Orbit optimization for measurements during eclipse and maximization of ground stations visibility

Francesco Dattilo

Thesis supervisor: Professor Mario Bertaina
Company tutors: Daniela Borla Tridon, Ilenya Salvoni

Stage in Altec

Aerospace Logistics Technology Engineering Company

Purpose

Analysis and optimization of a low Earth orbit (LEO) in order to maximize:

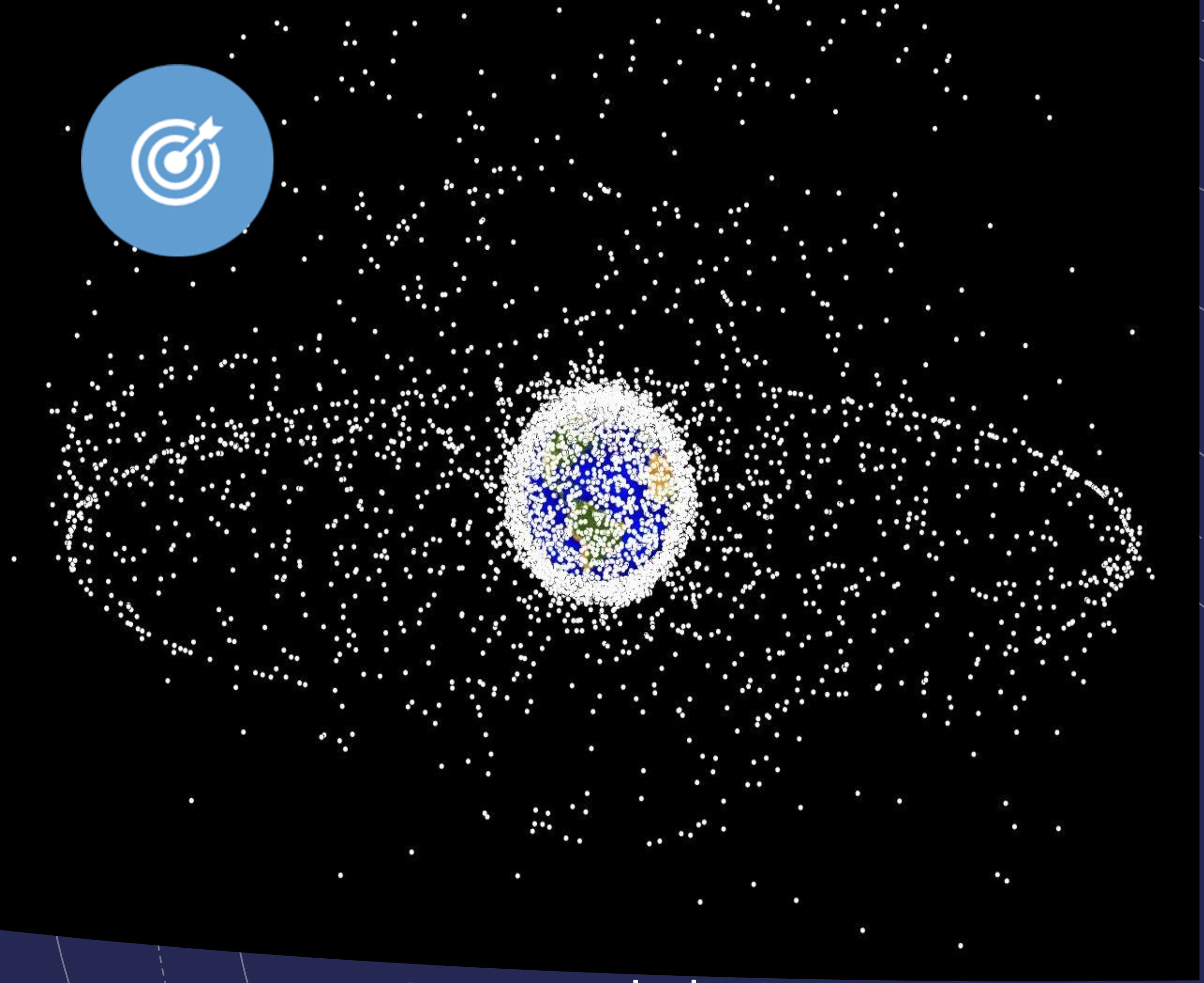
- the eclipse time for night light measurements
- the visibility over a defined ground stations network for data downlink

Summary

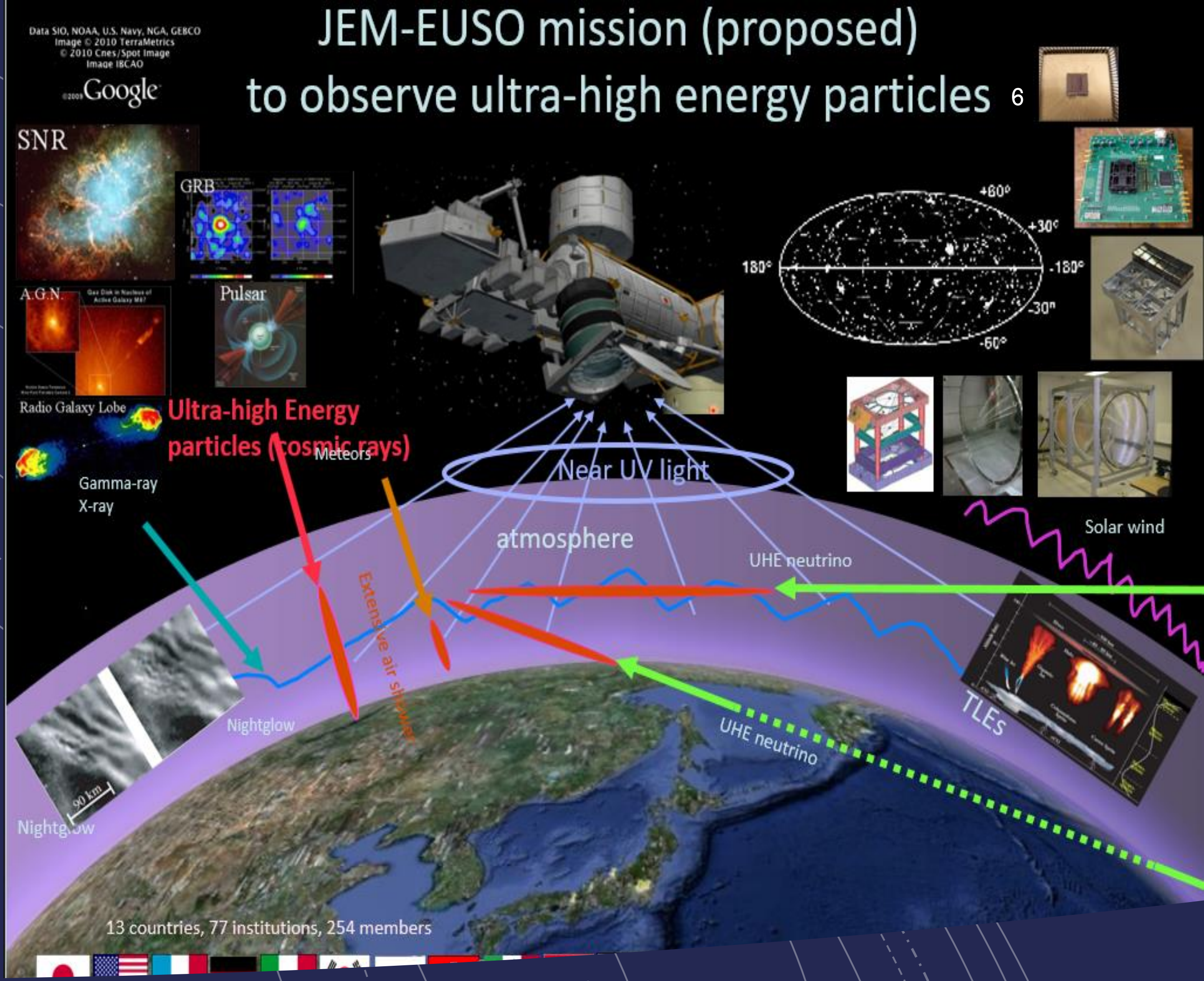
- INTRODUCTION
- OREKIT
- GROUND STATION NETWORK
- ECLIPSE ANALYSIS
- ORBIT OPTIMIZATION
- CONCLUSIONS

Introduction

- The importance of data acquisition in eclipse
- The time of trasmission with ground stations



Space debris



Cosmic rays

Physical interest

Orbit analysis

OREKIT

- Open source, free, but requires to be implemented

STK

- Intuitive, but close and very expensive

Orbit extrapolation kit

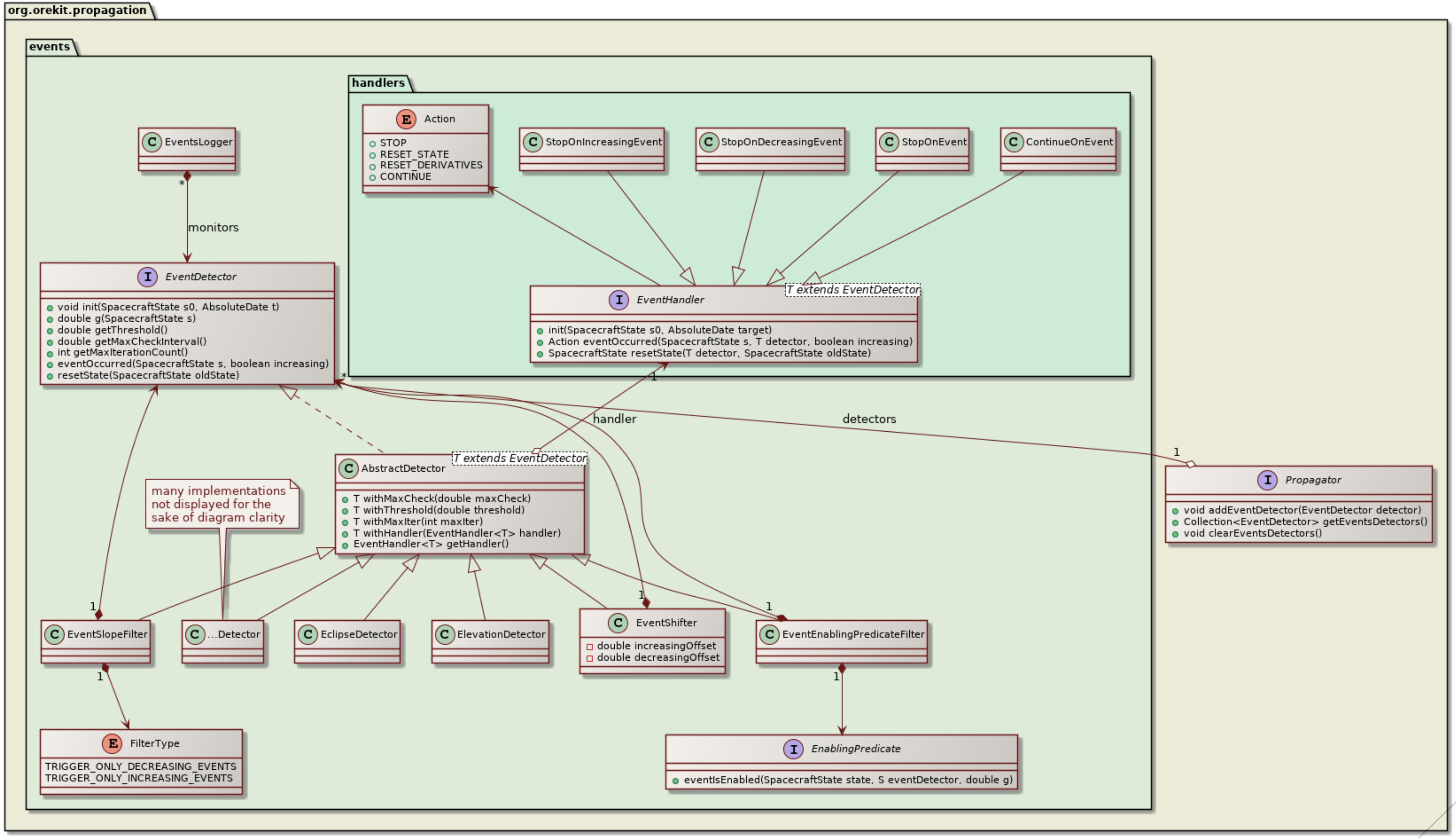
Orekit

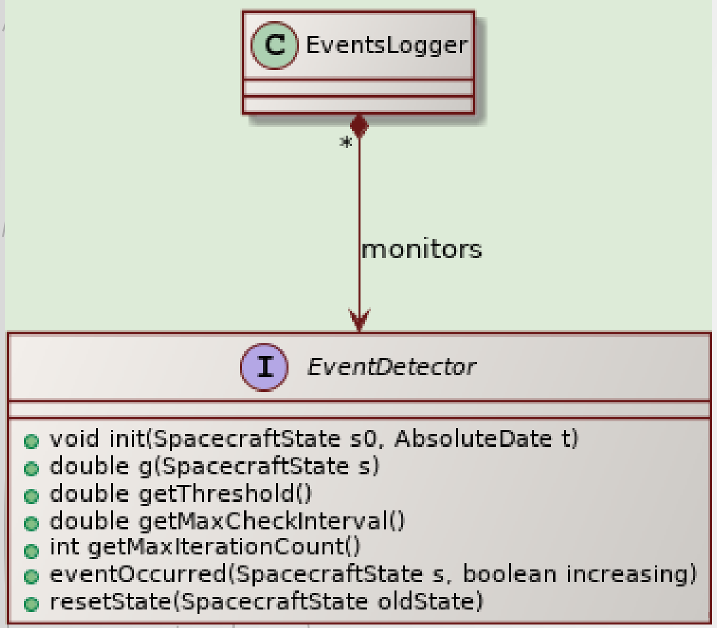
- **What is orekit?**

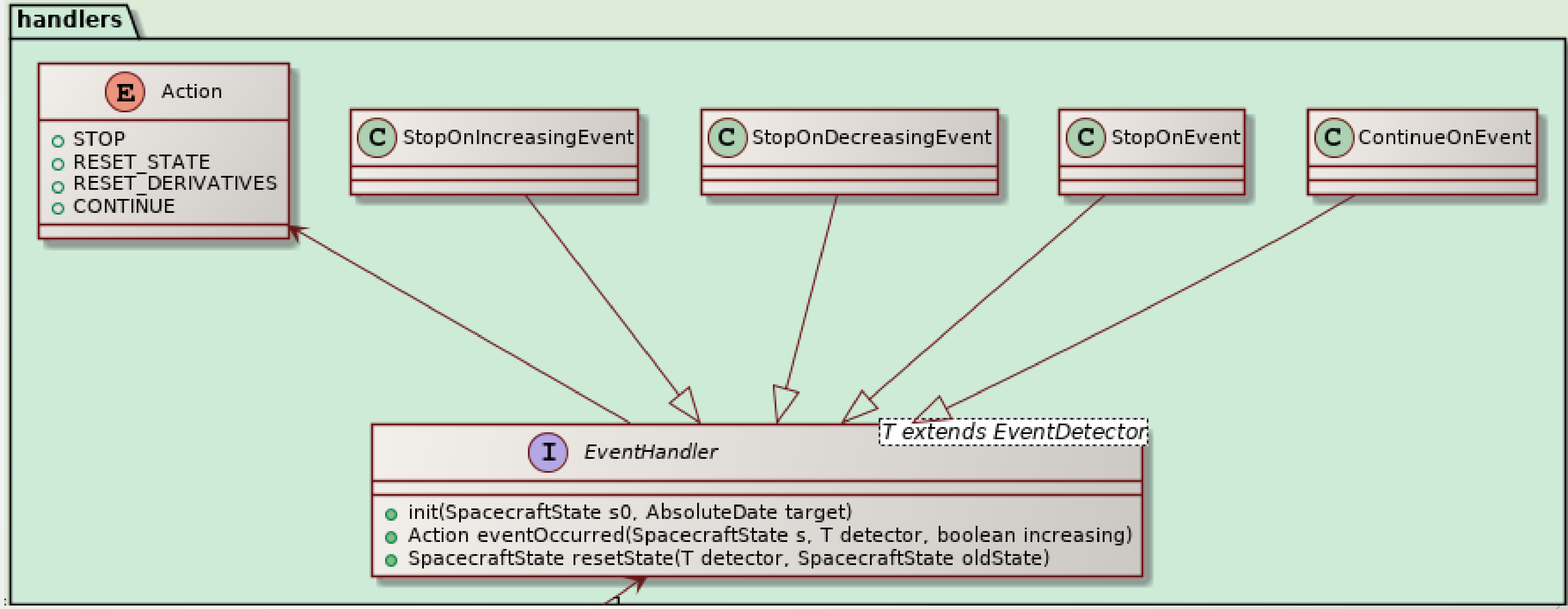
An open source libraries kit

- **How does it work?**

Java language, several packages for each topic


org.orekit.propagation
events
handlers
Action
STOP
RESET_STATE
RESET_DERIVATIVES
CONTINUE
StopOnIncreasingEvent
StopOnDecreasingEvent
StopOnEvent
ContinueOnEvent
EventHandler
T extends EventDetector
init(SpacecraftState s0, AbsoluteDate target)
Action eventOccurred(SpacecraftState s, T detector, boolean increasing)
SpacecraftState resetState(T detector, SpacecraftState oldState)
EventsLogger
monitors
EventDetector
void init(SpacecraftState s0, AbsoluteDate t)
double g(SpacecraftState s)
double getThreshold()
double getMaxCheckInterval()
int getMaxIterationCount()
eventOccurred(SpacecraftState s, boolean increasing)
resetState(SpacecraftState oldState)
handler
detectors
AbstractDetector
T extends EventDetector
T withMaxCheck(double maxCheck)
T withThreshold(double threshold)
T withMaxIter(int maxIter)
T withHandler(EventHandler<T> handler)
EventHandler<T> getHandler()
many implementations
not displayed for the
sake of diagram clarity
Propagator
void addEventDetector(EventDetector detector)
Collection<EventDetector> getEventsDetectors()
void clearEventsDetectors()
EventSlopeFilter
...Detector
EclipseDetector
ElevationDetector
EventShifter
double increasingOffset
double decreasingOffset
EventEnablingPredicateFilter
FilterType
TRIGGER_ONLY_DECREASING_EVENTS
TRIGGER_ONLY_INCREASING_EVENTS
EnablingPredicate
eventIsEnabled(SpacecraftState state, S eventDetector, double g)


EventsLogger
*
monitors
EventDetector
void init(SpacecraftState s0, AbsoluteDate t)
double g(SpacecraftState s)
double getThreshold()
double getMaxCheckInterval()
int getMaxIterationCount()
eventOccurred(SpacecraftState s, boolean increasing)
resetState(SpacecraftState oldState)


handlers
Action
STOP
RESET_STATE
RESET_DERIVATIVES
CONTINUE
StopOnIncreasingEvent
StopOnDecreasingEvent
StopOnEvent
ContinueOnEvent
EventHandler
T extends EventDetector
init(SpacecraftState s0, AbsoluteDate target)
Action eventOccurred(SpacecraftState s, T detector, boolean increasing)
SpacecraftState resetState(T detector, SpacecraftState oldState)

Rayuela:
a program built with orekit’s libraries

What Rayuela does

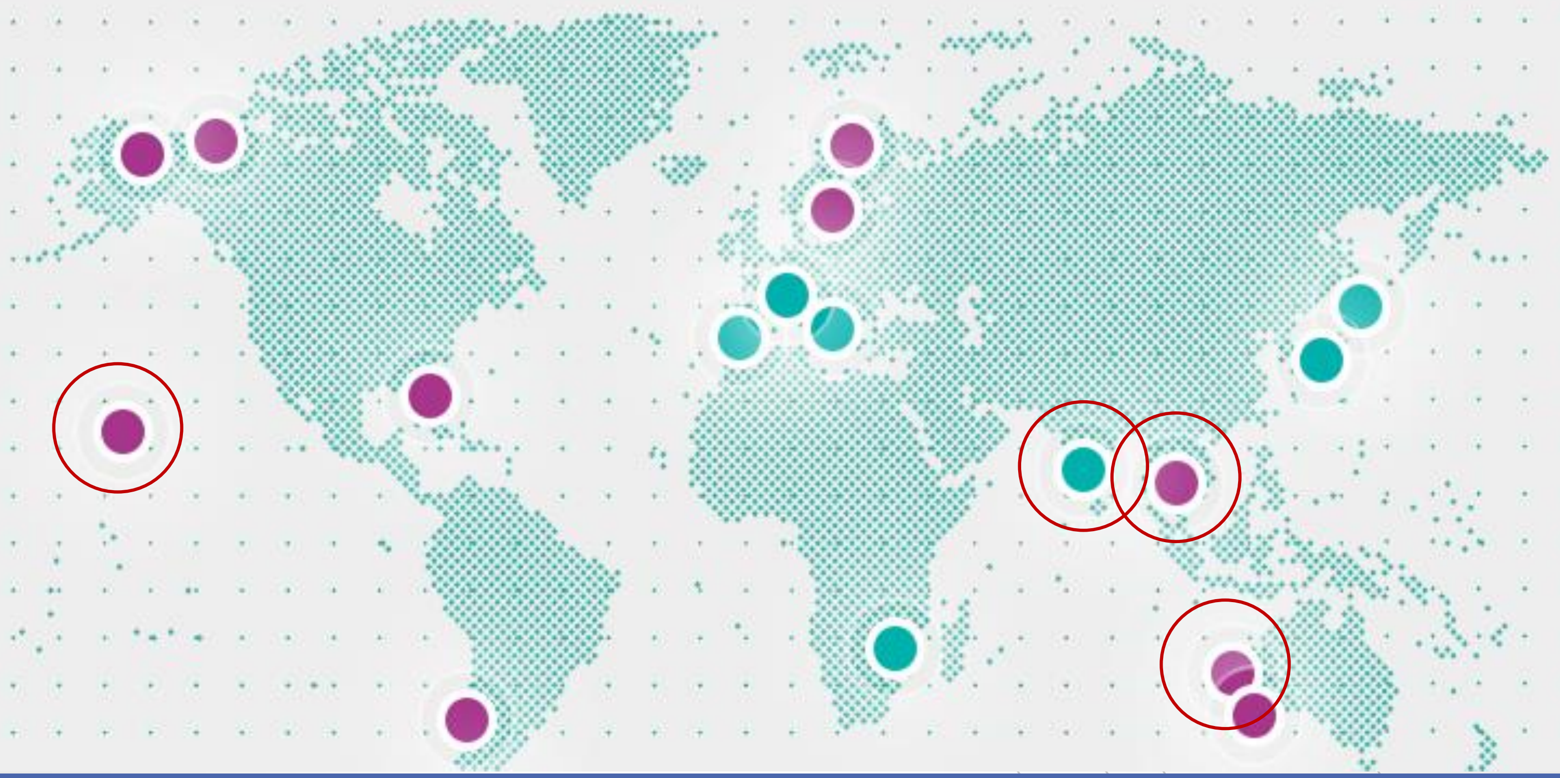
It detects satellite visibility and eclipse events

```java
    // Event definition
    final double maxcheck  = 60; /**maximum checking interval (s)*/
    final double threshold =  0.001; /**maximum divergence threshold (s)*/
    final double elevation = FastMath.toRadians(5.0);/**constrain angle*/

    final EventDetector sta1Visi =
            new ElevationDetector(maxcheck, threshold, sta1Frame)
            .withConstantElevation(elevation).
            withHandler(new VisibilityHandler())
            ;

    final EventDetector sta2Visi =
            new ElevationDetector(maxcheck, threshold, sta2Frame)
            .withConstantElevation(elevation).
            withHandler(new VisibilityHandler())
            ;

   final EventDetector sta3Visi =
            new ElevationDetector(maxcheck, threshold, sta3Frame).
            withConstantElevation(elevation).
            withHandler(new VisibilityHandler());

    final EventDetector sta4Visi =
            new ElevationDetector(maxcheck, threshold, sta4Frame).
            withConstantElevation(elevation).
            withHandler(new VisibilityHandler());

   final EventDetector overLap =  BooleanDetector.orCombine(sta1Visi,sta2Visi).withHandler(new   MyHandler());
```

How it works

Managing events: EventDetector and EventHandler

```java

 /** Handler for visibility event. */
private static class VisibilityHandler implements EventHandler<ElevationDetector> {
    private AbsoluteDate start = null;
    public Action eventOccurred(final SpacecraftState s, final ElevationDetector detector,
                                final boolean increasing) {


        if (increasing) {

             start = s.getDate();



        } else if(start !=null)  {
            double duration = s.getDate().durationFrom(start);
           System.out.println(s.getDate()+"      " + duration);

        }
return Action.CONTINUE ;


                }
  }
```

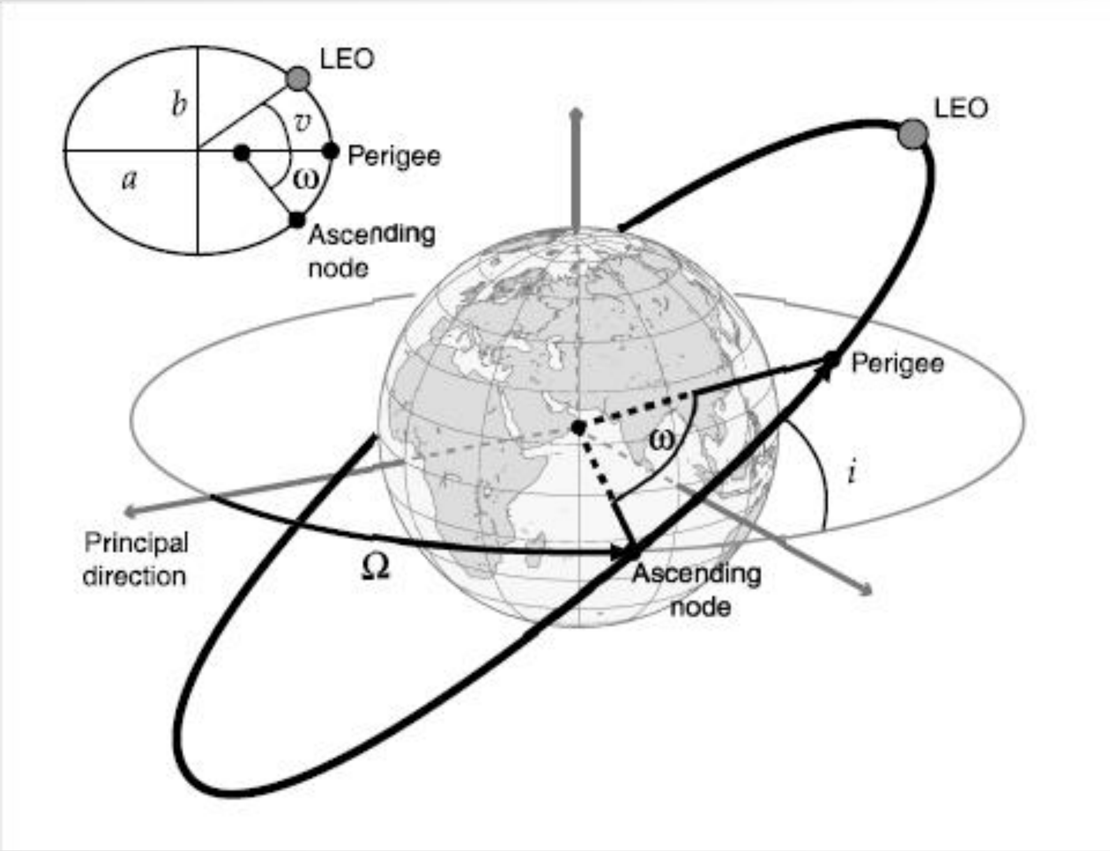


GROUND STATION NETWORK

Very different latitude between Australia’s station and the others

Overlap between India and Thailand

Orbit parameters



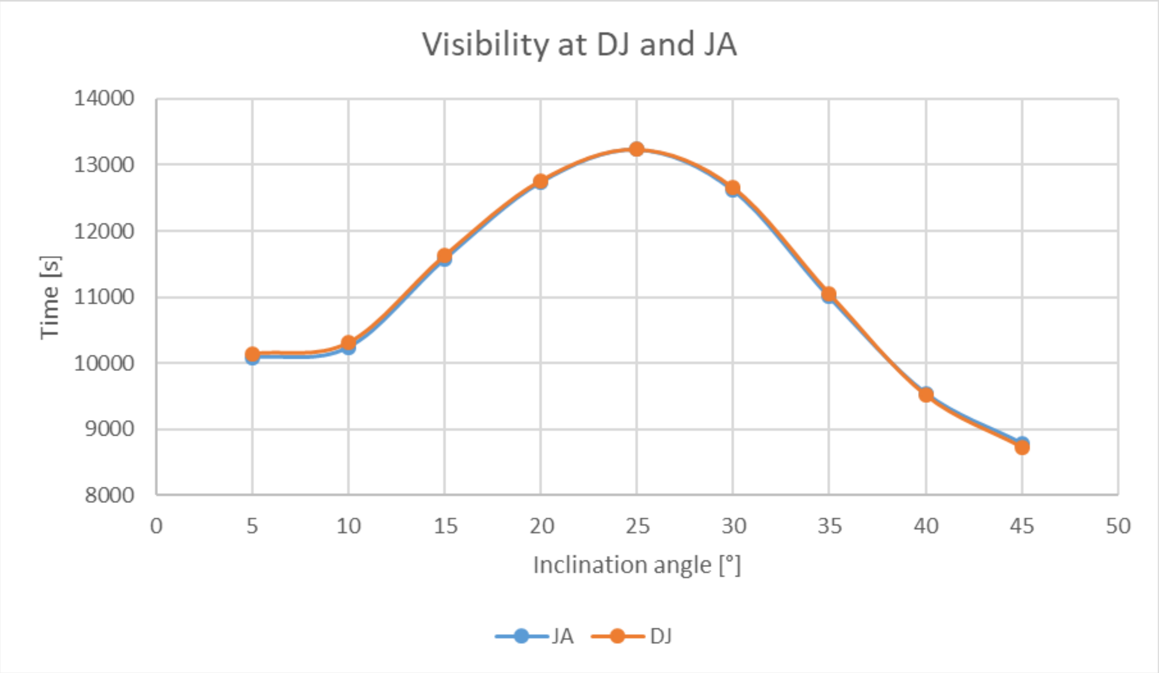
- **a** = semimajor axis
- **e** = eccentricity
- **i** = inclination
- **ν** = true anomaly
- **ω** = argument of perigee
- **RAAN** = right ascension o the ascending node

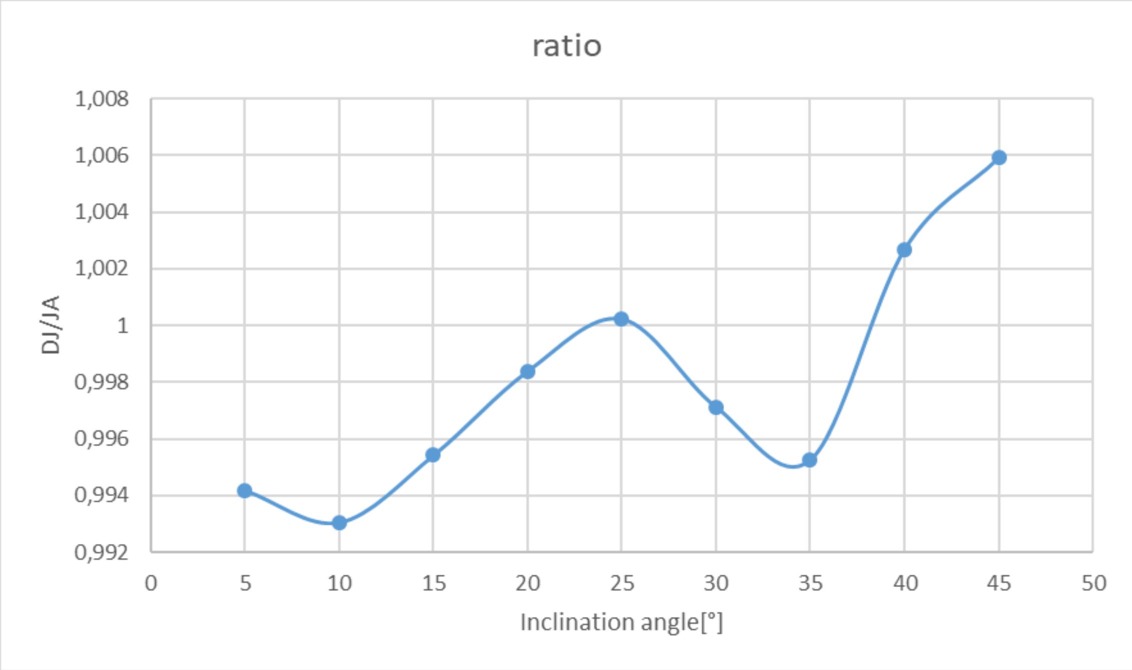
My LEO

- **Altitude** = 500 km
- **e** = 0

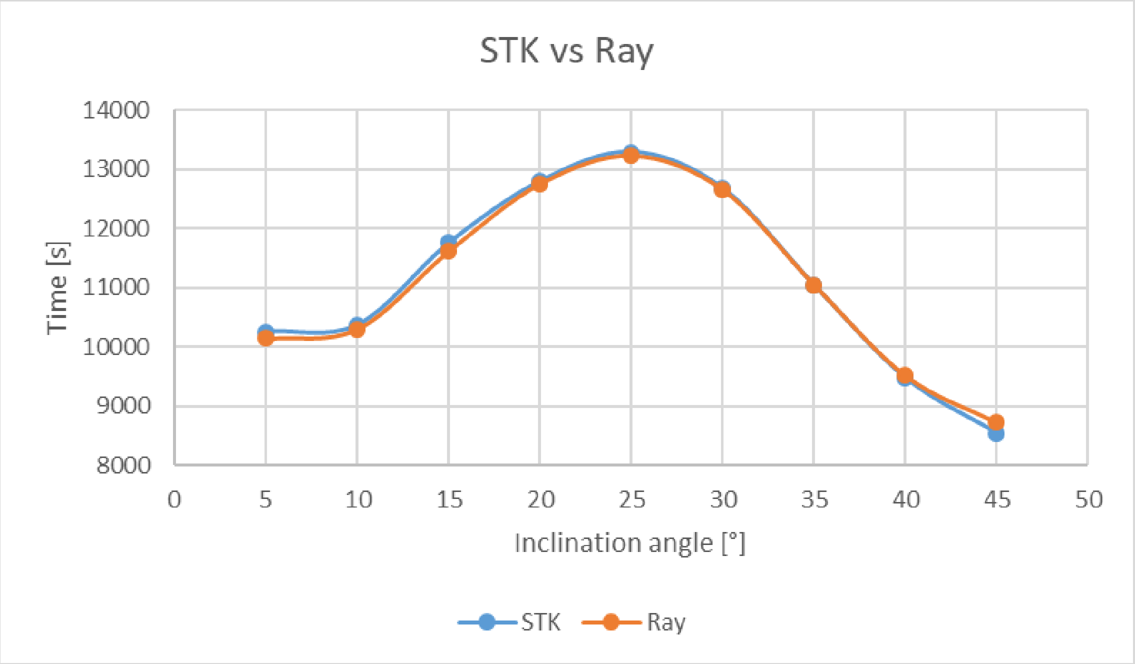
Average time of visibility per day in one month with varying inclination

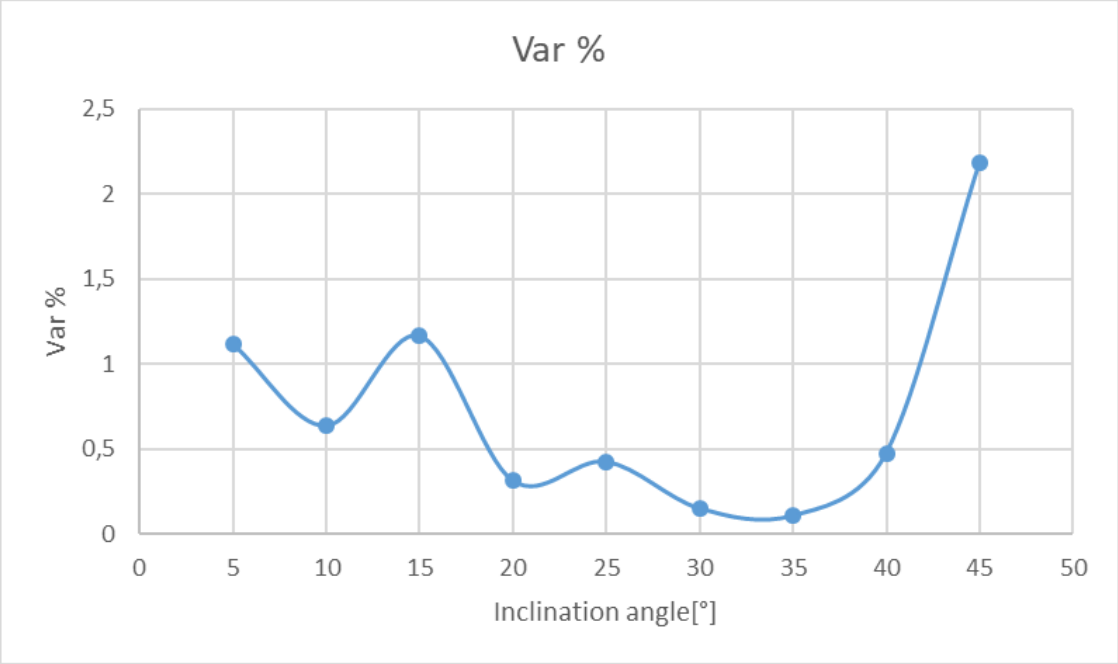
- DJ = December-January
- JA = July-August





Validation with STK

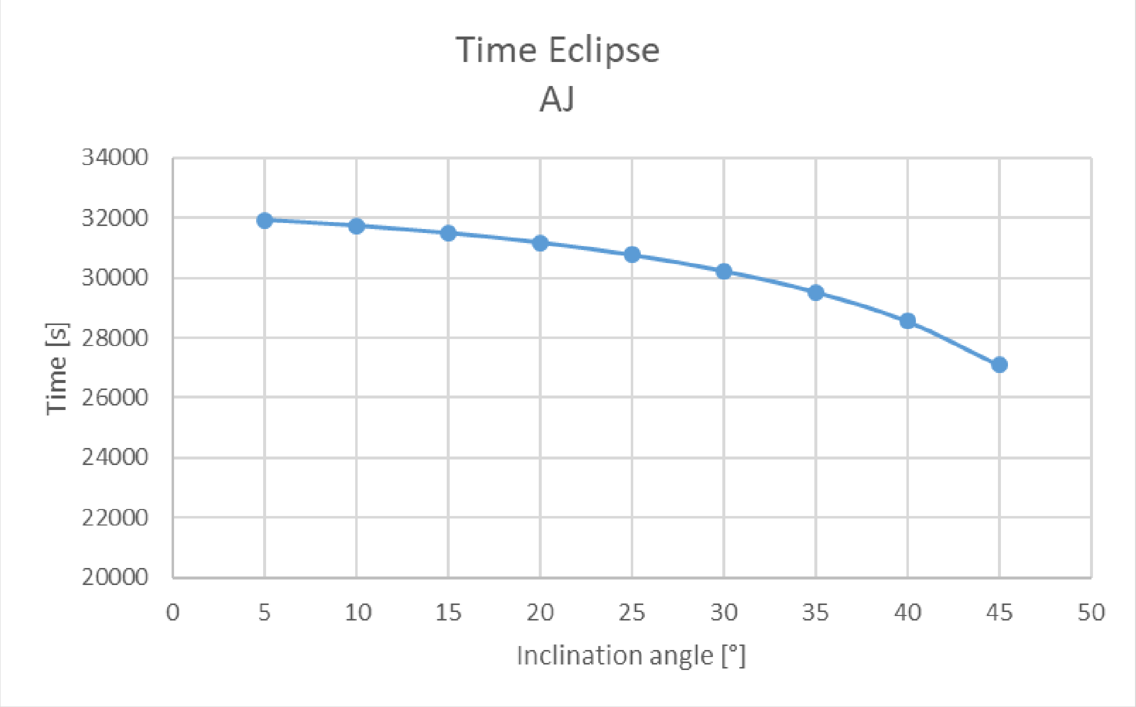


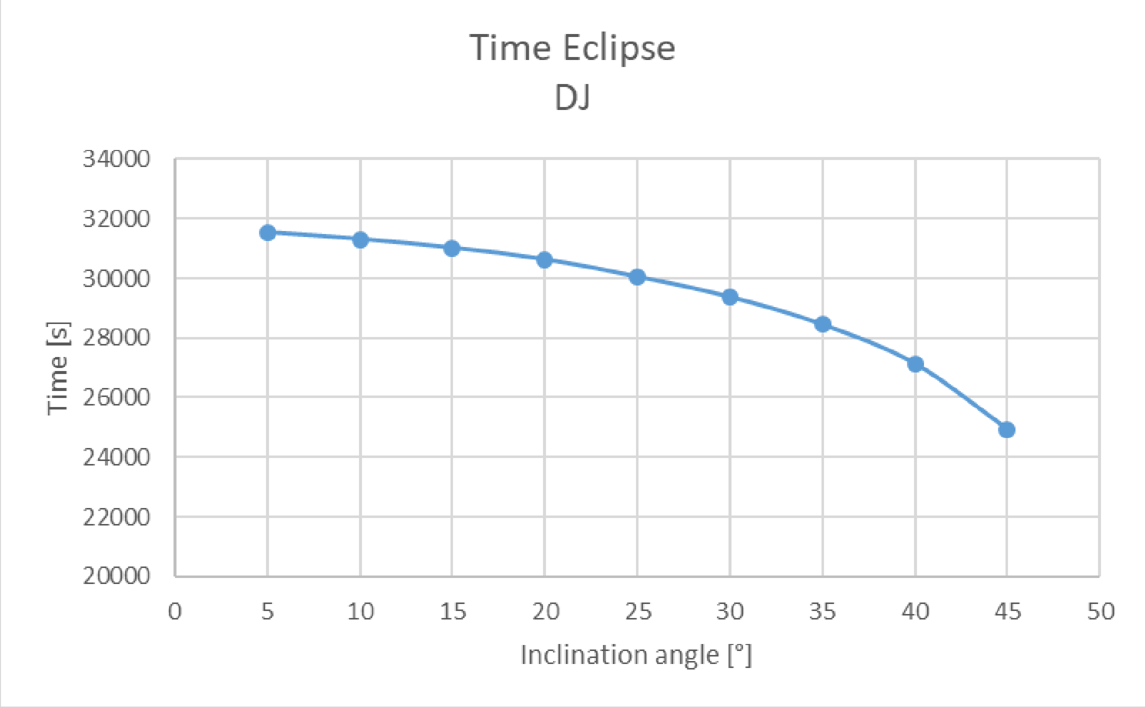


Eclipse events

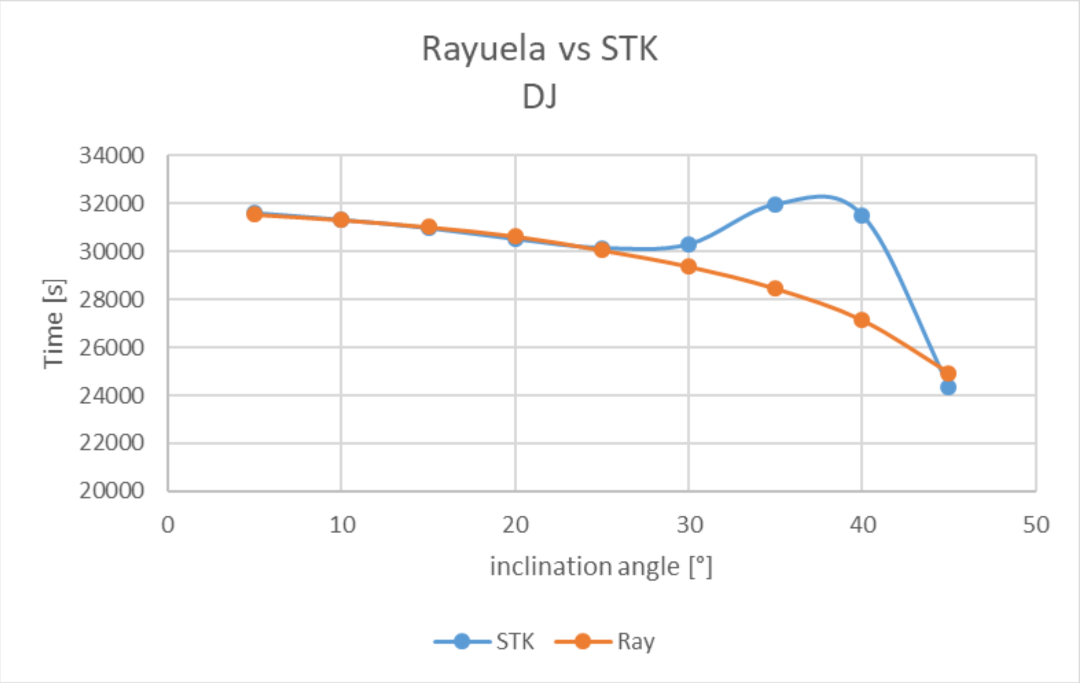
Average time of eclipse during one day, with varying inclination

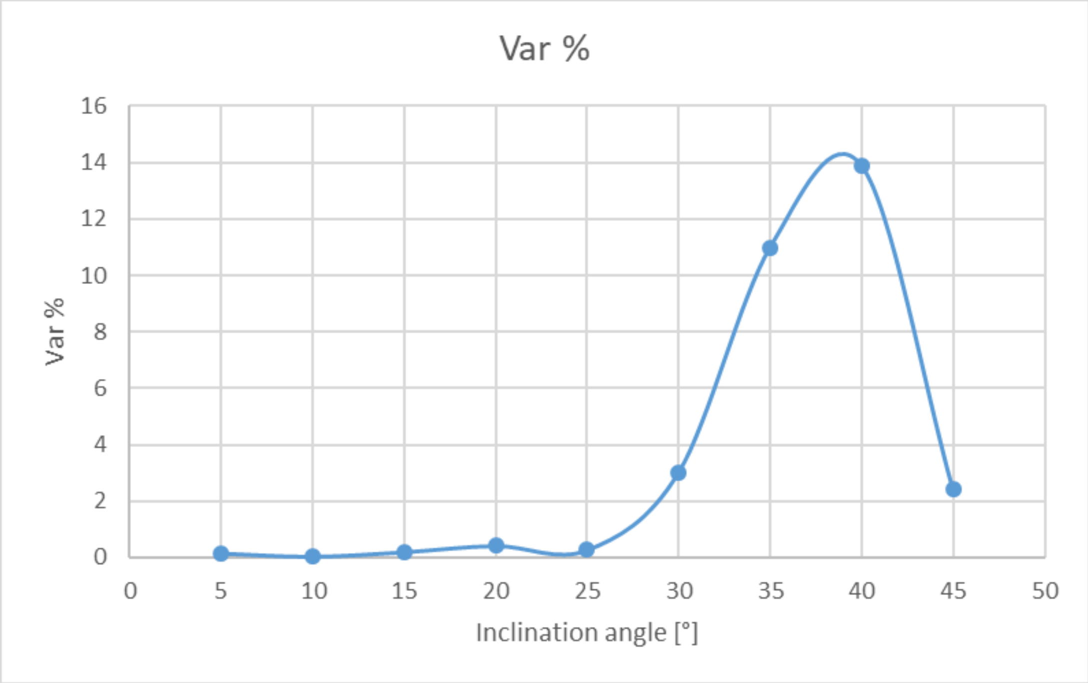
- DJ = December-January
- JA = July-August



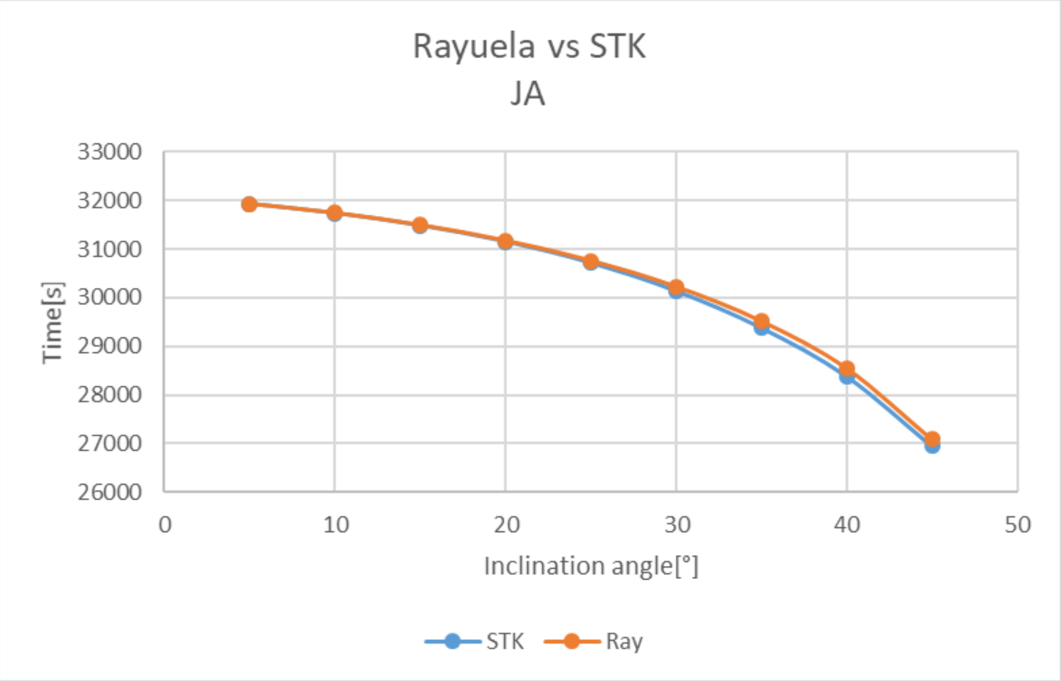


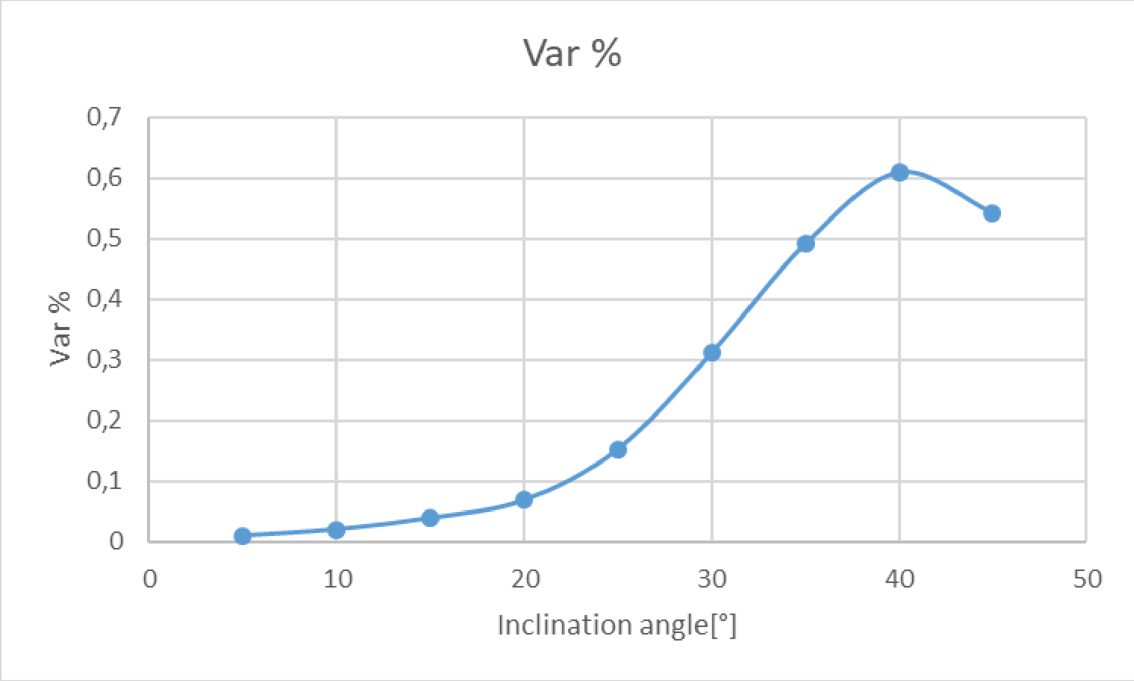
Validation with STK





Validation with STK





Orbit optimization

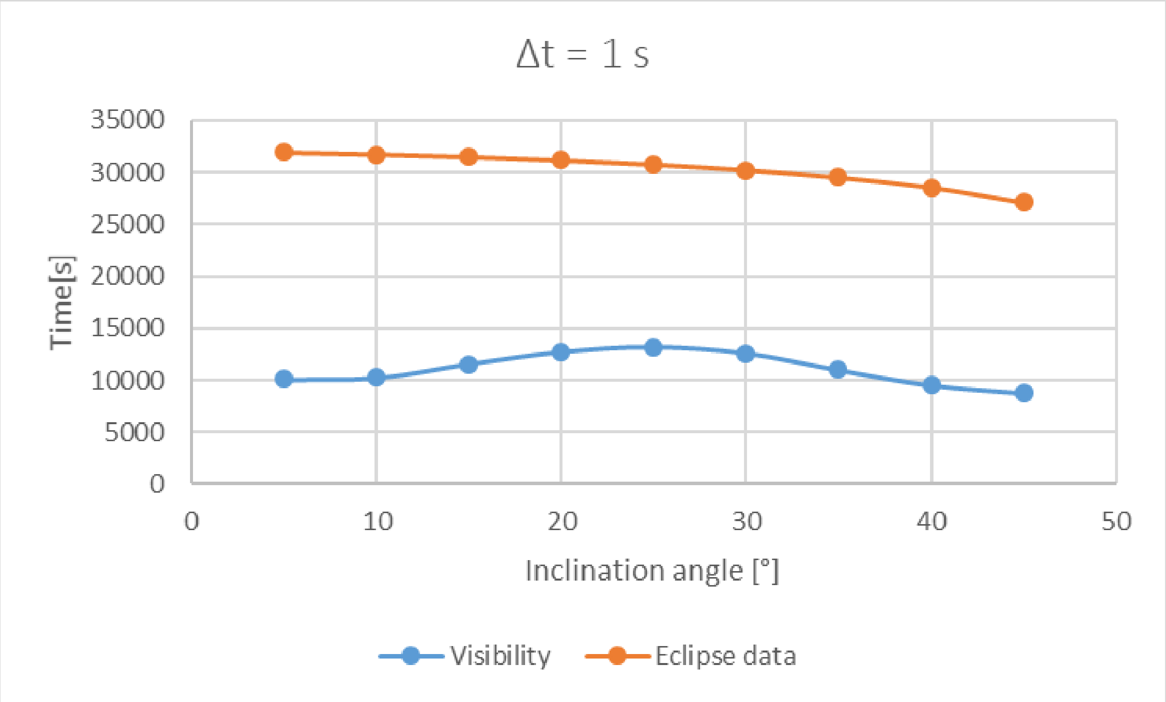
Two paths

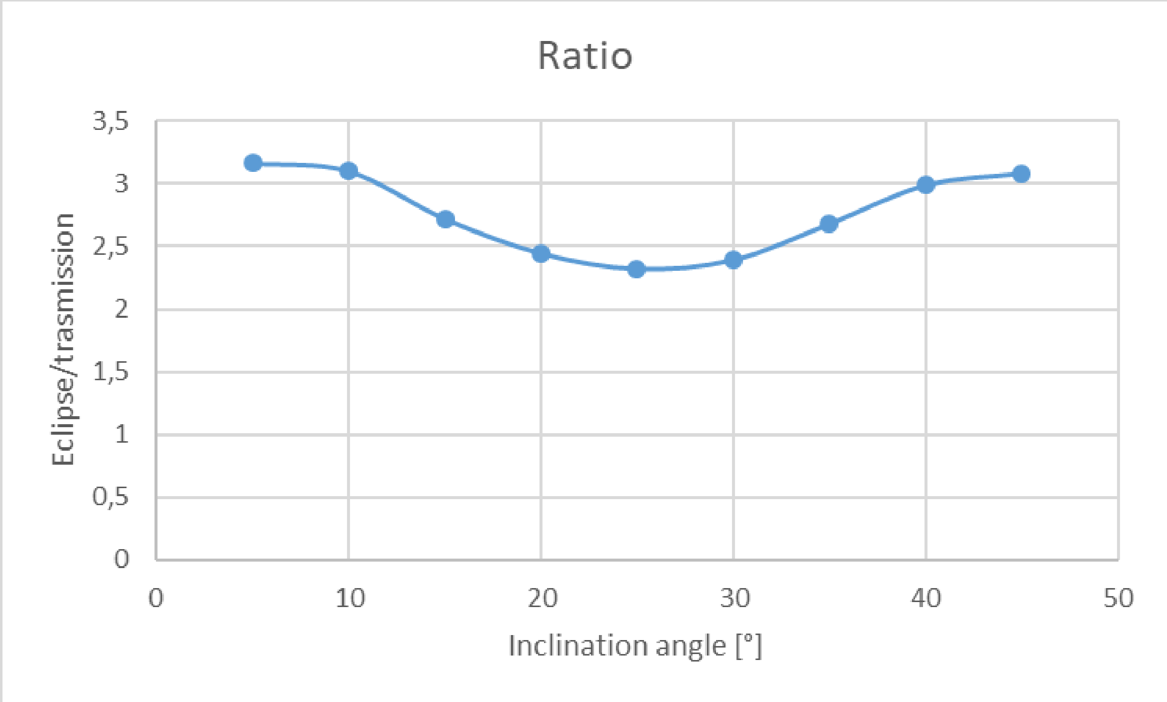
- Maximization of time for data acquisition

- Maximization of time to trasmission data

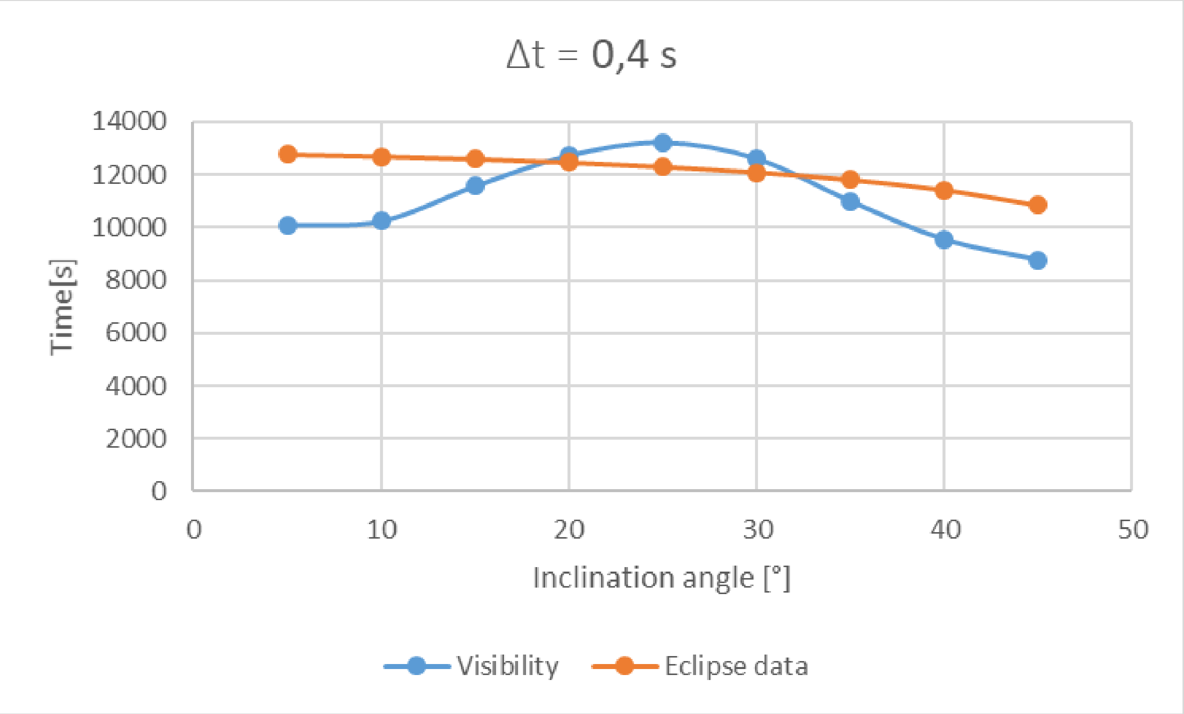
$$T_e \cdot \frac{\Delta t_{TRAS}}{1\ s} < T_{VIS}$$

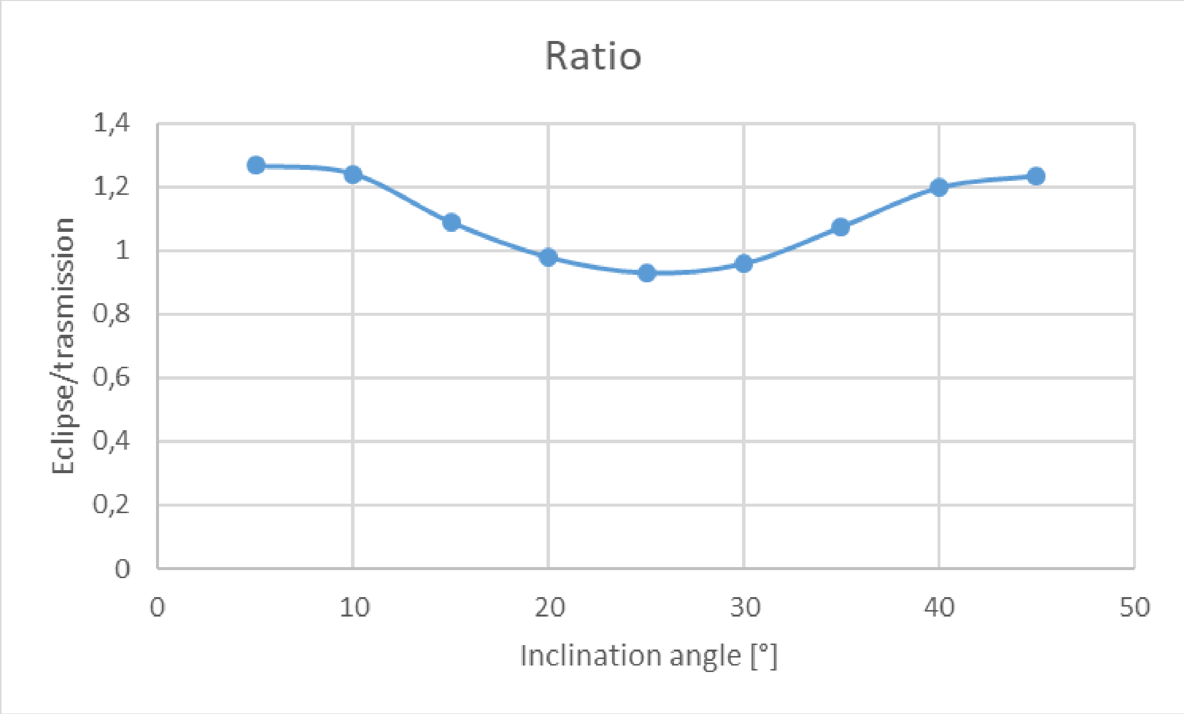
- T_e = Eclipse time
- Δt_{TRAS}= Trasmission time for each second
- T_{VIS} =Visibility time



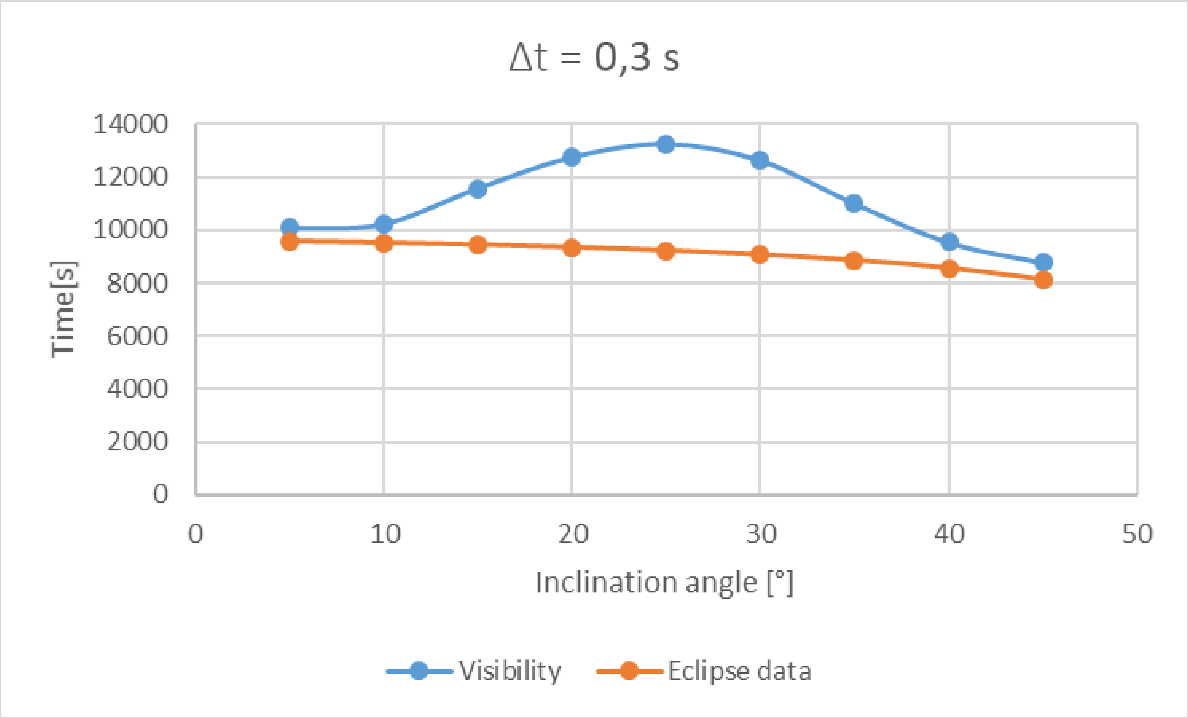


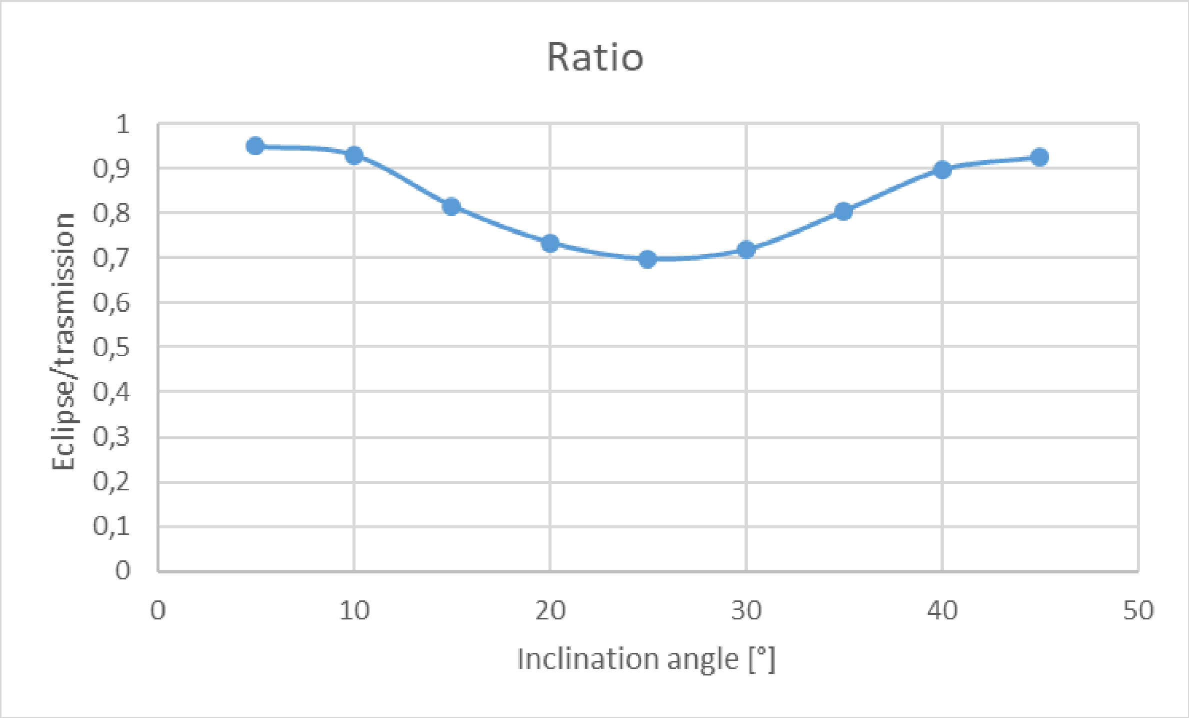
Extreme case





The best configuration





Final considerations

- Rayuela is comparable with STK
- Eclipse events need a deeper investigation
- At $\Delta t_{TRAS} \leq 0{,}3\ s$ inclinations maximize data acquisition

Special thanks

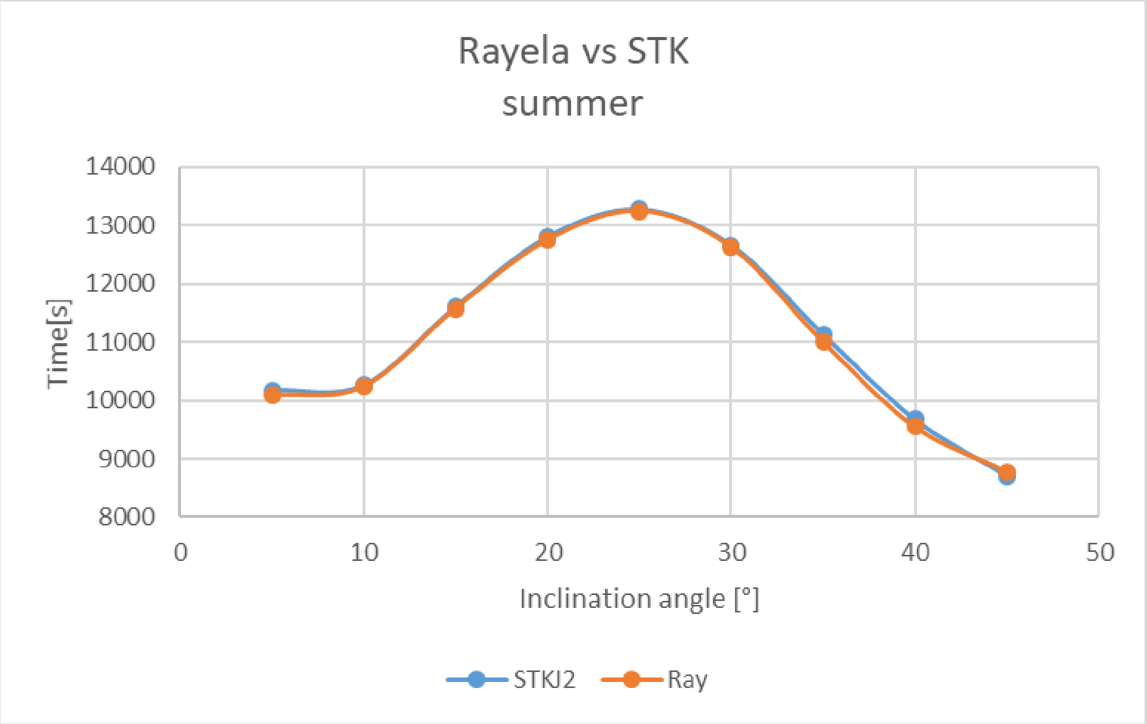
- Professor Mario Bertaina
- Professoressa Raffella Bonino
- Daniela Borla Tridon
- Ilenya Salvoni

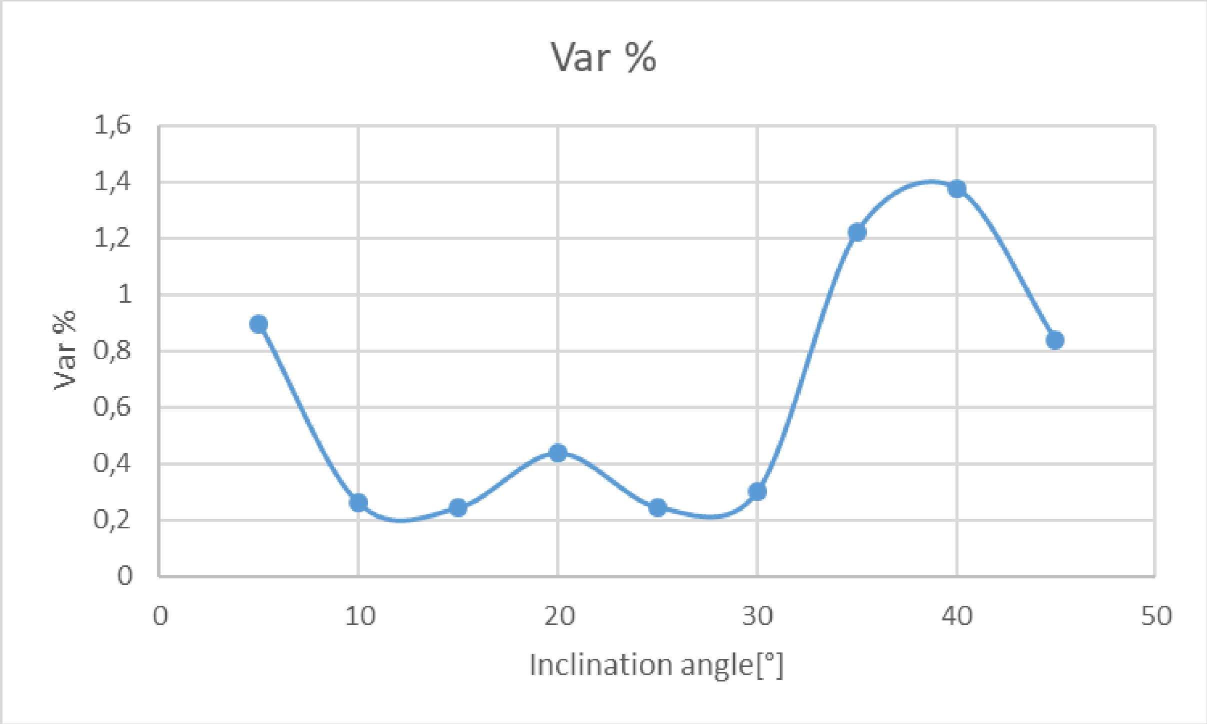
And also

- Roberto Belcastro Marasco

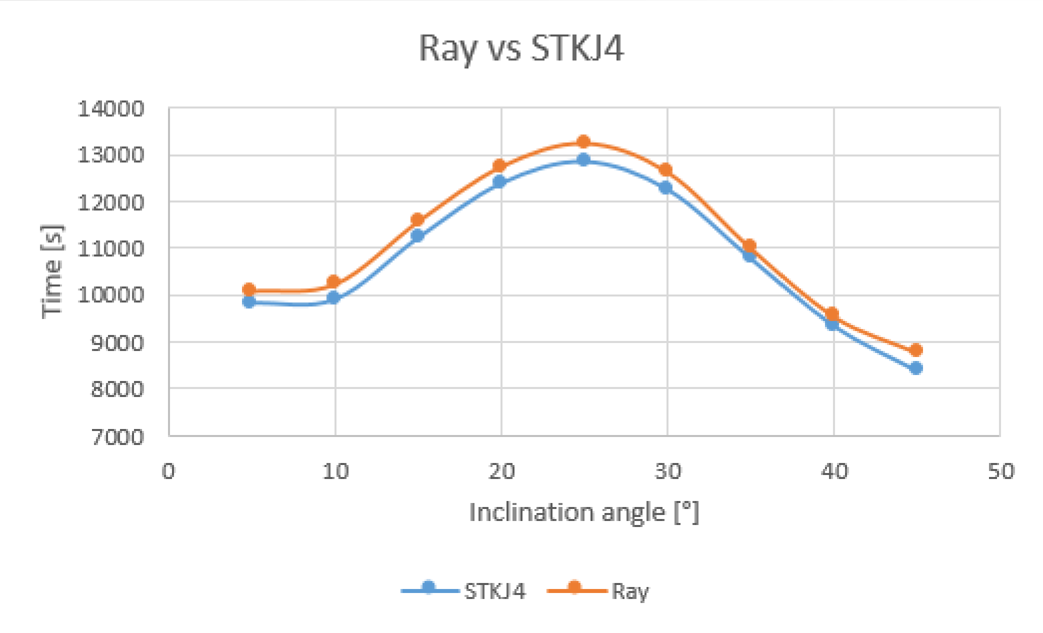
Backup

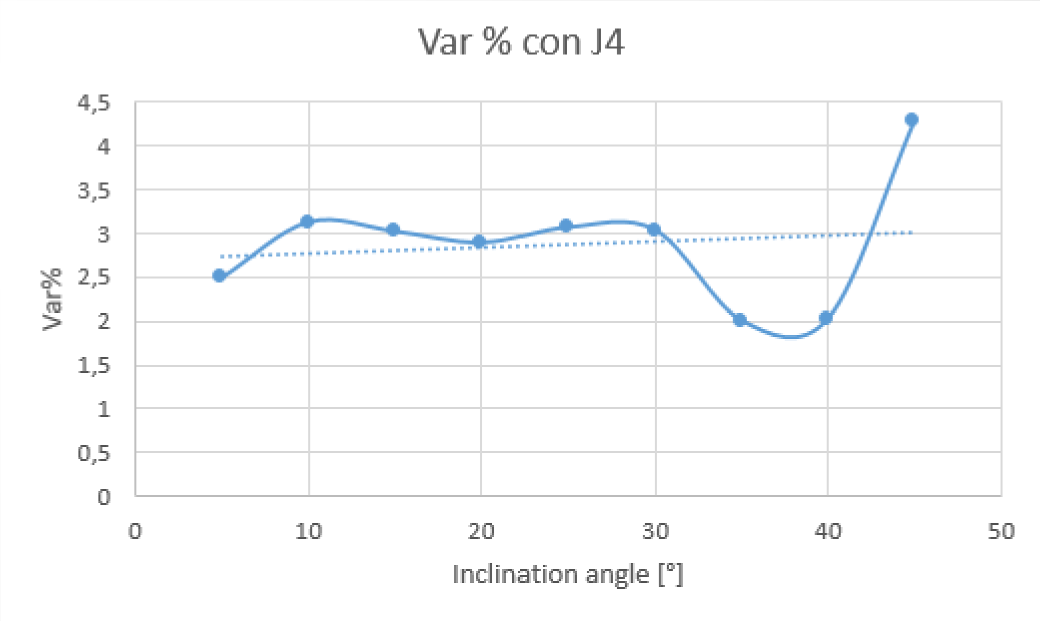
Validation with STK



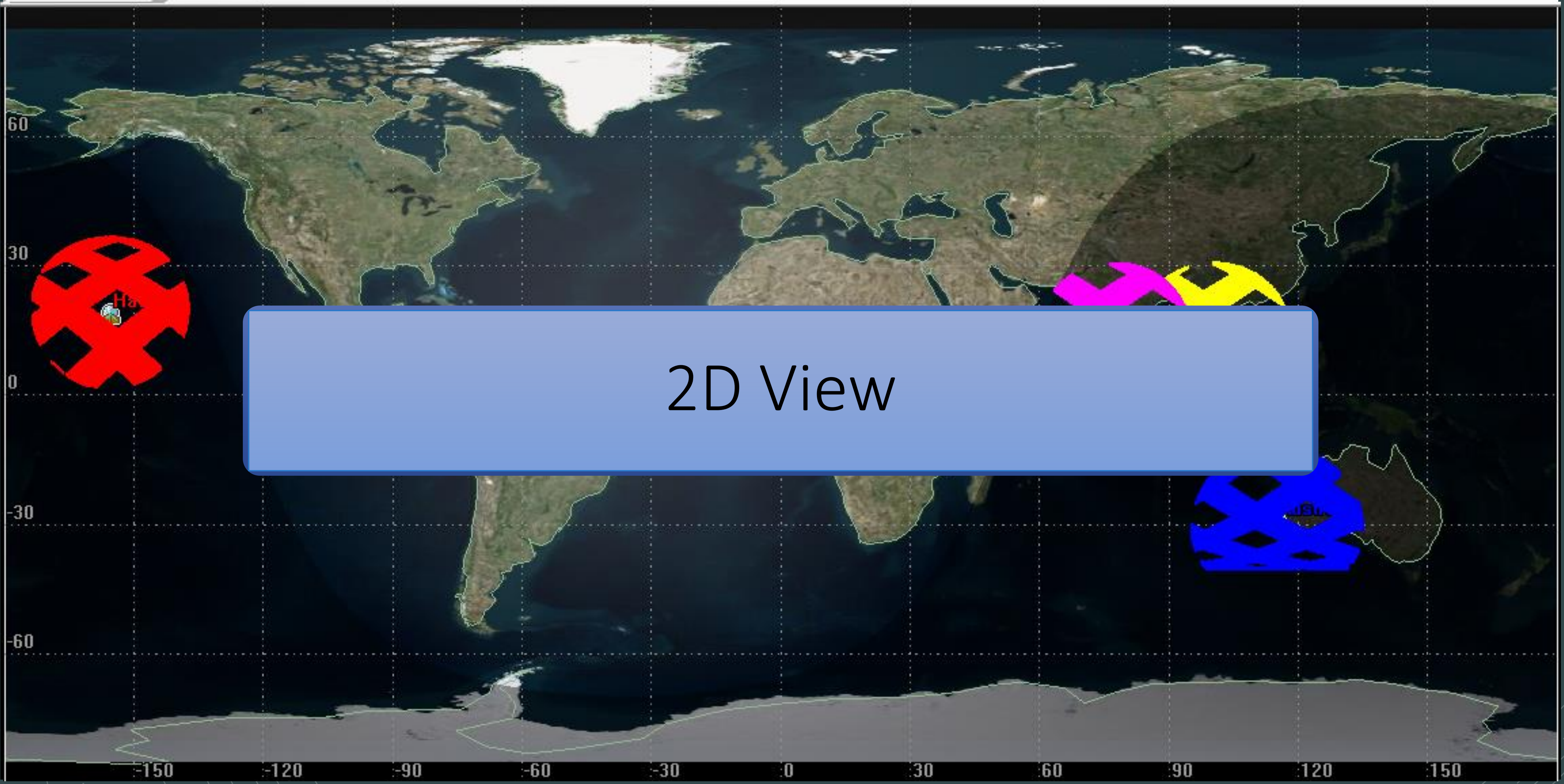


With another propagator (STKJ4)

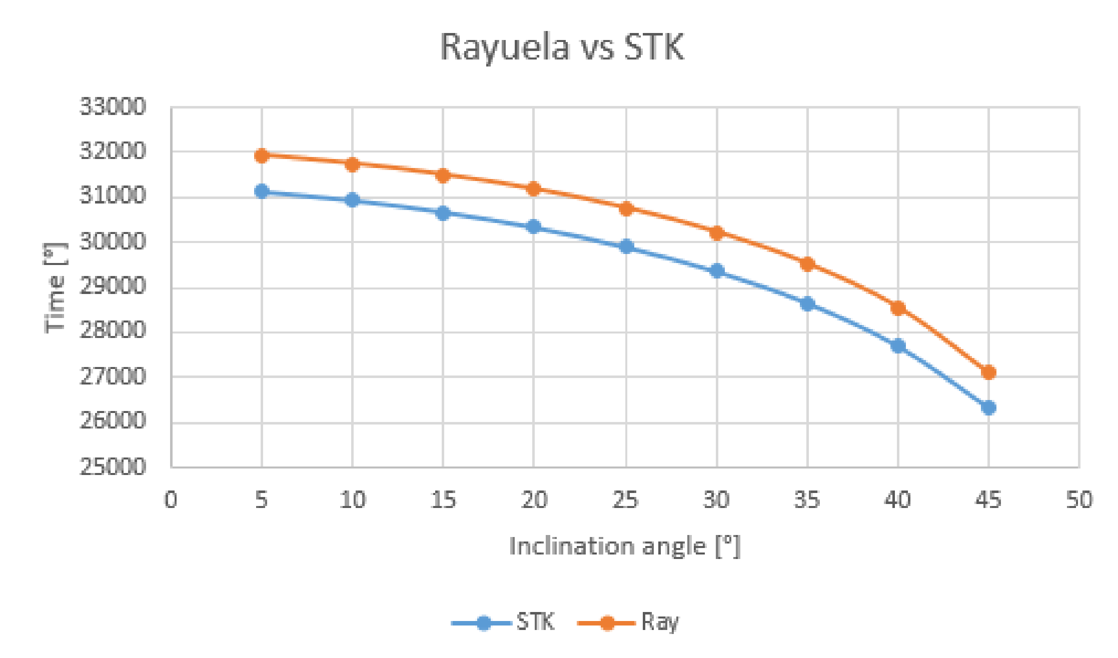


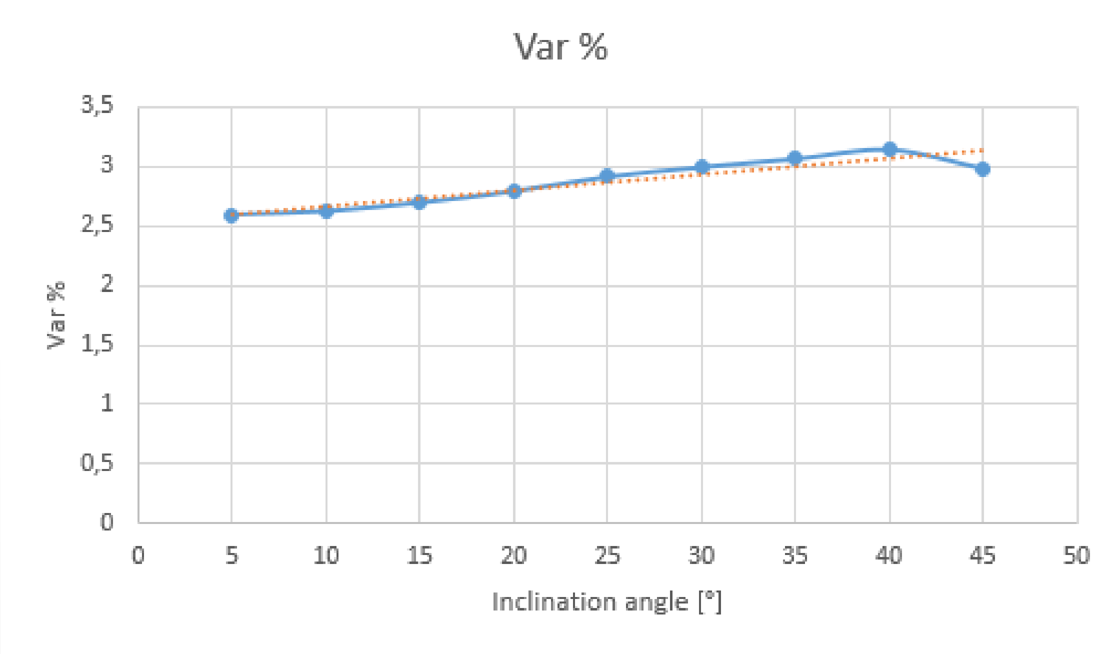


2D View



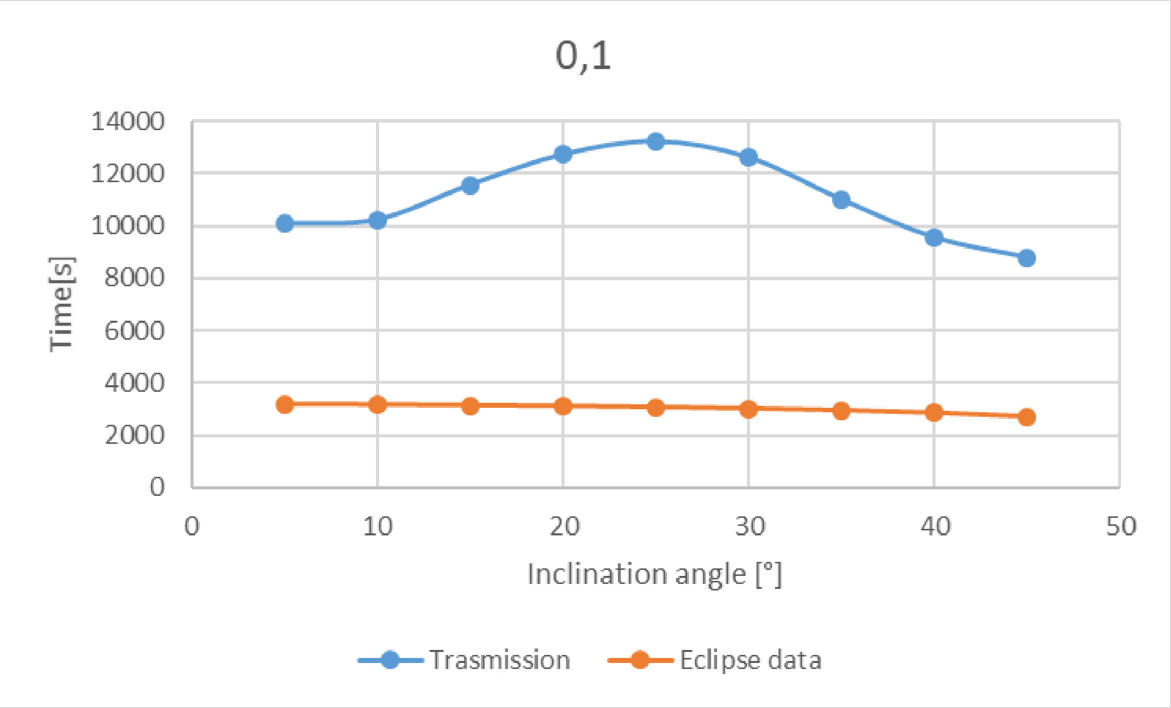
With another propagator (STKJ4)

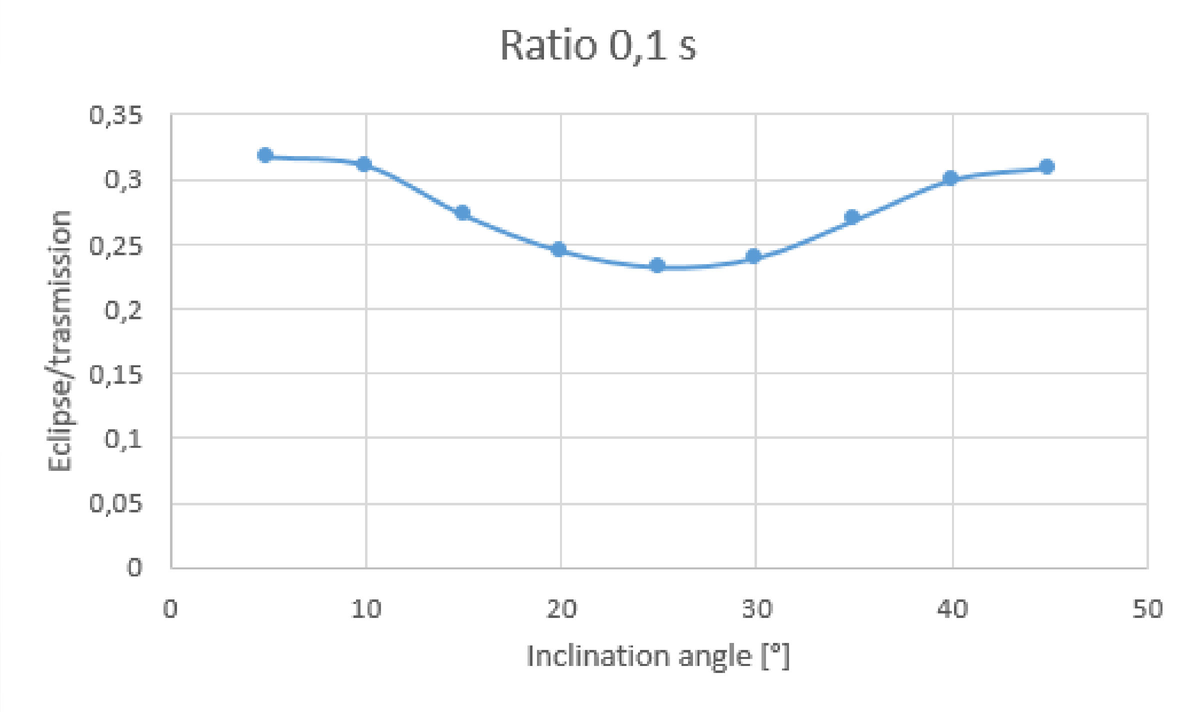




Critical issues

- Overlap between India and Thailand
- Very different latitude between Australia’s station and the others


0,1
14000
12000
10000
8000
6000
4000
2000
0
Time[s]
0
10
20
30
40
50
Inclination angle [°]
Trasmission
Eclipse data


Ratio 0,1 s
0,35
0,3
0,25
0,2
0,15
0,1
0,05
0
Eclipse/trasmission
0
10
20
30
40
50
Inclination angle [°]