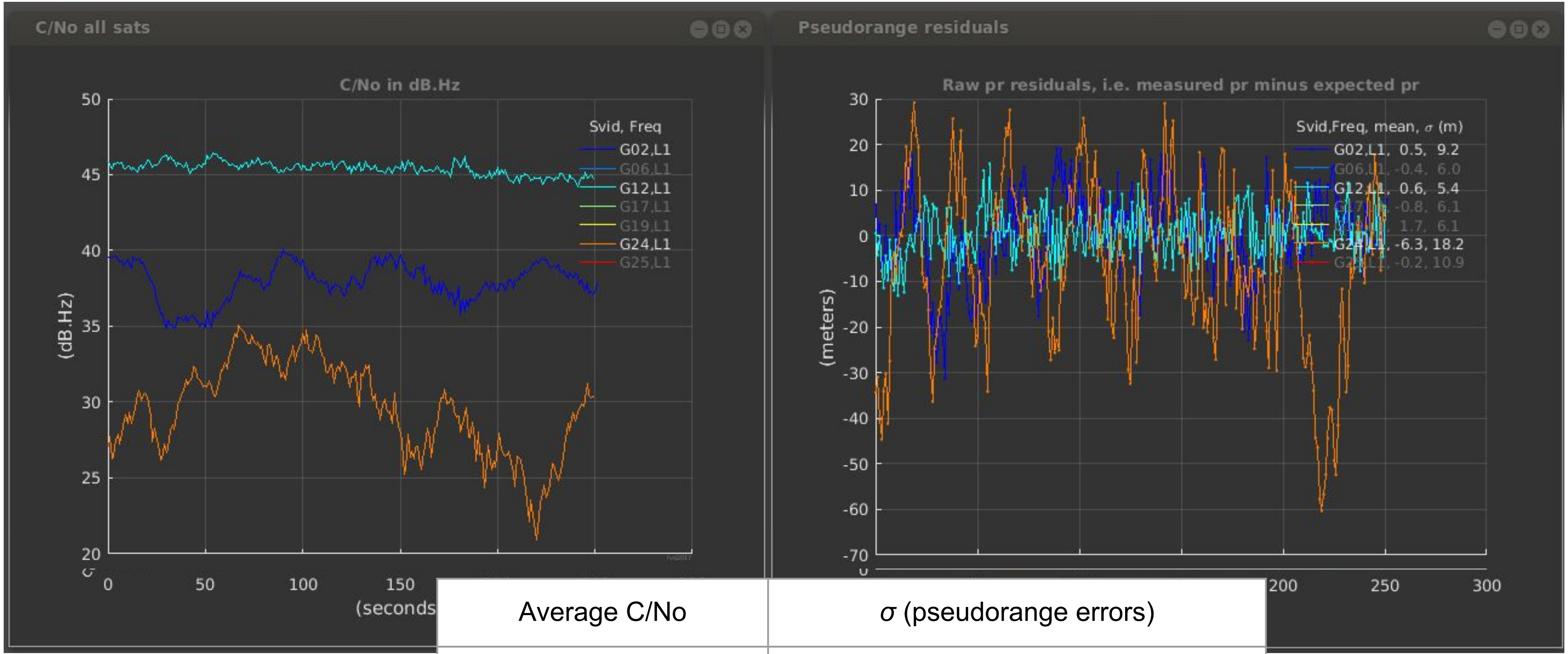
```
alt=98.12682183987123 vel=0.045861617 bear=111.502
{Bundle[mParcelledData.dataSize=40]}]
NavigationMsg | onGnssNavigationMessageReceived:
GnssNavigationMessage:
  Type           = GPS L1 C/A
  Svid           = 2
  Status         = ParityPassed
  MessageId      = 0
  SubmessageId   = 2
  Data           = {
     34, -61, 121, 25, 12, -107, -118, -41, 23, 127, -126, -65,
14, -98, 104, 52, 51, -111, -36, -78, 63, -107, 66, 47, 9, 120,
-1, -100, 2, 53, -24, 112, 3, 63, -84, -97, 9, -85, 31, -49 }
```

And in log file:

```
#
# Header Description:
#
# Version: 1.4.0.0, Platform: N
#
# Nav,Svid,Type,Status,MessageId,Sub-messageId,Data(Bytes)
Nav,2,257,1,0,3,34,-61,121,25,12,-108,107,35,0,33,-42,115,35,46,-77,-78,63,-5,-55,-81,29,76,25,-91,8,-23,106,-11
Nav,12,257,1,0,3,34,-61,121,25,12,-108,107,35,63,-5,2,54,6,-27,120,-7,63,-13,10,55,22,-69,6,-108,6,-99,-120,59,9,
Nav,25,257,1,0,3,34,-61,121,25,12,-108,107,35,63,-8,-63,106,63,25,3,-49,63,-6,-55,-21,55,-49,35,111,6,-63,-56,18,
Nav,98,769,1,0,1,8,87,-128,22,-95,96,-81,-109,-100,30,-104
```

Decimal equivalent of each byte, for example: 0b00111111 = 63

4) Analysis example, radio noise effect on pseudorange:



Average C/No	σ (pseudorange errors)
46 dB.Hz	5.4 m
37 dB.Hz	9.2 m
30 dB.Hz	18.2 m

A few more R&D ideas ...

- Access/security
- Authenticated location
- Automated street/trail mapping
- Drones, sports filming
- Golf - distance to pin
- Lane-level traffic & directions
- Geocaching
- Map-my-yard/landscaping
- Robotic lawnmower
- Outdoor sports precision meas. (ski, bike …): how am I doing on this turn?
- Ecology, environment monitoring
- Athlete tracking
- Parking spot apps
- Cooperative navigation

Summary

- Get raw measurements from Android phones
- Details and software at https://g.co/GnssTools
- Much analysis you can do with the tools directly
- Save derived data, and do further analysis with it
- Pursue research and app development based on these measurements

Resources.
Google: https://g.co/GnssTools (GNSS Logger, Analysis Tools, Open-source code, APIs, Phones, Feedback)
https://sites.google.com/view/gnsstutorial (These slides, sample data sets, pseudorange spreadsheet)
http://insidegnss.com/gnss-analysis-tools-from-google/
http://gpsworld.com/how-to-achieve-1-meter-accuracy-in-android/
One-meter location-accuracy from Android devices (Google I/O '18), https://youtu.be/vywGgSrGODU
ESA: White paper on Android Raw Measurements,
www.gsa.europa.eu/newsroom/news/available-now-white-paper-using-gnss-raw-measurements-android-devices
https://www.gsa.europa.eu/system/files/reports/gnss_raw_measurement_web.pdf

the end. Thank You!