Orekit Open Source -kirjasto meteoriparvien partikkelimallinnuksessa

Aurinkokuntatapaaminen 2022 verkkotapahtuma, Ursan Meteoriryhmä, Markku Nissinen

Esityksen sisältö

- Yleistä asiaa Orekit -kirjastosta.
- Ohjelmaesimerkki: kuinka Orekit -kirjastoa käytetään radan laskennassa. Esimerkki on Orekit kotisivulta ja siinä näytetään kuinka satelliitin rata lasketaan Orekit- kirjastolla tilanteessa, jossa se lähtee Maan ja Kuun L2 pisteestä. Konversiota ei ole vielä tehty Esko Lyytisen mallille, joten sen vuoksi esitän tässä nyt yleisen esimerkin.
- Yksityiskohtainen selostus Esko Lyytisen partikkelimallista. Perusmalli sekä ns. parannettu malli.
- Esimerkkeinä Leonidit ja Perseidit.

ORbit Extrapolation KIT 11.0.2

https://www.orekit.org

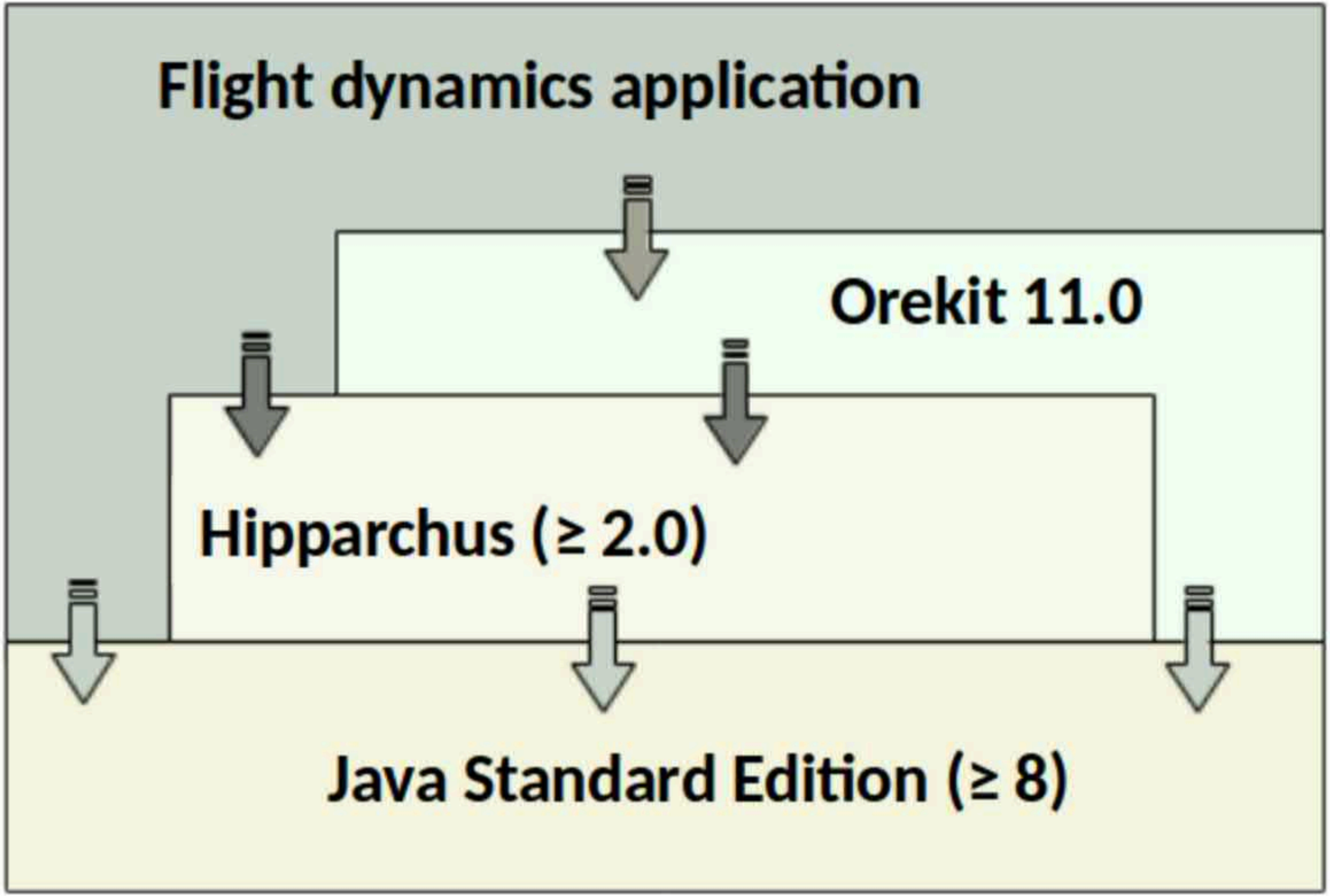

Orekit
An accurate and efficient core layer for space flight dynamics applications
Orekit 11.0.2 released, get it!

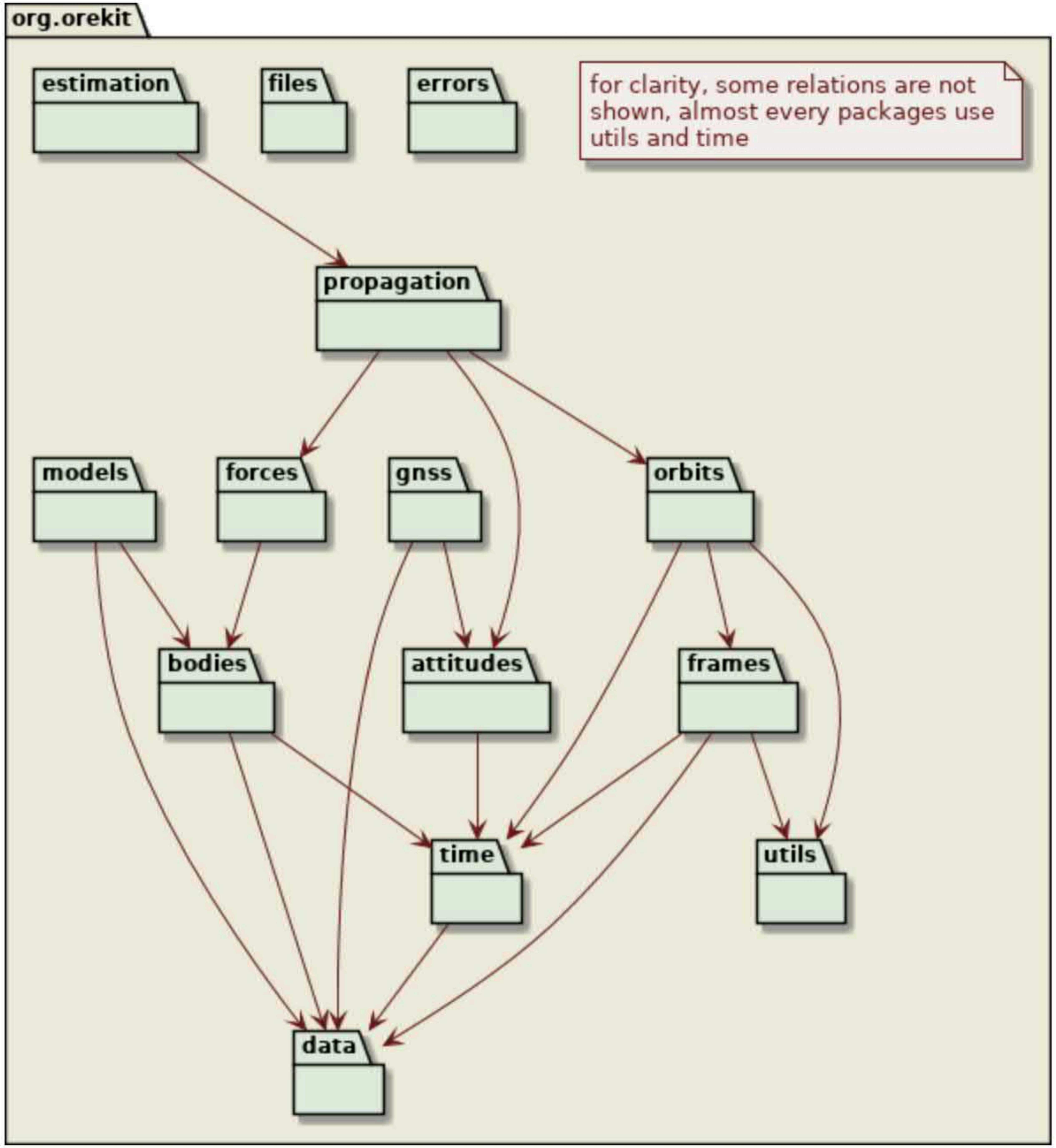

Rugged
A sensor-to-terrain mapping tool
Rugged 2.2 released, get it!


Stavor
A spacecraft 3D attitude visualization tool for mobile devices
Stavor 2.2 released, get it!

- Javalla tehty matalan tason Open Source kirjasto.
- Lisenssi Apache version 2.0
- Tämän esityksen Orekit materiaalit on Orekitin kotisivulta. Creative Commons Attribution 3.0 lisenssi
- Kehitysympäristö esimerkiksi Eclipse. Maven on käytössä, mutta se täytyy ottaa pois päältä, jos muokkaa kirjastoa ja käyttää muokattua koodia paikallisesti.
- Orekitille on myös Python Wrapper, jonka on tehnyt Petrus Hyvönen, Swedish Space Corporation

- Kehittäjänä CS GROUP. Aloitettu kehitys vuonna 2002.
- Käyttäjinä Airbus Defence and Space
- U.S. Naval Research Laboratory
- Swedish Space Corporation
- Thales Alenia Space
- European Space Agency
- EUMETSAT


Flight dynamics application
Orekit 11.0
Hipparchus (≥ 2.0)
Java Standard Edition (≥ 8)



org.orekit
estimation
files
errors
for clarity, some relations are not shown, almost every packages use utils and time
propagation
models
forces
gnss
orbits
bodies
attitudes
frames
time
utils
data

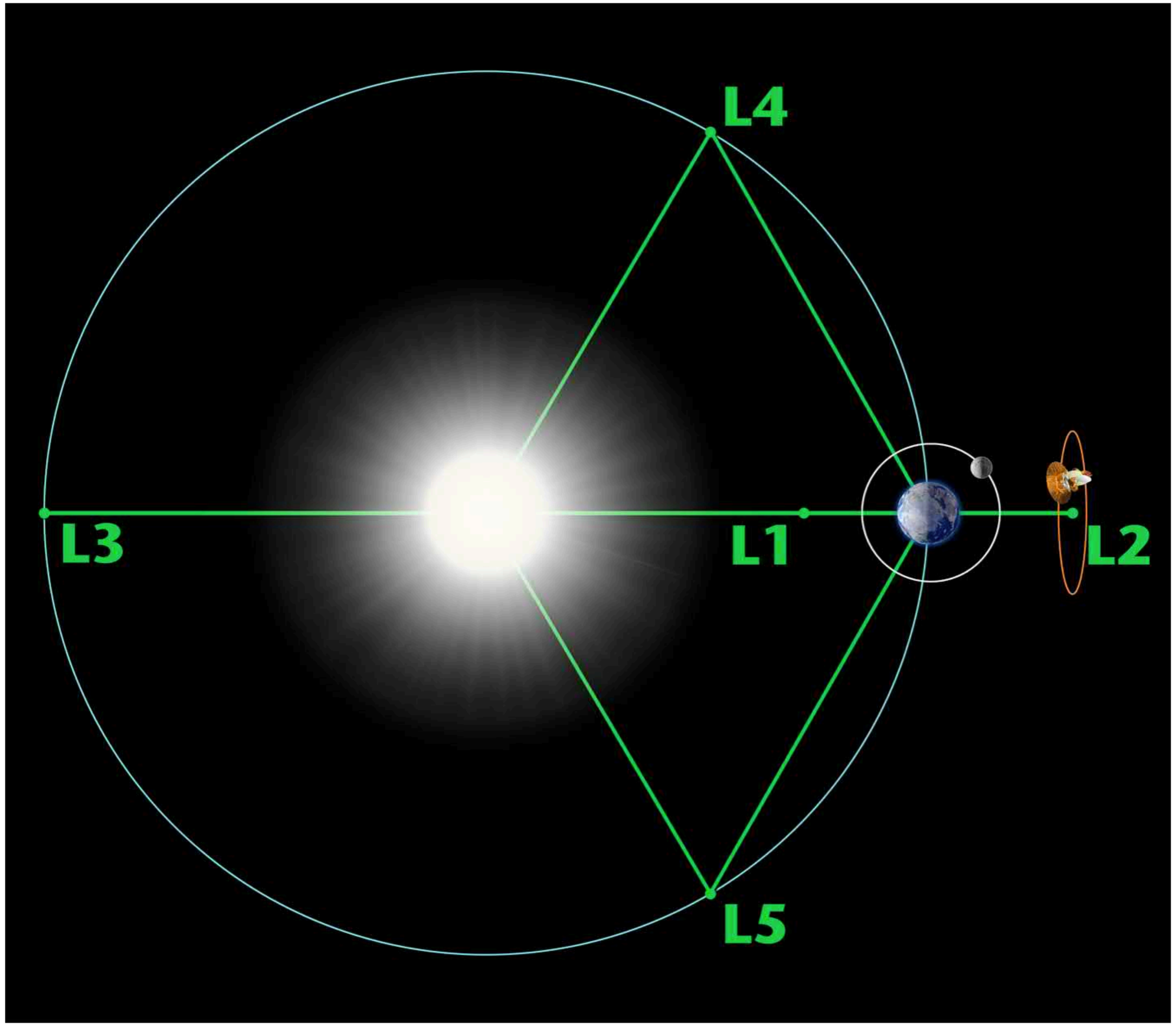
Tutorials

Java tutorials, by the Orekit team

The tests suite is the first place to look for getting help with Orekit use. Several tutorials are also provided with the source code in order to emphazise particular use cases:

- **Attitude**
  This tutorial emphasizes a specific usage of the attitude package described in the attitudes section of the library architecture documentation.
- **Frames**
  This tutorial shows how to solve three problems: (1) computation of the Doppler effect of a satellite with respect to a ground station, (2) computation of the position and velocity of the center of gravity in the EME2000 inertial frame and (3) plot some of the effects on the motion of a spacecraft which attitude is guided by a yaw steering law.
- **Propagation**
  This tutorial details some elementary usages of the propagation package described in the propagation section of the library architecture documentation (propagation modes, events management).
- **Propagation in non-inertial frame**
  This tutorial aims to introduce orbital integration using SingleBodyAttraction and InertialForces classes, which allowing the user to compute the motion of a satellite in a reference frame that is not necessarily centered on the main attractor and does not necessarily possess inertial axis.
- **Time**
  This tutorial shows how to handle time in Orekit.
- **Geomagnetic Field Model**
  This tutorial shows how to calculate the geomagnetic field of the earth.
- **Integration in other languages**
  The Orekit library can be accessed from other languages, for integration into existing software or for quick prototyping in a scripting language such as Matlab, Python or Jython. This allows for direct interaction with the different objects, either from a script or directly at the command prompt, and quick access to plotting tools.

What is a Lagrange Point?



Source: NASA/WMAP Science Team

Published: March 27, 2018

Propagation in non-inertial Frame

The goal of this tutorial is to introduce orbital integration using `SingleBodyAttraction` class. This class can replace all the different kinds of point mass interactions (`ThirdBodyAttraction`, `NewtonianAttraction`). Using `SingleBodyAttraction` and `InertialForces` enables a richer modelling, allowing the user to compute the motion of a satellite in a reference frame that is not necessarily centered on the main attractor and does not necessarily possess inertial axis.

Initialization

We will propagate the trajectory of a satellite departing from the Lagrange point L2 of the Earth-Moon system. The equations of motion will be computed in a reference Frame centered on L2, its X-axis continuously oriented toward Earth and Moon. The rotation of this frame means that we cannot simply use the fundamental principle of dynamics (Newton's second law). Since we will consider only the gravitational attractions of the Earth and the Moon, the inertial reference frame most closely related to our problem is a frame that is centered on the Earth-Moon barycenter, with inertial axis.

Let's initialize our program. First, the time settings : the initial moment of integration, the length of integration (here in seconds), and the time interval between each output.

```
final AbsoluteDate initialDate = new AbsoluteDate(2000, 01, 01, 0, 0, 00.000, TimeScalesFactory.getUTC());
double integrationTime = 600000.;
double outputStep = 600.0;
```

Then the initial position and velocity of the satellite, located exactly at L2 point. We still haven't introduced a reference Frame, so the user must keep in mind in which reference Frame he wants to define the initial conditions of its satellite.

```
final PVCoordinates initialConditions = new PVCoordinates(new Vector3D(0.0, 0.0, 0.0), new Vector3D(0.0, 0.0, 0.0));
```

Frames setting

We need to load a few Celestial bodies, those we consider in gravitational interaction with our satellite, as well as the Earth-Moon barycenter, since we will compute our inertia forces with respect to its attached frame. We then create the reference frame attached to the L2 point, and the inertial reference frame attached to the Earth-Moon barycenter.

```
final CelestialBody earth = CelestialBodyFactory.getEarth();
final CelestialBody moon  = CelestialBodyFactory.getMoon();
final CelestialBody earthMoonBary = CelestialBodyFactory.getEarthMoonBarycenter();

final Frame l2Frame = new L2Frame(earth, moon);
final Frame earthMoonBaryFrame = earthMoonBary.getInertiallyOrientedFrame();

final Frame inertiaFrame = earthMoonBaryFrame;
final Frame integrationFrame = l2Frame;
final Frame outputFrame = l2Frame;
```

Propagator preparations

We now transform our `PVCoordinates` into `AbsolutePVCoordinates`, define the satellite attitude, and use all of it to build the `SpacecraftState`

```
final AbsolutePVCoordinates initialAbsPV = new AbsolutePVCoordinates(integrationFrame, initialDate, initialConditions);
Attitude arbitraryAttitude = new Attitude(integrationFrame,
                                          new TimeStampedAngularCoordinates(initialDate,
                                          new PVCoordinates(Vector3D.PLUS_I, Vector3D.PLUS_J),
                                          new PVCoordinates(Vector3D.PLUS_I, Vector3D.PLUS_J)));
final SpacecraftState initialState = new SpacecraftState(initialAbsPV, arbitraryAttitude);
```

We will use a variable-step 8(5,3) Dormand-Prince integrator. This integrator needs a few initialization parameters. First, let’s set boundaries on the integration steps, to prevent a too long computing or a too long integration step.

```
final double minStep = 0.001;
final double maxstep = 3600.0;
```

We also need to set the acceptable error of our integration. This error is used to adjust the integration step and does not stand for the overall error of the propagation.

```
final double positionTolerance = 0.001;
final double velocityTolerance = 0.00001;
final double massTolerance     = 1.0e-6;
final double[] vecAbsoluteTolerances = {
        positionTolerance, positionTolerance, positionTolerance,
        velocityTolerance, velocityTolerance, velocityTolerance,
        massTolerance
        };
final double[] vecRelativeTolerances = new double[vecAbsoluteTolerances.length];
```

We can now define the numerical integrator of our propagator.

```
AdaptiveStepsizeIntegrator integrator = new DormandPrince853Integrator(minStep, maxstep,
                                                                       vecAbsoluteTolerances,
                                                                       vecRelativeTolerances);
```

Propagator building and use

And finally, we can build our propagator. First we use the `NumericalPropagator` constructor with the previously defined numerical integrator, and then we specify the properties of the evolution model. The use of `SingleBodyAttraction` means that we do not constrain the type of orbit, and that we need to use `setOrbitType(null)` as well as `setIgnoreCentralAttraction(true)`. The non-inertial aspect of our integration reference frame is handled by `InertialForces`, which will compute the inertial accelerations that appear in this frame. Each attracting body will be provided to the propagator by calling `SingleBodyAttraction` for each body in gravitational interaction with our spacecraft. We then only need to set the inital spacecraft state, and the propagator mode, more details about propagator modes can be found in the Propagation tutorial .

```
NumericalPropagator propagator = new NumericalPropagator(integrator);
propagator.setOrbitType(null);
propagator.setIgnoreCentralAttraction(true);
propagator.addForceModel(new InertialForces(earthMoonBaryFrame));
propagator.addForceModel(new SingleBodyAbsoluteAttraction(earth));
propagator.addForceModel(new SingleBodyAbsoluteAttraction(moon));
propagator.setInitialState(initialState);
propagator.getMultiplexer().add(outputStep, new TutorialStepHandler("test.dat", outputFrame));
```

In the end we can do the propagation itself, from an initial date, for a given duration.

```
SpacecraftState finalState = propagator.propagate(initialDate.shiftedBy(integrationTime));
final PVCoordinates pv = finalState.getPVCoordinates(outputFrame);
System.out.println("initial conditions: " + initialConditions);
System.out.println("final conditions: " + pv);
```

Stephandling

Now, we can deal with our stephandler, it will print the position, velocities and acceleration at each output step of the integration.

```
 private static class TutorialStepHandler implements OrekitFixedStepHandler {
 private Frame outputFrame;
 private TutorialStepHandler( final Frame frame) {
    outputFrame = frame;
    }

 public void init(final SpacecraftState s0, final AbsoluteDate t) {
        System.out.format(Locale.US,
                                 "%s %s %s %s %s %s %s %s %s %s %n",
                                 "date", "                          X", "                     Y",
                                 "                 Z", "                 Vx", "                 Vy",
                                 "                Vz", "                ax", "                ay",
                                 "                az");
    }

    public void handleStep(SpacecraftState currentState, boolean isLast) {
        try {
            final TimeScale utc = TimeScalesFactory.getUTC();
            final AbsoluteDate initialDate = new AbsoluteDate(2000, 01, 01, 0, 0, 00.000,
                                                 TimeScalesFactory.getUTC());
            final AbsoluteDate d = currentState.getDate();
            final PVCoordinates pv = currentState.getPVCoordinates(outputFrame);
            System.out.format(Locale.US,
                              "%s %18.12f %18.12f %18.12f %18.12f %18.12f %18.12f %18.12f %18.12f %18.12f%n",
                              d, pv.getPosition().getX(),
                              pv.getPosition().getY(), pv.getPosition().getZ(),
                              pv.getVelocity().getX(), pv.getVelocity().getY(),
                              pv.getVelocity().getZ(), pv.getAcceleration().getX(),
                              pv.getAcceleration().getY(),
                              pv.getAcceleration().getZ()
                              );

            if (isLast) {
                final PVCoordinates finalPv =
                                currentState.getPVCoordinates(outputFrame);
                System.out.println();
                System.out.format(Locale.US,
                                  "%s %12.0f %12.0f %12.0f %12.0f %12.0f %12.0f%n",
                                  d, finalPv.getPosition().getX(),
                                  finalPv.getPosition().getY(),
                                  finalPv.getPosition().getZ(),
                                  finalPv.getVelocity().getX(),
                                  finalPv.getVelocity().getY(),
                                  finalPv.getVelocity().getZ());
                System.out.println();
            }
        } catch (OrekitException oe) {
            System.err.println(oe.getMessage());
        }
    }
}
}
```

Results

The printed results are shown below:

date	X	Y	Z	Vx	Vy	Vz	ax	ay	az
2000-01-01T00:00:00.000	0.000000000000	0.000000000000	0.000000000000	0.000000000000	0.000000000000	0.000000000000	0.000007203264	-0.000023617334	0.000000988884
2000-01-01T00:10:00.000	1.310181754844	-4.509937950657	0.177995376805	0.004389441414	-0.015464490795	0.000593311195	0.000007424953	-0.000027930975	0.000000988817
2000-01-01T00:20:00.000	5.292761075816	-19.075027371572	0.711964514771	0.008906039828	-0.033517172552	0.001186577792	0.000007627122	-0.000032244627	0.000000988736
2000-01-01T00:30:00.000	12.020526802001	-45.248275712283	1.601878056606	0.013538115033	-0.054158399174	0.001779791095	0.000007710027	-0.000035388726	0.000000989294
2000-01-01T00:40:00.000	21.178373170978	-80.022897721805	2.850254273796	0.017011800885	-0.062143736699	0.002381480483	0.000005918985	-0.000015387558	0.000001002821
2000-01-01T00:50:00.000	32.466478604114	-120.337158982044	4.459645552958	0.020640742565	-0.072667569026	0.002983148787	0.000006174238	-0.000019691906	0.000001002738
2000-01-01T01:00:00.000	45.976718715462	-167.740438088863	6.430022015915	0.024416959249	-0.085773707784	0.003584763069	0.000006410327	-0.000024000459	0.000001002638
2000-01-01T01:10:00.000	61.793504290626	-223.776772118858	8.761351689337	0.028327931996	-0.101452129474	0.004186320438	0.000006626318	-0.000028300879	0.000001002529
2000-01-01T01:20:00.000	79.995099948214	-290.000479343336	11.453591760838	0.032363731793	-0.119724013059	0.004787800539	0.000006823101	-0.000032605395	0.000001002403
2000-01-01T01:30:00.000	100.652428154023	-367.962135000874	14.506696359194	0.036511612039	-0.140576870435	0.005389201731	0.000006303102	-0.000028624115	0.000001006865
2000-01-01T01:40:00.000	123.451861739137	-454.645479496582	17.923153578986	0.039511558100	-0.148775025605	0.005998989925	0.000005129778	-0.000015774359	0.000001016276
2000-01-01T01:50:00.000	148.097431329154	-547.007595557232	21.705469789230	0.042665311712	-0.159528224853	0.006608717996	0.000005379490	-0.000020069660	0.000001016148
2000-01-01T02:00:00.000	174.679033538614	-646.594792449283	25.853598980322	0.045963054147	-0.172858638623	0.007218365096	0.000005609742	-0.000024365065	0.000001016006
2000-01-01T02:10:00.000	203.279559120647	-754.953413963500	30.367489987968	0.049393110759	-0.188766311565	0.007827922613	0.000005820538	-0.000028660515	0.000001015850
2000-01-01T02:20:00.000	233.974894931263	-873.629820586518	35.247086470272	0.052943809883	-0.207251254684	0.008437381900	0.000006011884	-0.000032955956	0.000001015679
2000-01-01T02:30:00.000	266.833927466020	-1004.170390063609	40.492326874541	0.056603523474	-0.228313934164	0.009046734005	0.000006203223	-0.000037487428	0.000001015363
2000-01-01T02:40:00.000	301.547225264656	-1143.577979776640	46.105651401557	0.059129787256	-0.236758043587	0.009664358716	0.000004333863	-0.000016135286	0.000001029339
2000-01-01T02:50:00.000	337.820135753835	-1288.794334653753	52.089537502703	0.061804333216	-0.247725104421	0.010281911014	0.000004578052	-0.000020421603	0.000001029166
2000-01-01T03:00:00.000	375.740556403237	-1441.362400813750	58.443923059929	0.064619473654	-0.261262891438	0.010899355930	0.000004806508	-0.000024752997	0.000001028954
2000-01-01T03:10:00.000	415.387681993014	-1602.804245773708	65.168752226799	0.067560866331	-0.277337889071	0.011516703202	0.000005008336	-0.000028994444	0.000001028777
2000-01-01T03:20:00.000	456.837148988767	-1774.683165398515	72.263941309722	0.070622616256	-0.296020485822	0.012133905439	0.000005194262	-0.000033280871	0.000001028561

```
2000-01-01T03:20:00.000         457        -1775           72             0          -0             0

initial conditions: {P(0.0, 0.0, 0.0), V(0.0, 0.0, 0.0), A(0.0, 0.0, 0.0)}
final conditions: {2000-01-01T03:20:00.000, P(456.8371489887673, -1774.6831653985146, 72.2639413097221), V(0.0706226162556273, -0.29602048582231744, 0.012133905438764159), A(5.194261804654593E-6, -3.328087090313
```


Yksiköt: Paikka X Y Z metriä. Nopeus Vx Vy Vz m/s. Kiihtyvyys ax ay az m/s2

- Orekit -kirjastoa on jo käytetty Holmes -komeetan tutkimuksessa mallintamaan Holmes -komeetan vuoden 2007 massiivista outburst -tapahtumaa sekä partikkelien liikettä ja pölyvanan muodostumista. Malli perustuu Esko Lyytisen partikkelimallinnukseen. Mallia on kehitetty edelleen. Uusi malli on esimerkiksi tarkempi ja monipuolisempi. Kehitystyö yhteistyössä Maria Gritsevichin työryhmän kanssa.

- Meteoriparvien partikkelimallinnuksesta on olemassa C++ koodilla tehty versio, jota käytettiin esimerkiksi Leonidien meteoriparven mallinnuksessa.

- Konvertointi on menossa Orekit -pohjaiselle toteutukselle meteoriparvien Esko Lyytisen partikkelimallille. Mallia kehitetään edelleen konversion teon yhteydessä. Kehitystyö yhteistyössä Maria Gritsevichin kanssa.

Kehityskohteita

- Mallin ajaminen pilvialustoilla.
- Kehitysympäristön helppo asennus ja koodin kopiointi tietokoneesta toiseen.
- Orekit -kirjaston muokkaaminen.
- Objektien lisääminen. Esimerkiksi asteroidien ja komeettojen lisääminen Chebyshev -polynomeina tms.
- Open Source.

Selostus Esko Lyytisen partikkelimallista, perusmalli sekä parannettu malli

- Komeetalla oletetaan olevan ydin ja sitä kiertävä partikkelijoukko. Komeetan ollessa perihelissä, eli lähimpänä Aurinkoa, tapahtuu gravitationaalinen partikkelien poistuma L1 ja L2 Lagrangen pisteiden kautta. Tämä tapahtuu samaan tapaan, kuin Orekit esimerkissä on esitetty Maan ja Kuun L2 pisteen ja satelliitin osalta, mutta Orekit esimerkissä on käsitelty vain gravitationaalista tilannetta. Tässä tapauksessa täytyy mallintaa vielä Auringon säteilypaineen vaikutus partikkeliin.

- Tähän vaikuttaa pääasiassa Auringon säteilypaine.

- Jos partikkelien ejektionopeudet ovat hyvin pieniä ja ne eivät paljon vaihtele, muodostuu partikkelivanoja yhden kierroksen jälkeen.

- Vana hajaantuu epäsymmetrisen absorption ja uudelleensäteilyn ansiosta.
- Hajaantuminen radan suuntaisesti ja poikittainen (transverse) hajaantuminen aiheuttavat pääasiassa meteoriparven aktiivisuusvaihtelut (ZHR = zenithal hourly rate).
- Planeettojen gravitationaalinen vaikutus muuttaa meteoroidin rataa.
- Muitakin vaikuttavia tekijöitä on olemassa.

Säteilypaine

Säteilypaine jakaantuu kolmeen komponenttiin seuraavasti:

- Absorptiosta johtuvaan komponenttiin
- Uudelleensäteilemisestä johtuvaan komponenttiin
- Sironnasta johtuvaan komponenttiin

Poynting-Robertson efekti

- Liikkeen suuntaan tapahtuva säteily on liikemäärältään suurempi kuin taaksepäin, josta aiheutuu partikkelin hidastuminen.

Ei-isotrooppinen sironta

- Lähde: Esko Lyytinen et al. Earth, Moon and Planets 2000.

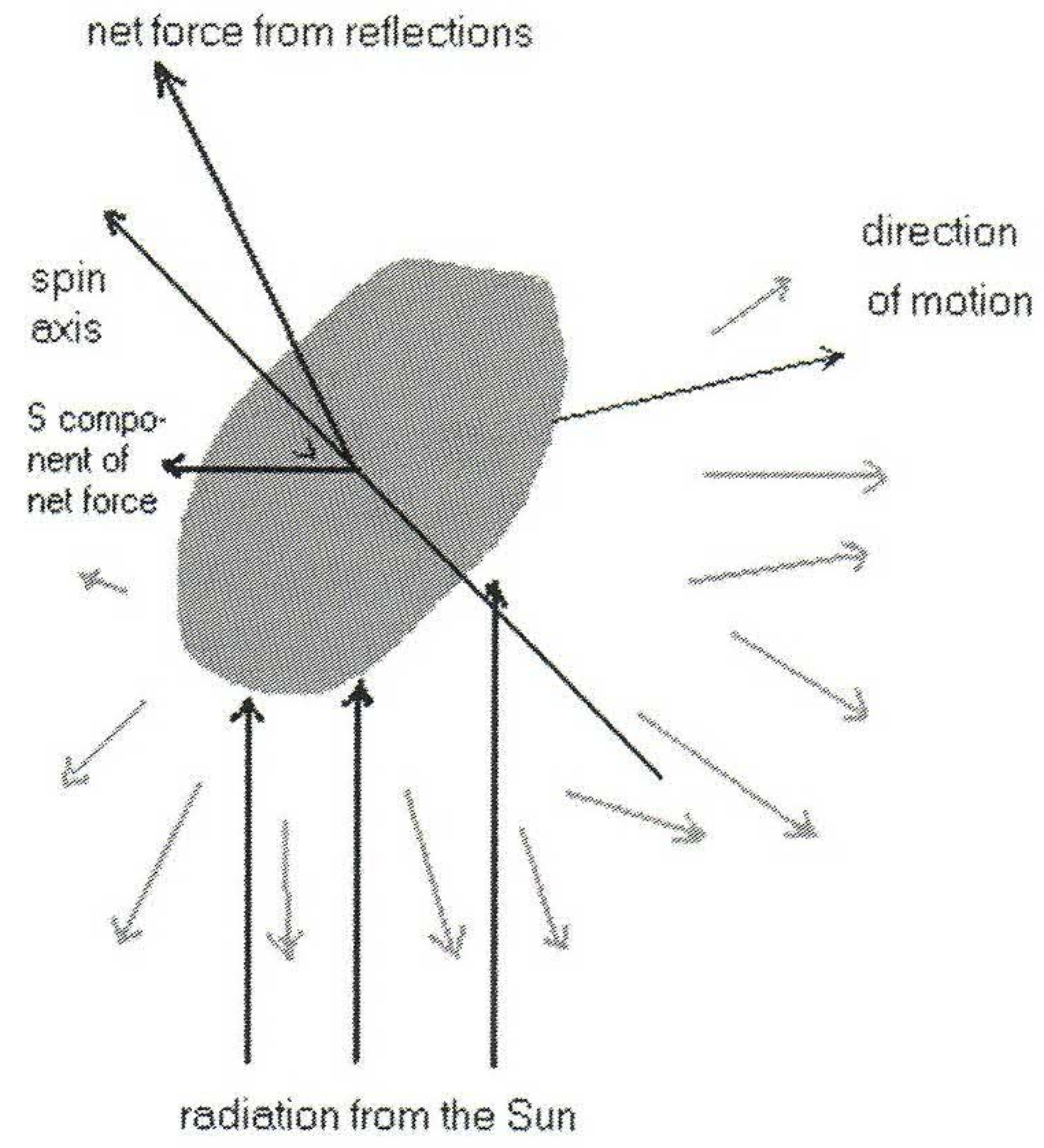


Figure 2. Non-isotropic scattering may give force components that change the orbit and, among others, affect the orbital period, i.e. produce an A2 effect.

Seasonal Yarkovsky efekti

- Lähde: Esko Lyytinen et al. Earth, Moon and Planets 2000.

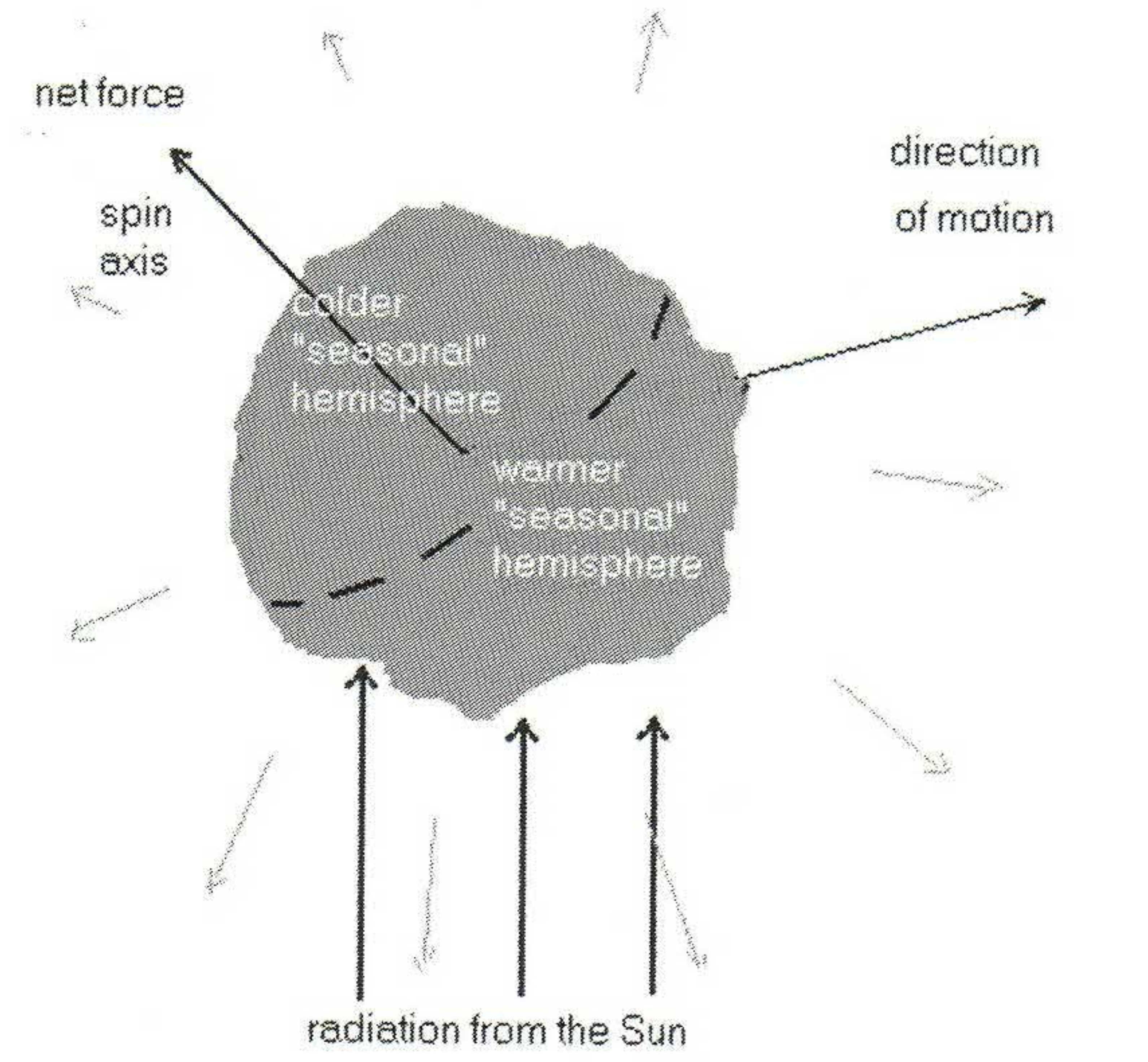


Figure 3. The mechanism for seasonal Yarkovsky is illustrated here. This effect can perturb the orbital motion of a particle much like the effect illustrated in Figure 2.

A2 efekti

- A2 efekti, eli jatkuva kiihtyvyys, sisältää kaikki ei-gravitationaaliset prosessit, jotka muuttavat meteoroidin kierrosaikaa.
- Keskeisillä kohtaamisilla ei kovin paljon vaikutusta, mutta ei-keskeisillä kohtaamisilla voi lisätä ZHR arvoa.

- delta-A: ”difference of original semi-major axis from that of the comet at ejection time”
- Mean Anomaly factor fM: Hajaantuminen radan suuntaisesti. (vrt. McNaught Asher 1999).
- Hajaantuminen kaikissa suunnissa otettu mallissa huomioon.

ZHR laskenta

- Lähde: Esko Lyytinen et al. Earth, Moon and Planets 2000.
- rE - rD = Maan ja meteroidin välinen etäisyys.
- Delta A ja fM on selostettu edellisellä dialla.
- fn(delta a) on selostettu seuraavalla dialla ja sitä seuraavalla on parametreja taulukkona.

The radial density ρ_r of the trails is assumed to be of the form:

$$\rho_r \sim (1 + \Delta r^2)^{-0.5p} \qquad (6)$$

where Δr is the difference in radial ecliptic plane crossing distances of Earth and meteoroid r_E-r_D, scaled according to:

$$\Delta r = (r_E - r_D) \;/\; (0.00059 \text{ x } n \text{ x } \Delta a) \qquad (7)$$

where n is the number of orbits since ejection and (r_E-r_D) and Δa are in astronomical units. Equation 6 was derived from the observed activity profiles of past meteor storms with the assumption, that the profile shape radially (from Sun) is the same as normal to the orbital plane. In equation 7, the terms n and Δa arise, because the trail width is expected to widen directly with n and also with an increase of Δa, practically resulting in a widening with increasing distance from the comet. From published graphs of the 1966 storm (Jenniskens, 1995; Mason 1995; Brown 1999), the free parameter p was derived as p = 2.7 ± 0.7. The scaling factor 0.0002 in Equation 7 was derived from the 1966 storm case, which had n = 2 and Δa = 0.169. In addition, the densities are assumed proportional to the distance from the center raised to the power -2.7 far away from the trailet center.
We now introduce the empirical function fn(Δa), which is the ZHR for a one-revolution central encounter having the original Δa. Noticing further the mentioned spread (and dilution) directly in two dimensions with n and further with f_M, we get this expression for ZHR:

$$ZHR = fn(\Delta a) \; (f_M / n^2) \; (1 + \Delta r^2)^{-0.5p} \qquad (8)$$

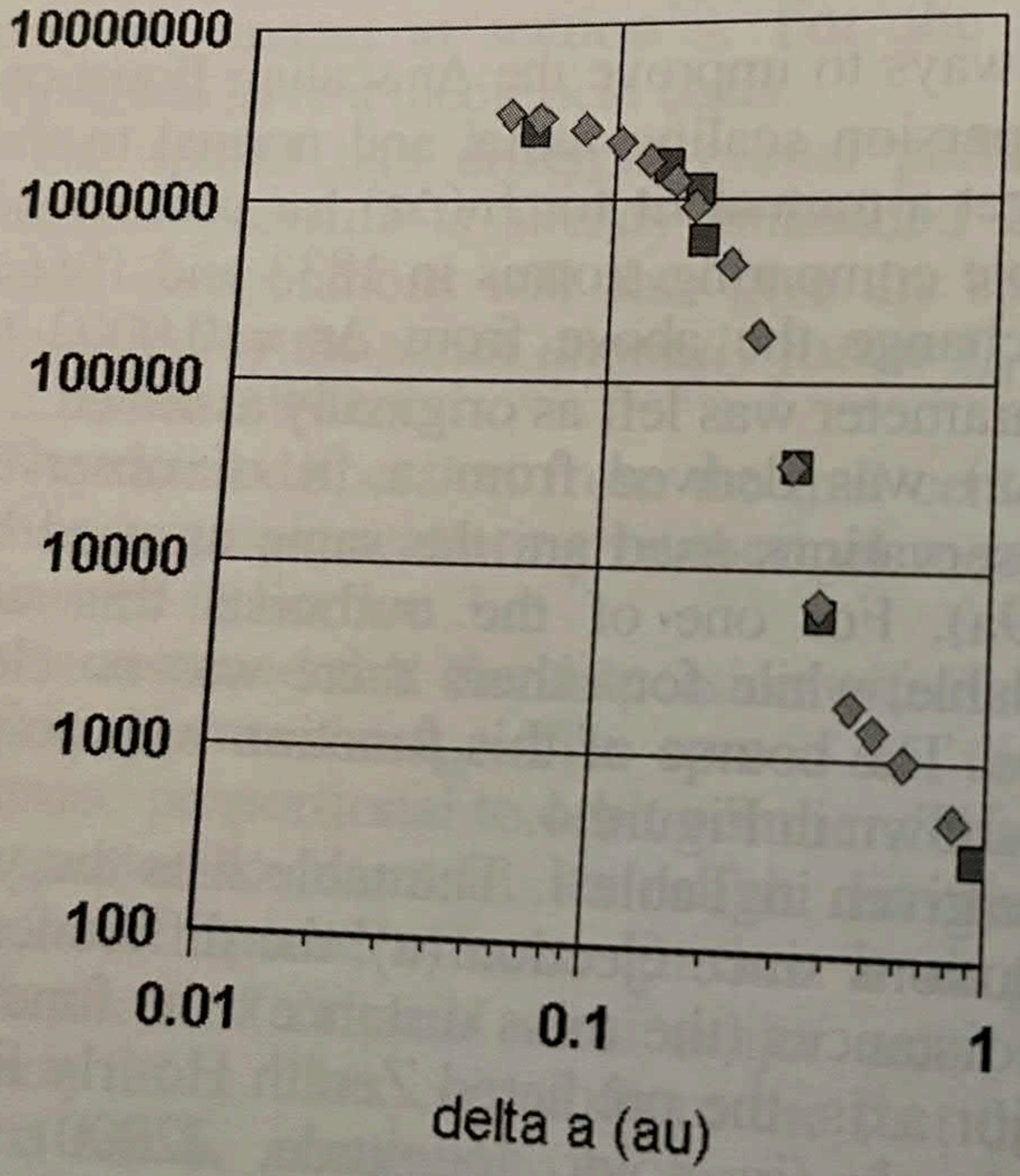


Figure 4. The graph shows the approximation of fn(Δa) from seven observed peak Zenith Hourly Rates. Squares are derived from observations of 1866, 1999, 1966, 1833, 1869, 1867, and 1969. Diamonds show the derived numerical approximation.

TABLE I

Year	Rev.	rE-rD	fM	delta a	Pred. ZHR	Sol.long. 2000.0	Date Nov.	Time UT	Comments
2000	2	-0.0012	0.55	0.30	215	235.270	17	07:50	mostly faint
2000	4	0.00080	0.135	0.116	700	236.279	18	07:50	
2000	8	0.00080	0.250	0.065	700	236.103	18	03:40	fM locally bigger, but strongly affected by A2 ef
2001	4	0.00025	0.135	0.144	6100	236.467	18	18:22	
2001	5	0.00178	0.114	0.095	60	236.29	18	14:10	
2001	6	0.00135	0.123	0.080	110	236.20	18	12:00	
2001	7	-0.00043	0.140	0.081	2000	236.115	18	09:58	
2001	9	0.0001	0.260	0.043	1500-2000	236.433	rev:s 9,10,11		strongly affected by A2 ef
2001	10	0.0006	0.160	0.030	600	236.423	ZHR comb.		
2001	11	0.0004	0.160	0.026	260	236.425	ab. 2500 to 3000		
							18	ab. 17:30	
2002	4	-0.00004	0.148	0.174	7400	236.894	19	10:44	
2002	5	0.00148	0.115	0.120	160	236.72	19	06:45	
2002	7	-0.00013	0.130	0.114	4500	236.612	19	04:02	
2003	15/14 co/me	-0.001 to -0.003	1 - 2.5 extr.conf.!	0.28	250	230.69 to230.78+	13 !!!	13 to19	very special also uncertain
2006	2	-0.0002	0.470	0.94	50	236.618	19	04:48	maybe strong teles-
2007	2	-0.0004	0.560	1.06	30	236.107	19	22:55	copic outbursts

Leonidit

- Lähde: Esko Lyytinen et al. Earth, Moon and Planets 2000.
- Maan ja Auringon välinen etäisyys on 0,988 ja 0,989 välillä.
- Kuvassa nähdään esimerkiksi vuoden 2001 marraskuussa keskittymä lähellä tätä arvoa.
- Emokomeetta on 55P/Tempel-Tuttle. Kiertoaika 33 vuotta.

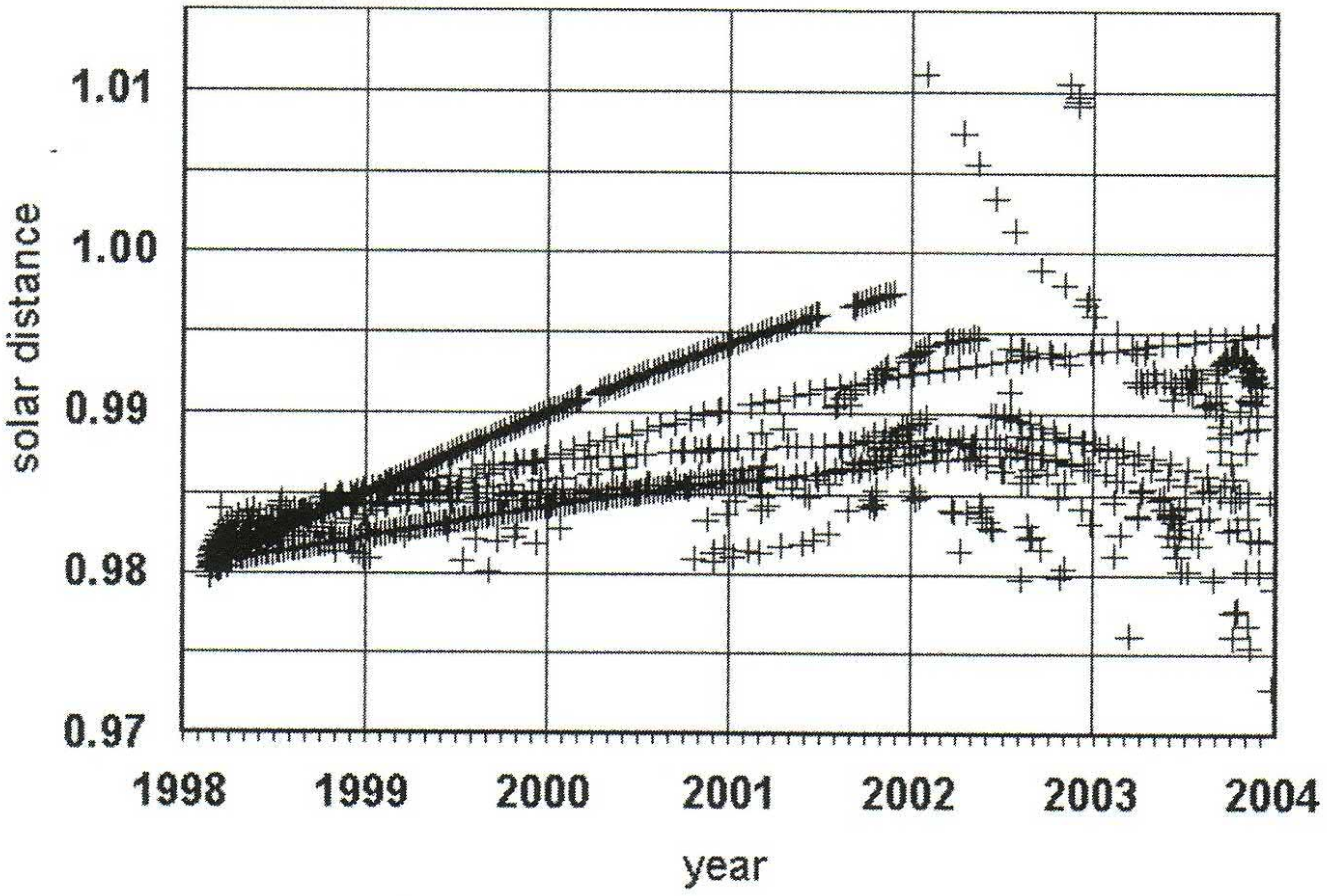


Figure 1. Distance from the Sun of modeled trail particles at the descending node. The vertical lines are at the start of January. The Earth-Sun distance is at about 0.988 to 0.989 AU. A concentration of points near this value in mid-November (as in the year 2001) indicates a likely meteor storm.

- Huomioitava, että tässä esityksessä tähän saakka esitetty malli on ns. perusmalli, eikä edelleen kehitetty ja parannettu malli, jolla sitten laskettiin lopulliset Leonidiennusteet. Tässä esityksessä oleva perusmalli on esitetty Earth, Moon and Planets kirjassa, Esko Lyytinen et al.

- Parannettu malli löytyy: Improved 2001 Leonid Storm Predictions from a Refined Model. Esko Lyytinen et al. WGN, the Journal of the IMO 29:4 (2000).

- Esitän tässä myös parannetulla mallilla laskettuja ennusteita seuraavana.

- Parannetussa mallissa on edelleen kehitetty mm. ns. A2 mallinnusta. Osa vanoista on vuodelle 2000, ja niillä on sovitettu mallia vuodelle 2001.

Leonidien 2001 ennusteet

- Edelleenkehitetty parannettu malli sovitettiin havaintotuloksiin sovittamalla maksimin ajankohta sekä ZHR käyrän muoto. Sovituksessa käytettiin IMO:n visuaalihavaintoja vuosilta 1999 ja 2000 sekä NASA:n MAC lentojen tuloksia (Peter Jenniskens) vuodelta 2000. ZHR käyrällä on havaittu olevan lähellä Lorentz jakaumaa oleva muoto.

- Jokaisella kohtaamisella on half strength, half-width arvo, mutta half strength vastaa 39% arvoa, koska jakauma ei ole aivan tarkkaan Lorentz jakauma.

- **Tämä malli antoi kaikista parhaimmat tulokset Leonidien aktiivisuudelle!**

Improved 2001 Leonid Storm Predictions from a Refined Model

Esko Lyytinen, Markku Nissinen and Tom Van Flandern

It is expected that the cumulative non-gravitational effect on the semi-major axis in the dust trails that produce meteor outbursts is an important factor affecting the particle spread along the orbit and therefore the apparent ZHR behavior. In this work, we determine a numerical value estimate of this effect from earlier observations, mainly those from the year 2000. Besides getting a better post-prediction of the course of the ZHR curve, we also find that the observed maximum ZHR value of the 8-revolution outburst (1733) is better explained with the new model when the derived non-gravitational A2 distribution is taken into account. The model and newly derived parameter values are used to improve the predictions for the year 2001. The predicted outbursts for the year 2002 have not yet been treated in this way.

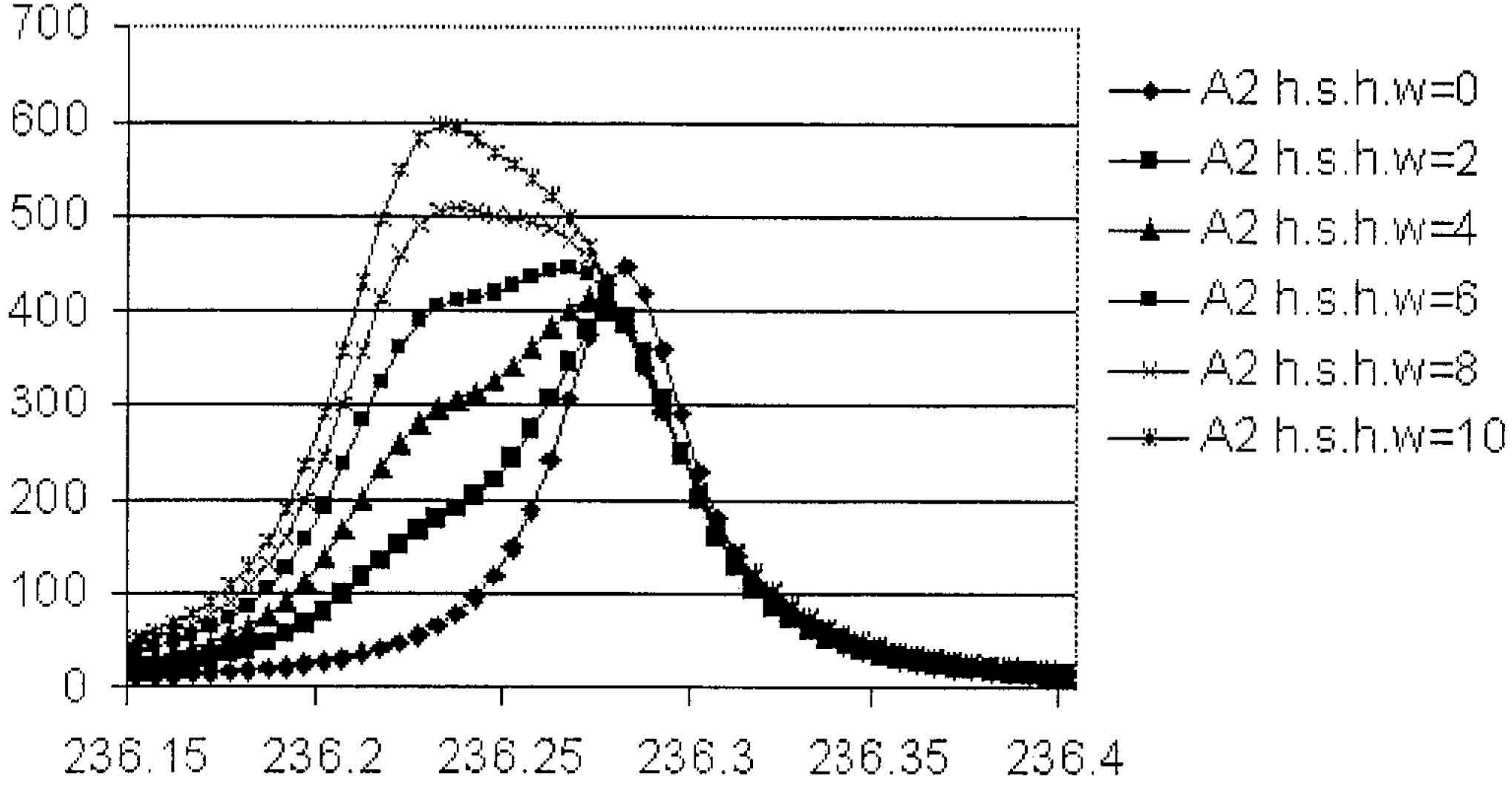


Figure 2 – The 2000, 4-revolution (1866) ZHR change with A2-distribution width parameter (half-strength half-width). Best fits around 4 to 7; data from [7,8]

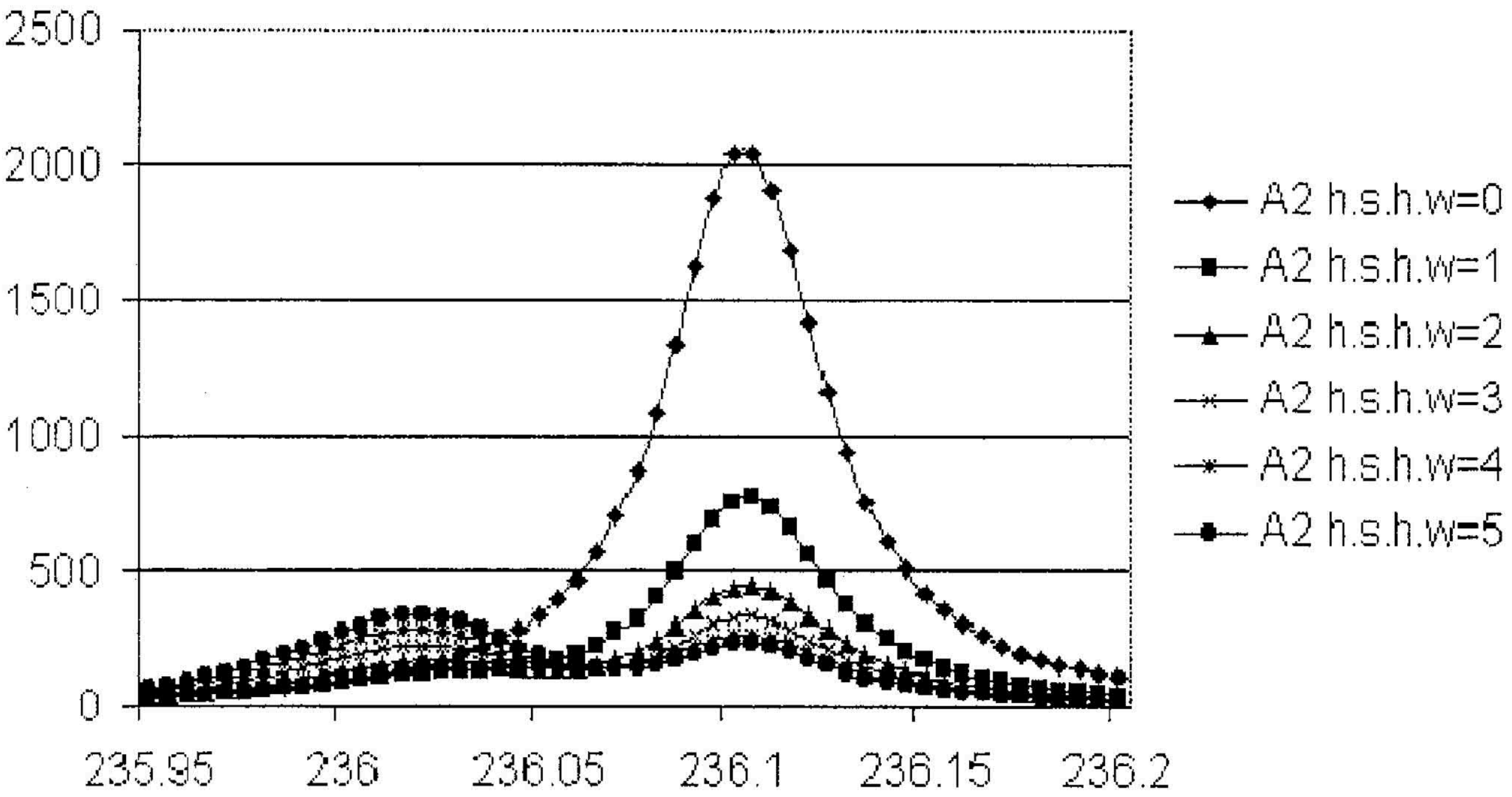


Figure 3 – The 2000, 8-revolution (1733) ZHR change with A2-distribution width parameter. Best fit 3.2; data from [7].

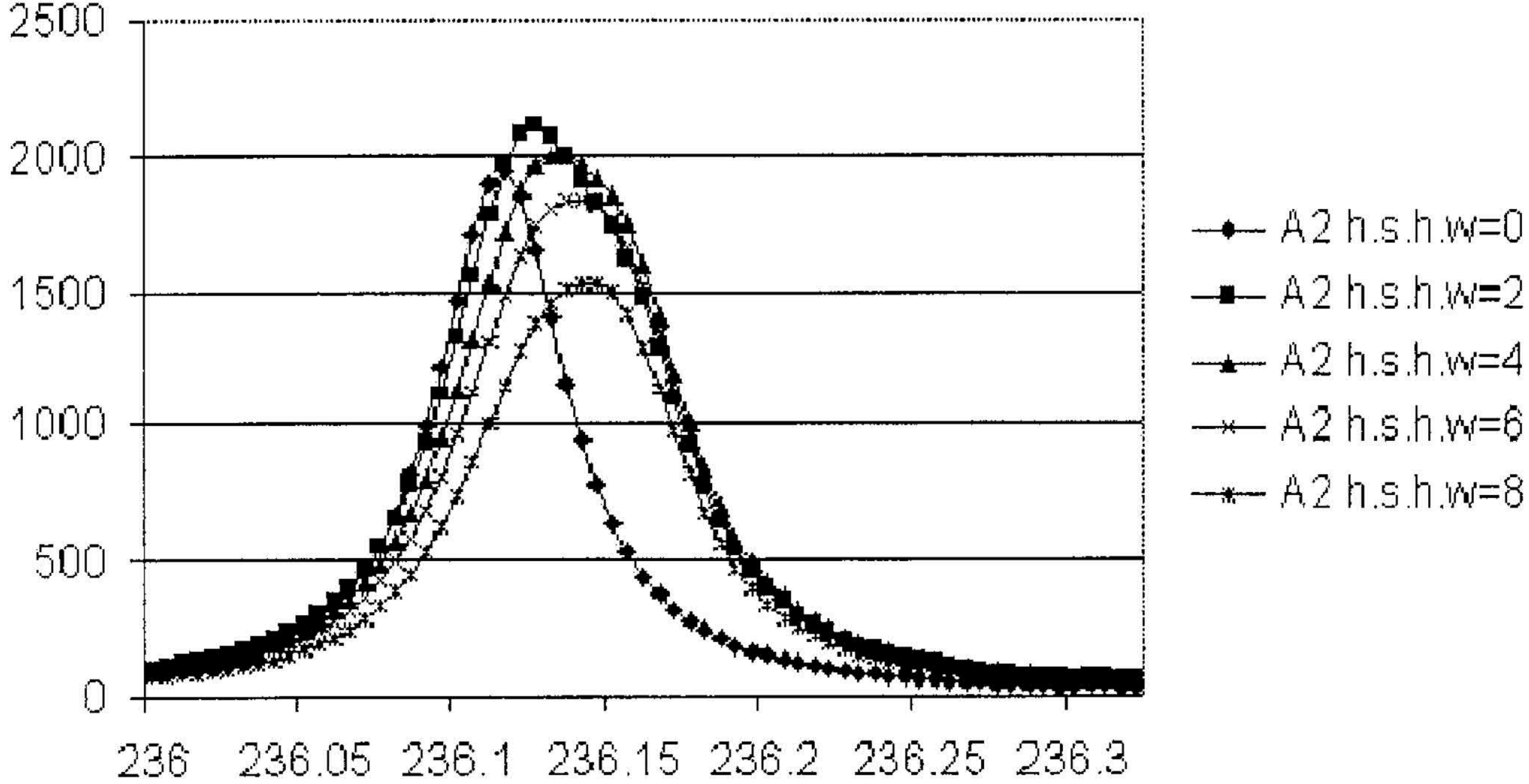


Figure 9 – The 2001, 7-revolution trail encounter ZHR-change with A2-distribution width parameter.

Table 2 – Predictions for 2001, in UT. Conversion into J2000 solar longitudes, on November 18, 2001, were made according to the formula $\lambda_{\odot} = 235^{\circ}\!.692 + 1^{\circ}\!.009 \times d$, where d is the time in days from the start of the day (November 18, 0^{h} UT). The predicted peaks values for trails 10 and 11 are rounded to the nearest ten minutes.

Trail	Nodal encounter (in original model)	Predicted peak (in A2 modeling)	Half-width (minutes)	ZHR
4 – rev	$18^{h}26^{m}$	$18^{h}20^{m}$	43	5000
5 – rev	$14^{h}10^{m}$	A2 modeling not applied	(29)	60
6 – rev	$12^{h}00^{m}$	A2 modeling not applied	(30)	110
7 – rev	$10^{h}04^{m}$	$10^{h}28^{m}$	58	2000
		(non symmetric	– 53/ + 62)	
9 – rev	$17^{h}38^{m}$	$18^{h}03^{m}$	62	2600
		(non symmetric	– 58/ + 65)	
10 – rev	$17^{h}23^{m}$	$19^{h}10^{m}$	$\geq$ 140	150
11 – rev	$17^{h}26^{m}$	$19^{h}10^{m}$	$\geq$ 90	150

Ja mitä havainnot kertoivat ?

The Leonids

Bulletin 17 of the International Leonid Watch First Global Analysis of the 2001 Leonid Storms

Rainer Arlt, Javor Kac, Vladimir Krumov, Andreas Buchmann, and Jan Verbert

Observers in America and Asia have monitored strong peaks of Leonid activity on November 18, 2001. We present a first analysis of global data based on the reports of 177 observers who recorded 137 146 Leonids. Main activity peaks are found for solar longitudes (all J2000.0) $\lambda_\odot = 236^\circ\!.137 \pm 0^\circ\!.003$ (November 18, $10^h39^m \pm 4^m$) and $\lambda_\odot = 236^\circ\!.458 \pm 0^\circ\!.003$ (November 18, $18^h16^m \pm 4^m$). Secondary peaks are found near the main Asian maximum at $\lambda_\odot = 236^\circ\!.448 \pm 0^\circ\!.003$ (November 18, $18^h02^m \pm 4^m$) and $\lambda_\odot = 236^\circ\!.467 \pm 0^\circ\!.003$ (November 18, $18^h30^m \pm 4^m$). The American peak actually exhibits a bimodal structure with two similar maxima at $236^\circ\!.137 \pm 0^\circ\!.003$ and $236^\circ\!.154 \pm 0^\circ\!.003$, the second of them being 24 minutes later. The population index shows sharp peaks for the storms, whereas the background level during the interval $\lambda_\odot = 235^\circ\!.6$–$237^\circ\!.0$ is characterized by $r < 1.9$.

Table 1 – Overview of predictions and observed activity of the 2001 Leonids. The two models refer to [2] and [1], respectively. The peak times with exclamation marks are the main maxima, whereas the other times denote slight enhancements of activity with medium significance. Model times in brackets are tentative associations with observed features. The number of individual observing periods is given as "Per."

Dust trail	Models		Observations			
	McNaught Asher	Lyytinen, Nissinen, van Flandern	$\lambda_\odot$ (J2000.0)	November 18 UT	ZHR	Per.
7-rev	(09^h10^m)	–	236°082	09^h21^m (!)	680 ± 60	19
7-rev	09^h55^m	10^h28^m	236°137	10^h39^m (!)	1620 ± 40	75
7-rev	(11^h00^m)	–	236°154	11^h03^m (!)	1610 ± 60	37
6-rev	–	(12^h00^m)	236°179	11^h39^m	650 ± 40	19
6-rev	–	(12^h00^m)	236°195	12^h01^m	520 ± 40	19
–	–	–	236°262	13^h40^m	400 ± 40	19
9-rev	17^h24^m	18^h03^m	236°448	18^h02^m	2830 ± 70	66
4-rev	18^h13^m	18^h20^m	236°458	18^h16^m (!)	3430 ± 90	39
	–	–	236°467	18^h30^m (!)	3010 ± 70	55
11-rev	18^h43^m	19^h10^m	236°491	19^h04^m	1840 ± 60	47

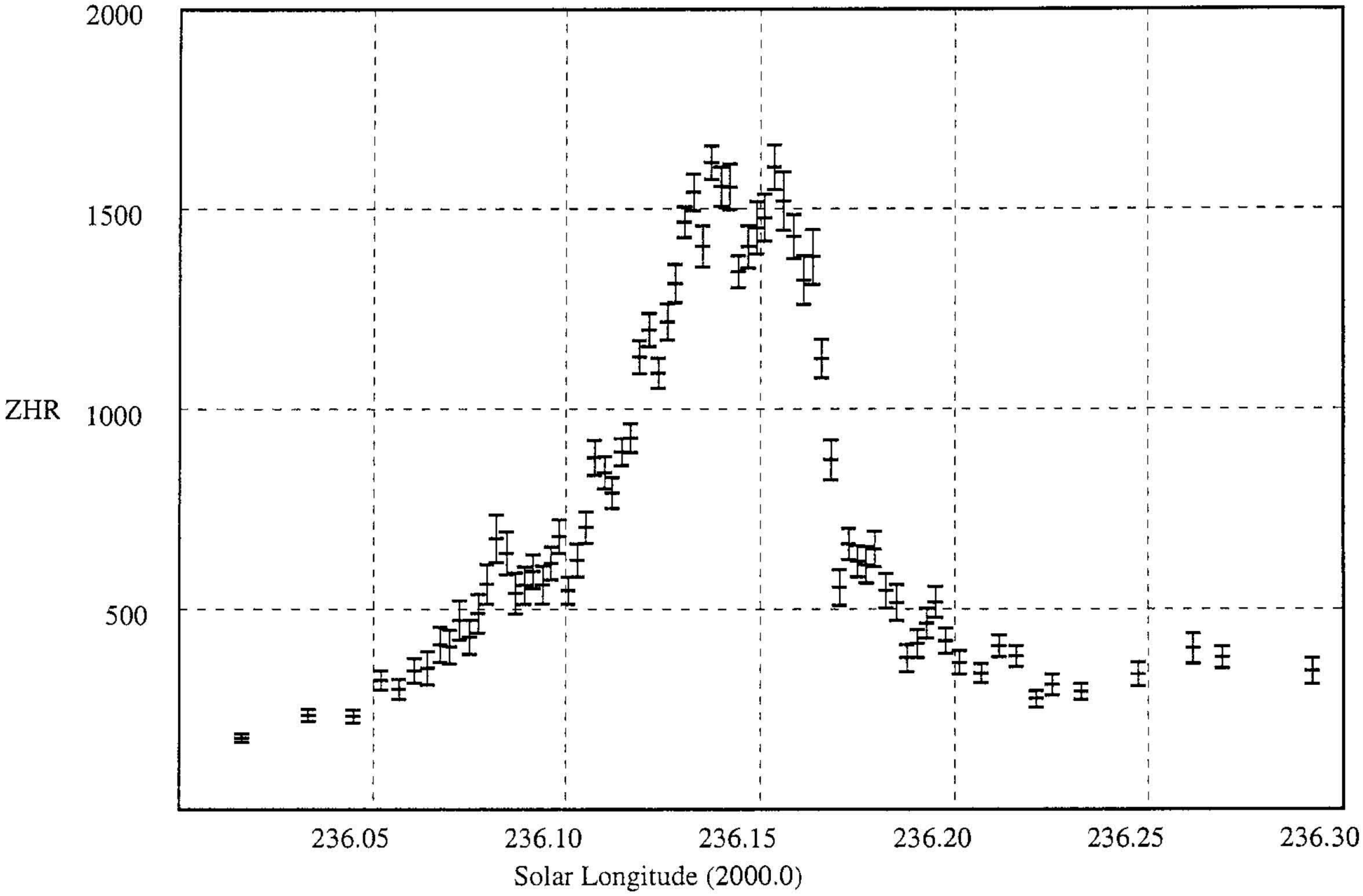


Figure 4 – Final profile of the first 2001 Leonid maximum as seen from American geographical longitudes.

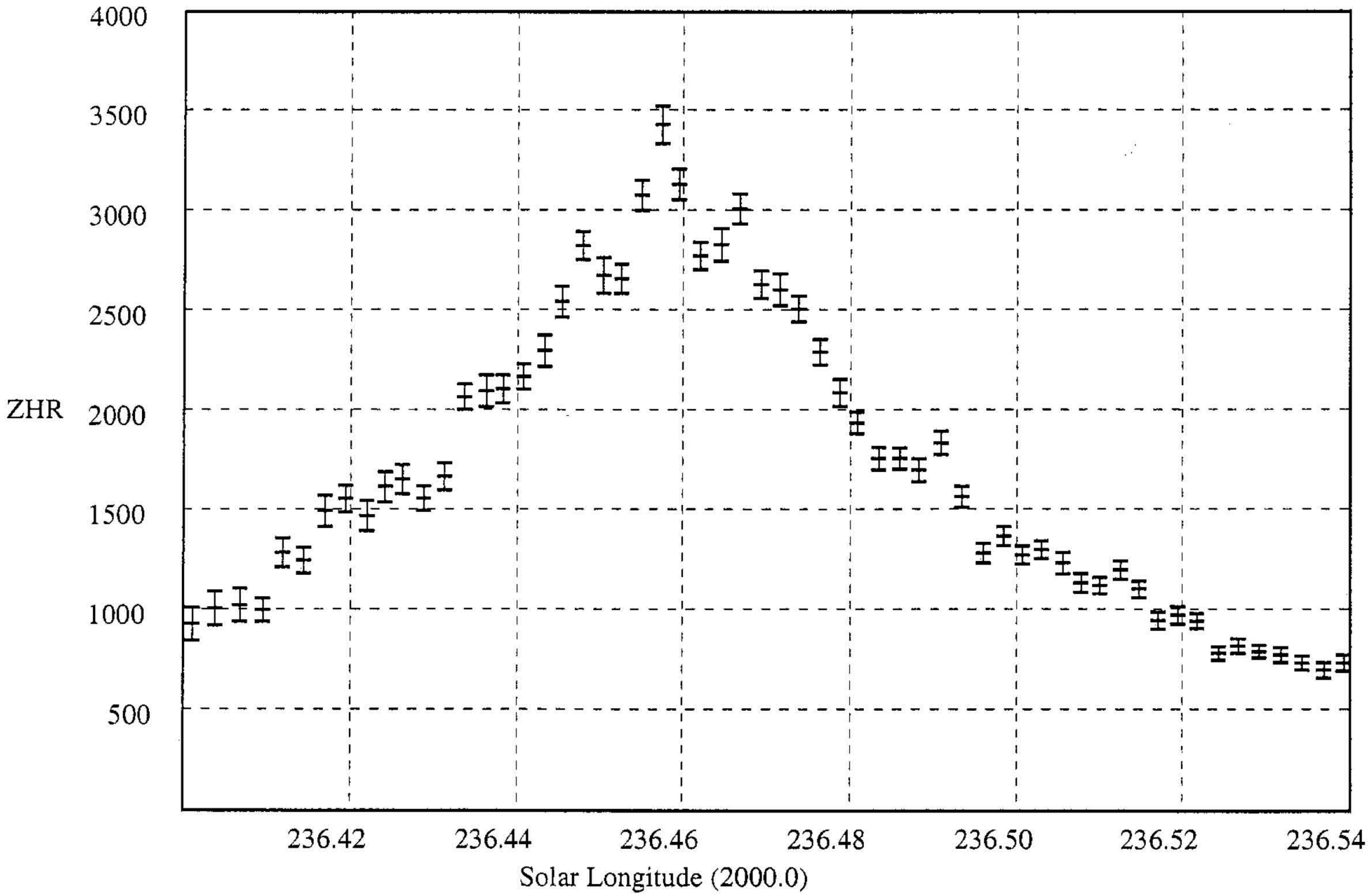
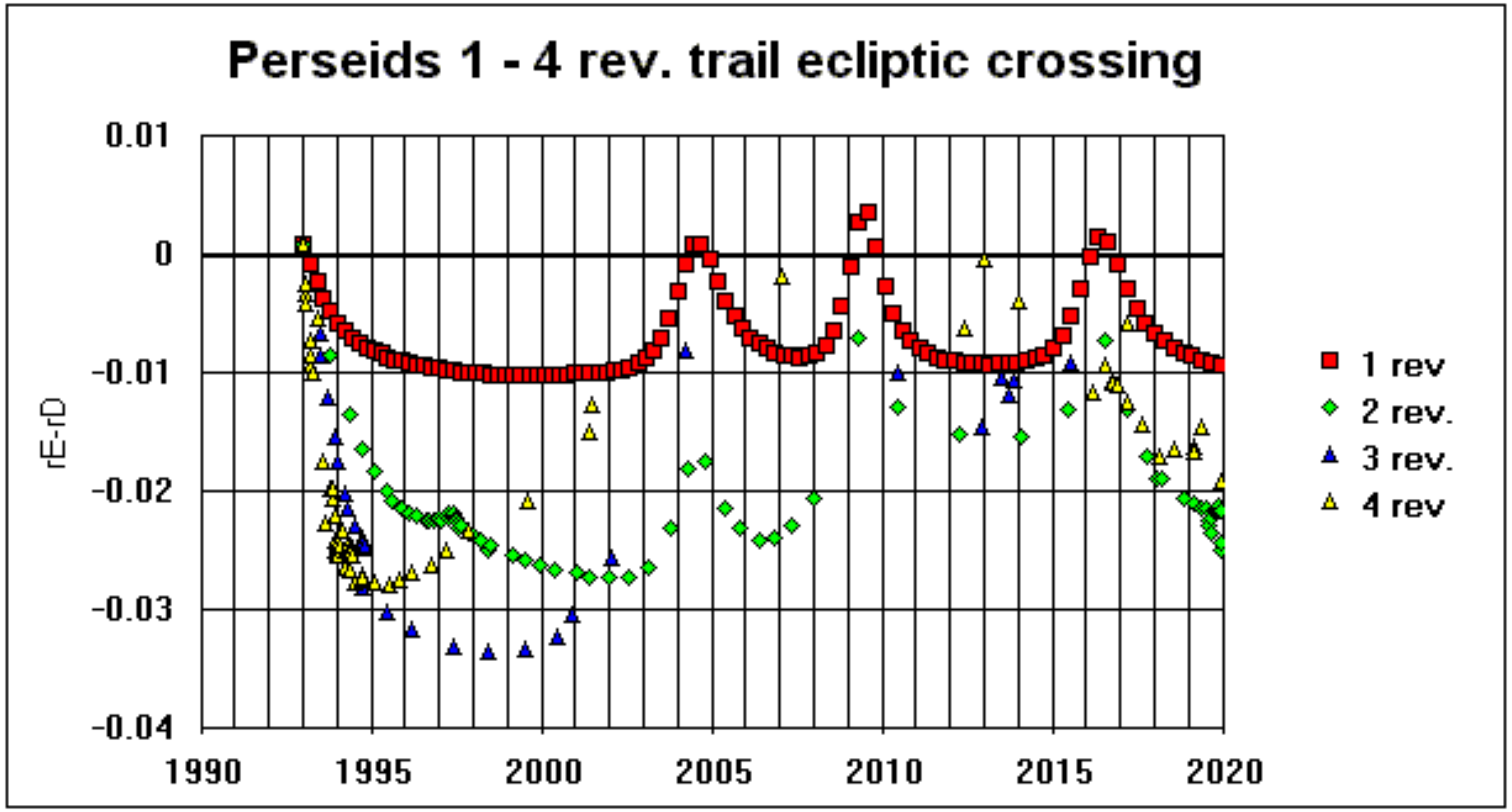


Figure 6 – Final profile of the Asian 2001 Leonid maximum. Observations with lm $\geq +5.8$ were used in the averaging procedure.

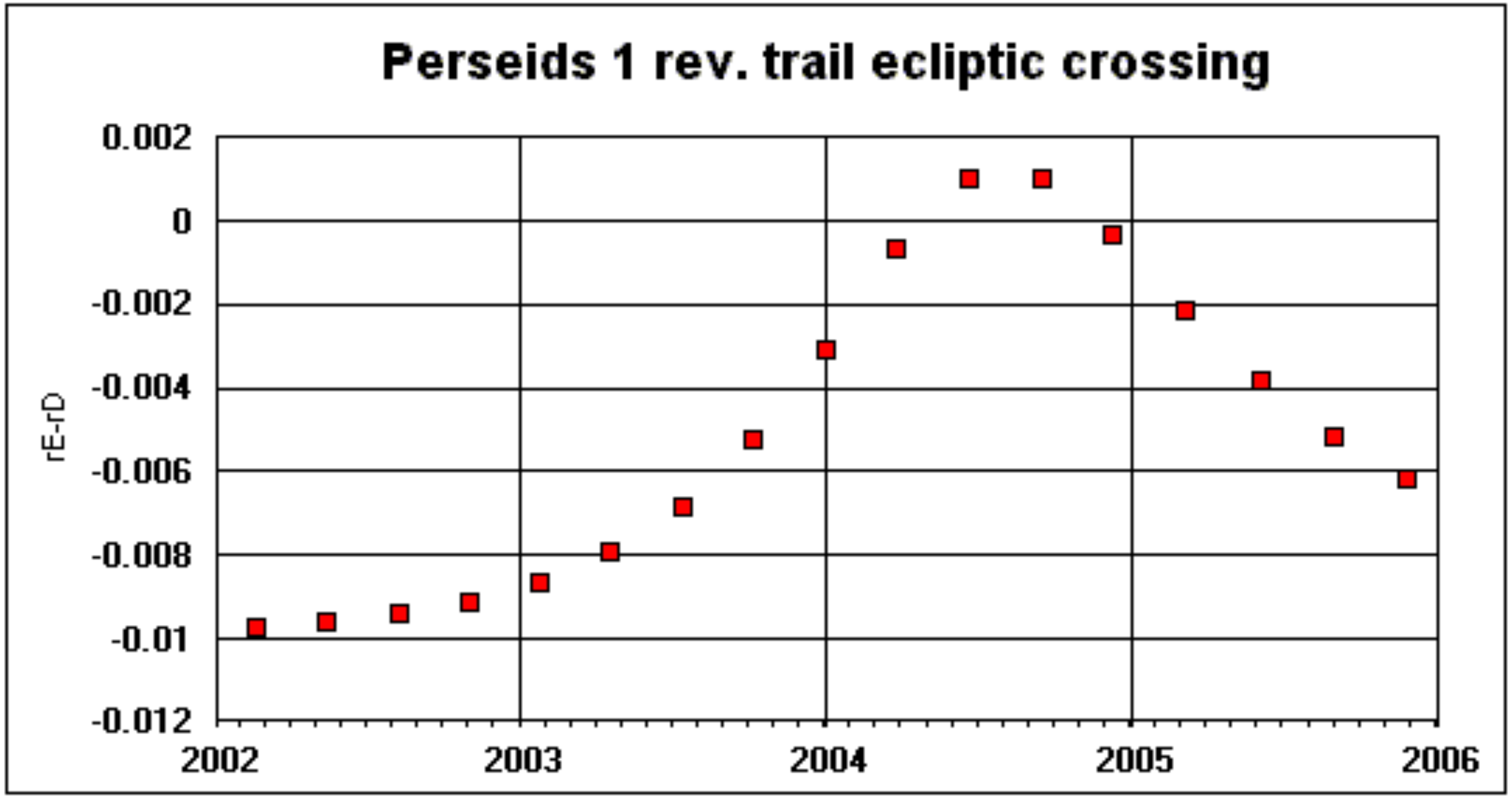
Perseidit

- Lähde: Esko Lyytinen et al.
- Kuvassa katsotaan y-akselin nollakohtaa ja kun vuoden elokuussa vana on nollakohdassa, voi Perseidien meteoriparvella olla suurempi aktiivisuus, kuin tavallisesti.
- Kuvassa on 1, 2, 4 ja 4 kierroksen vanat.
- Emokomeetta on 109P/Swift-Tuttle. Kiertoaika 133 vuotta.



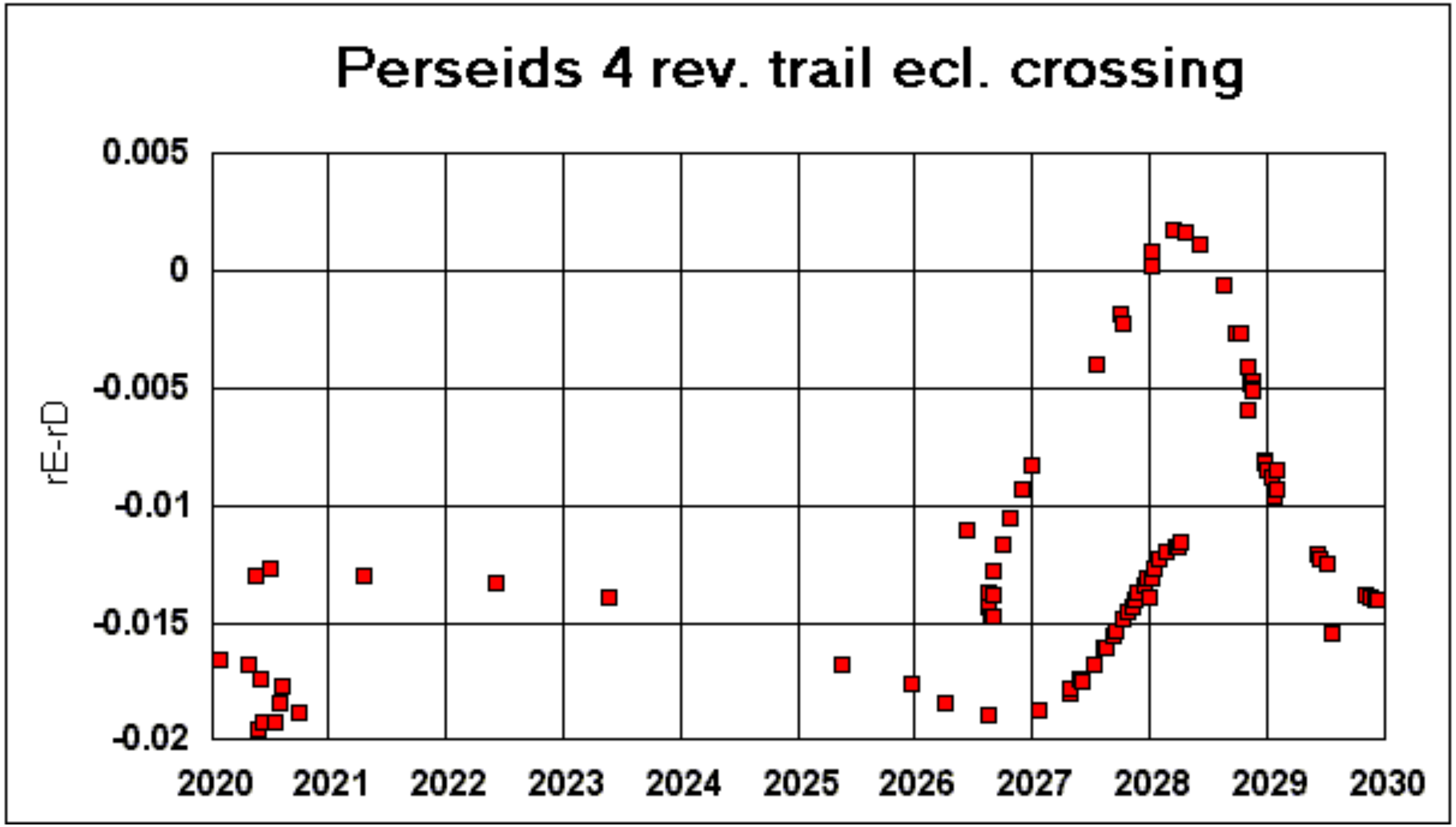
Perseidit 2004

- Lähde: Esko Lyytinen et al.
- yhden kierroksen pölyvana ohittaa Maan radan 0.0013 AU etäisyydeltä.
- Ennustettu maksimiaika oli vuodelle 2004 11.8 klo 21 UT.
- Myös vuotuinen aktiivisuus voi olla korkeammalla johtuen Jupiterin vaikutuksesta, 0.01 AU lähempää kuin tavallisesti.
- vuotuinen maksimi 12.8 11 – 13:20 UT.



Perseidit tulevaisuudessa

- Lähde: Esko Lyytinen et al.



Yhteenveto

- Esko Lyytisen partikkelimallilla on pitkä ja menestyksellinen historia ja se on tuottanut hyviä tuloksia useilla meteoriparvilla. Se soveltuu lähinnä 109P/Swift-Tuttle -, ja 55P/Tempel-Tuttle -tyyppisille komeetoille. Tähän oli tulossa muutos jo Eskon vielä eläessä ja nyt toteutetaan laajempi mallinnus tältäkin osin.

- Orekit -kirjasto tuo tavallaan mallinnusmenetelmän ns. tähän päivään mahdollistaen mallin edelleenkehittämisen ja yhteistyön ja standardoi menetelmät nykyaikaisen standardoidun ohjelmistotyökalun avulla.

Hyödyllisiä linkkejä

- https://www.taivaanvahti.fi
- https://www.ursa.fi/meteorit/yleista-ryhmasta.html
- https://www.ursa.fi/wiki/Meteoriopas/Meteoriopas
- https://www.ursa.fi/wiki/Meteoriopas/Havaintolomake
- https://www.ursa.fi/wiki/Meteoriopas/HavaintolomakkeenTayttoohje
- https://www.ursa.fi/wiki/Meteoriopas/RajasuuruusluokanMaarittelyalueet
- https://www.ursa.fi/wiki/Meteoriopas/KarttalehdetLiite