ASTRA®

Performance of Kalman Filter versus Least Squares estimator for orbit determination using trilateration

Jean-Philippe Halain

Faculté des Sciences Appliquées

Université de Liège

May 22, 1998

Société Européenne des Satellites, Château de Betzdorf
L - 6815 Betzdorf, Grand-Duché de Luxembourg

A Frédéric Krier

Acknowledgements

I am very grateful to the Société Européenne des Satellites to have given me all facilities for my work, the Flight Dynamic Team for their welcome, and especially MM. Dr. Philippe Francken and Dr. Georges Krier who make this work possible.

I also would like to thank the department of M. Prof. Beckers, of the Faculté des Sciences Appliquées de l'Université de Liège, and especially MM. Andry and Cugnon for their computer assistance during this year.

Table of Contents

Acknowledgements 4
Table of Contents 5
Task Description 9
List of Symbols 10

Chapter 1. Statement of the problem 11

1.1. Propagation and observation models 11
1.2. Orbit determination and prediction 12
1.3. Least Squares and Kalman estimators 12
1.4. Monte Carlo Simulations 13
1.5. Structure of this work 14

Chapter 2. Satellite orbit and tracking data 15

2.1. Satellite orbit 15
2.1.1. Newtonian movement 15
2.1.2. Keplerian movement 15
2.1.3. Orbital and equinoctial elements 16
2.1.4. Perturbations 17
2.1.5. Propagation model 21
2.2. Tracking data 22
2.2.1. Time notions 22
2.2.2. Coordinate transformations 23
2.2.3. Tracking systems 27
2.2.4. Observation model 30

Chapter 3. Estimation and filtering methods 31

3.1. Introduction 31
3.1.1. Models 31
3.1.2. Linearisation about a reference trajectory 31
3.1.3. Discretisation of the linearised equations 32
3.2. Batch Least Squares Estimation 33
3.2.1. Maximum likelihood estimator 33
3.2.2. Normal equations 35
3.2.3. Least Squares estimate and Covariance matrix 36
3.2.4. Practical considerations 36
3.2.5. Recursive Least Squares estimation 41
3.3. Kalman Filter 44
3.3.1. Linearised discrete Kalman Filter 44
3.3.2. Extended discrete Kalman Filter 45
3.3.3. Extended continuous-discrete Kalman Filter 46
3.3.4. Interpretation of the position uncertainty 47
3.3.5. Factorisation of the filtering equations 49
3.3.6. Filter divergence 50

Chapter 4. The simulation software design and description 51
4.1. Estimations and artificial tracking data 51
4.1.1. The Orbit and Kalman processes 51
4.1.2. The Tracking Data Simulator process 52
4.2. Simulator process 52
4.3. Process analysis 54
4.4. General software design and description 54
4.4.1. Data Flow Diagram 54
4.4.2. Data Dictionary 61
4.4.3. Command procedure description 64
4.4.4. Call tree 65
4.4.5. Environment 66
Chapter 5. Statistical analysis 67
5.1. Statistical analysis of results 68
5.1.1. Summary statistics 68
5.1.2. Elementary sampling theory 68
5.1.3. Central limit theorem 69
5.1.4. Statistical estimation 69
5.1.5. Statistical decision, test of hypotheses 71
5.1.6. Adjustment tests, normality tests 72
5.1.7. Monte Carlo Methods 74
5.2. The analysis post process 75
5.2.1. Analysis of the MCSIM results 75
5.2.2. MCSTAT design and description 77
5.2.3. EXCEL application description 81
Chapter 6. Monte Carlo simulations 83
6.1. Analysis plan 84
6.1.1. Analysis with measurement errors 84
6.1.2. Analysis with dynamic errors 89
6.1.3. Simulation plan 91
6.2. Simulation results 92
6.2.1. Results of a typical simulation 92
6.2.2. Error model effects 95
6.2.3. Normality results 98
6.2.4. Bias testing 99
6.2.5. EKF and LS behaviour 101
6.2.6. Dynamic model effect 106
6.2.7. Overview of results 110
6.3. Supplementary investigations 110
6.3.1. Variation of the process noise level 110
6.3.2. Non linearity response 111
6.3.3. Adaptive behaviour of the EKF 112
Chapter 7. Conclusions 115

Annexes. 119
A.1. Hierarchical design of the simulator 119
A.2. The MCSIM Command File 127
A.3. User manual 135
A.4. Test plan of the simulator 137
A.5. Input files 138
A.6. Output files 141
A.7. Simulation plan tables 142
A.8. Hierarchical design of the post process 145

Summary. 149

Bibliography. 151

Task Description

The *Société Européenne des Satellites* (SES) has in charge 7 geostationnary satellites, 6 of which being co-located. All satellites are controlled in real-time from Betzdorf (Luxembourg) and tracked from different antenna systems. These tracking measurements are used to realise a precise determinations of spacecraft trajectories, which are used to subsequently compute the required orbit correction manoeuvres.

The software *ORBIT* [Mon 96] of the *German Space Operation Center* (GSOC) is used by SES to realise these orbit determinations. It is based on a *Least Squares* method which processes observations through a dynamic model and an observation model, and provides estimation of the satellites positions and velocities. Ordinarily, the tracking system consists in a single antenna providing one range and two angles measurements. Nevertheless, as range measurements are much more accurate than angle measurements, SES has recently developed a *trilateration* tracking system using four ranging stations. In addition, both the single- and multi-stations tracking systems developed by SES offer the advantage of providing a continuous flow of data.

The dynamic and observation models are only accurate to a certain level as it is not possible to exactly model all forces acting on the satellite and all effects perturbing measurements, so additive observation noise and unmodeled forces are generally taken into account. However, the improvement of the tracking system realised by SES allows a significant decrease of the observation errors and thus reinforces the relative importance of forces not modelled in the propagation model.

In parallel to the tracking system improvement, SES has explored a second method of orbit determination, the *Kalman Filtering*. This method is suited to handle noise in the dynamic propagation model, contrary to the Least Squares method, and allows a sequential treatment of the tracking data. The Kalman Filter has been implemented last year at SES [Wel 97], based on the ORBIT software.

The purpose of the present work is to compare the implemented Least Squares and Kalman Filtering methods when measurements are acquired via a trilateration tracking system. For this purpose a simulator process has been developed. It allows to compare the differences between a reference trajectory and estimations obtained with the two methods, for different observation and propagation models. Statistical analysis of the results based on Monte Carlo methods is applied to compare the evolution of the Least Squares and Kalman Filter estimates, and determine if one of these methods is better suited for trilateration measurements.

Additionally, an article for the 13th International Symposium on Space Flight Dynamics at the Goddard Space Flight Center, NASA, has been published [Hal 98]. It extends the present comparison to both single- and multi-stations tracking systems.

List of Symbols.[1]

$\mathbf{r}$	Cartesian position vector (3 components)
$\mathbf{x}$	state vector (6 components)
$\mathbf{y}$	trajectory vector, state vector & estimated parameters (m components)
$\mathbf{z}$	observation vector (n components)
$\mathbf{w}$	propagation noise vector (m components)
$\mathbf{v}$	observation noise vector (n components)
$\mathbf{u}$	control vector (3 components)
t	time variable
$\mathbf{P}$	estimation error covariance matrix
$\mathbf{Q}$	propagation noise covariance matrix
$\mathbf{I}$	identity matrix
$\mathbf{0}$	zero matrix
$\delta(.)$	Dirac delta function
$E[.]$	expectation operator
$p(.)$	probability density function
$p(.\|.)$	conditional probability density function
Pr	probability of an event
$tr(.)$	trace of a matrix
$\overset{def}{=}$	by definition
$\lVert\, . \,\rVert$	Euclidean norm
$\lvert\, . \,\rvert$	determinant of a matrix
$(\ )^T$	vector or matrix transposed
$(\ )^{-1}$	scalar or matrix inverse
$\dfrac{\partial \mathbf{y}}{\partial \mathbf{x}}$	partial derivative of $\mathbf{y}$ with respect to $\mathbf{x}$
$\dfrac{d}{dt}$ *or* $\dot{}$	derivation with respect to time
$\hat{\mathbf{x}}$	estimate of $\mathbf{x}$
$\hat{\mathbf{x}}(t\|t_l)$	estimate of $\mathbf{x}$ at the time t given observations through time t_l
$\mathbf{P}(t\|t_l)$	conditional covariance matrix at t given observations through t_l

[1] Letters in bold denotes vectors or matrices.

Chapter 1.

Statement of the problem

1.1. Propagation and observation models

The propagation model

The *propagation model* describes mathematically the evolution of a body submitted to different external forces. In orbital dynamics, one generally considers the movement of a satellite as a mass point whose movement is described by classical mechanics, the mass being negligible relatively to the masse of planets. The satellite movement is thus described by a first order differential equation for the 6-components state vector $\mathbf{x}(t)$. This *dynamic equation* can be written

$$\dot{\mathbf{x}}(t) = \mathbf{f}\left[\mathbf{x}(t), \mathbf{p}(t), t\right] + \mathbf{w}(t) \qquad (1.1)$$

where $\mathbf{p}(t)$ contains the parameters of the dynamic model (acceleration vector of eventual manoeuvres, solar radiation pressure coefficients,...), and $\mathbf{w}(t)$ is the *process noise* and represents all unmodeled forces.

Due to the random acceleration term $\mathbf{w}(t)$, the state vector is itself a random variable which is defined by its probability density function $p(\mathbf{x}(t))$. When all forces acting on the body are supposed to be completely known, this term is null and the orbit is determined by the only six components of the state vector at time t_0 and parameter values.

The observation model

The measurements may also be described by mathematical relations, called the *observation model*. It consists in a relation between the measurements and the state vector and may be represented by the following equation, called *measurement equation*

$$\mathbf{z}(t_k) = \mathbf{h}\left[\mathbf{x}(t_k), \mathbf{q}, t_k\right] + \mathbf{v}(t_k) \qquad (1.2)$$

where $\mathbf{z}(t_k)$ is the observation vector containing M types of measurement at time t_k ($k = 1, \ldots, N$), $\mathbf{v}(t_k)$ represents an additive zero-mean Gaussian white noise due to the imperfection of tracking measurement system, and $\mathbf{q}$ contains the different parameters of this model.

Note : The statistics of dynamic and observation noises are supposed to be known.

1.2. Orbit determination and prediction

Orbit propagation

The problem of orbit propagation consists in computing, for any time *t*, the solution of the equation **(1.1)**, assuming that the parameter vector **p** and an initial conditions $\mathbf{x}(t_k)$ are perfectly known.

Orbit determination

When the term $\mathbf{w}(t)$ is null, the problem of orbit determination consists in computing the minimum error estimate of a vector $\mathbf{y}(t_0) = \{\mathbf{x}(t_0), \mathbf{p}, \mathbf{q}\}$ using the set of $N \times M$ time-discrete measurements $z_i(t_k)$, $i = 1,\ldots,M$, influenced by random errors. This problem is a *parameters estimation* problem, and is generally solved by a Least Squares algorithm.

With the additive process noise $\mathbf{w}(t)$, the orbit determination consists in a recursive determination of an estimate $\mathbf{y}(t_k)$, using a set of *k* time-discrete observations $z_i(t_k)$, with $t_1 \le t_2 \le \ldots t_k$, influenced by random errors. This problem is known as a *filtering* problem, and was first solved by Kalman [Kal 60] for discrete linear systems. The combination of continuous propagation equations and discrete observations, with the applications of the linear theory to non-linear filtering problems have been discussed by Jazwinski [Jaz 70] and Gelb [Gel 74].

1.3. Least Squares and Kalman estimators

Both Kalman Filter and Least Squares algorithms are by essence *linear estimators* which, by linearisation around a *reference trajectory*, are applied to the non-linear estimation problem. The optimality criterion in both cases is the minimisation of the mean squared estimation error. The associated conditional error covariance matrix

$$\mathbf{P}\left(t\middle|t_k\right) \overset{def}{=} E\left[\left(\mathbf{y}(t) - \hat{\mathbf{y}}\left(t\middle|t_k\right)\right)\left(\mathbf{y}(t) - \hat{\mathbf{y}}\left(t\middle|t_k\right)\right)^T\right] \tag{1.3}$$

which is computed for both parameter estimation and filtering, may be used practically to formulate this optimality criterion

$$\begin{aligned}
&\min E\left\|\mathbf{y}(t) - \hat{\mathbf{y}}\left(t\middle|t_k\right)\right\|^2 \Leftrightarrow \min E\left(\left\{\mathbf{y}(t) - \hat{\mathbf{y}}\left(t\middle|t_k\right)\right\}^T \left\{\mathbf{y}(t) - \hat{\mathbf{y}}\left(t\middle|t_k\right)\right\}\right) \\
&\Leftrightarrow \min E\left(tr\left\{\mathbf{y}(t) - \hat{\mathbf{y}}\left(t\middle|t_k\right)\right\}\left\{\mathbf{y}(t) - \hat{\mathbf{y}}\left(t\middle|t_k\right)\right\}^T\right) \\
&\Leftrightarrow \min tr\left\{\mathbf{P}\left(t\middle|t_k\right)\right\}
\end{aligned} \tag{1.4}$$

The *Iterated Least Squares* (LS) and the *Extended Kalman Filter* (EKF), presented in **Chapter 3**, constitute the theoretical basis of both implementations used at SES. If both algorithms are mathematically equivalent in the linearised formulation without propagation noise, they nevertheless differ in the way they cope with the non-linearity's of the equations of motion. LS proceeds by iterating the linearisation process several times over the whole series of tracking data, whereas the EKF performs a new linearisation at each new time step. The LS algorithm which treats the whole batch of observations at each iteration is thus called a *batch estimator*, while the Kalman algorithm which processes observations one by one is called a *sequential estimator*.

Another major practical difference is that the LS algorithm estimates the state vector at a fixed epoch, the estimate and its covariance matrix being subsequently propagated to other times by numerical propagation. On the contrary, the EKF algorithm provides an estimate at each measurement time as it processes sequentially new observations. This way of proceeding is well suited for real-time software applications where the actual filter update is triggered by the arrival of a new measurement and where the constant knowledge of the instantaneous state of the system is the only information needed to be maintained.

Because of the iterative nature of the LS algorithm, it may be expected to be less effective in terms of computational load. However, it can use multi-steps numerical integrators more efficiently since the reference trajectory remains the same over the whole batch of observations at each iteration. On the other hand, the EKF has to restart the integrator at each measurement update time, and restarting a self-starting integration algorithm may require an amount of computational effort which outweighs the gains realised by using a single-pass method.

A great advantage of the Kalman Filter lies in its ability to take process noise into account. If in most applications the performance of computers allow a precise modelling of the orbital motion of geostationnary satellites, so that neglecting unmodeled forces in the LS process is fairly well justified, in satellite using thrusting devices over long periods of time may require the incorporation of process noise to reflect the randomness of this thrust.

1.4. Monte Carlo Simulations

The true trajectory of a satellite is never exactly known. In order to compare the estimation errors of the implemented LS and EKF algorithms, we must compute differences between a *reference trajectory*, which is supposed to be a true trajectory This trajectory is observed via the observational function **h** and various additive observation noise, the batch of corresponding observations being processed by both algorithms, and outgoing estimations being compared to the reference trajectory.

To realise a correct comparison of both methods, a dedicated *simulator* which realises the above operations must be used in a systematic and automatic approach which allows *Monte Carlo simulations*, where statistical properties of the system are analysed from a great number of simulated realisations.

Important note :

The reference trajectory of these simulations has not the same meaning as the one used for linearisation of the non-linear model equations. The former is generated independently of estimations and is only used to construct artificial tracking data and realise comparisons, while the latter is used for linearisation in the estimation algorithms.

1.5. Structure of this work

This diploma thesis is composed of three main parts.

- The theoretical description of both propagation and observation models in **Chapter 2**, and both LS and Kalman Filter methods are described in **Chapter 3**. They are based on lectures of several authors as [Roc 98], [Soo 94], and [Zar 87] for models, and [Bra 93], [Fra 98], [Gre 93], [Boz 79] and [Wel 97] for estimation methods.
- The simulation process and statistical analysis post process are described in **Chapter 4** and **Chapter 5** respectively. The simulation software uses different notions described in [Mon 96], [Fra 96] and [Wei 97], and statistical analysis is based on considerations coming from [Fra 92], [Dag 92], [Spi 61], [Ban 95] and [Wau 95].
- The simulation results and conclusions presentation in **Chapter 6** and **Chapter 7**, and detailed simulation cases are presented in **Annexes**.

Note :

The description of both models in **Chapter 2** is given for the completeness of the work. Nevertheless, they are not necessary to understand the comparison of both estimation methods if one accepts (**1.1**) and (**1.2**) as dynamic and observation models. In that case, the reading of this chapter may be skipped.

Chapter 2.

Satellite orbit and tracking data

2.1. Satellite orbit

In this part, the dynamic model is detailed through the development of both sides of equation (**1.1**).

2.1.1. Newtonian movements

Earth orbiting satellites are generally considered as negligible mass points, whose position vector is $\mathbf{r} = (x, y, z)^T$ in inertial geocentric coordinates. The satellite is submitted to forces and its movement follows the Newton second law

$$m\ddot{\mathbf{r}} = \sum \mathbf{F} \qquad (2.1)$$

This second order equation may be written as the first order differential equation

$$\frac{d}{dt}\begin{pmatrix} \mathbf{r}(t) \\ \dot{\mathbf{r}}(t) \end{pmatrix} = \begin{pmatrix} \dot{\mathbf{r}}(t) \\ m^{-1}\sum \mathbf{F} \end{pmatrix} \qquad (2.2)$$

where $\mathbf{x}(t) \overset{def}{=} \begin{pmatrix} \mathbf{r}(t) \\ \dot{\mathbf{r}}(t) \end{pmatrix}$ is the *state vector* containing position and velocity vectors, m the mass of the satellite and $\sum \mathbf{F}$ the sum of all forces acting on the satellite [Zar 87].

The solution of this equation gives, with initial conditions and all forces completely known, the state vector at any time t, that is the *orbit* of the satellite.

2.1.2. Keplerian movements

For a perfect spherical Earth gravitational force acting on the satellite, the right-hand side of the second Newton law simplifies in a one body central force

$$\mathbf{F}(t) = -\frac{m\mu}{|\mathbf{r}(t)|^3}\mathbf{r}(t) \qquad (2.3)$$

where $\mu = GM_{Earth}$, G being the universal gravitation constant and M_{Earth} the mass of the Earth. This force derives from the potential $U = -\frac{\mu}{r}$, and the trajectory generated by the resulting second order differential equation is a *conic*, that is an ellipse, a parabola or an hyperbola depending on the initial conditions [Roc 98]. This movement is called Keplerian and has the following properties

- It is plane, and its kinetic moment **C** is constant because $\dot{\mathbf{C}} = \frac{d}{dt}\left(\mathbf{r} \wedge \dot{\mathbf{r}}\right) = 0$, where $\boldsymbol{r}$ is the position vector whose origin is at centre of the Earth.
- The *aerial speed* is constant $\frac{dA}{dt} = \frac{1}{2} r^2 \dot{\theta} = \frac{C}{2}$, where θ is the angle between **r** and an axe in the plane of the movement.
- The variables r and θ are related throughout the *orbit equation*

$$\frac{1}{r} = \frac{\mu}{C^2}\left[1 + e\cos(\theta - \theta_0)\right] \tag{2.4}$$

  where e and θ_0 are integration constants. In the case of an ellipse, e is negative and this equations may be written as $\frac{p}{r} = 1 + e\cos(\theta)$, where $p = \frac{C^2}{\mu}$.
- The conic is determined by the conservation of energy equation

$$V^2 - \frac{2\mu}{r} = -\frac{\mu}{a} \tag{2.5}$$

  where V is the velocity and $a = \frac{p}{(1-e^2)}$
- The conic is an ellipse, a parabola or an hyperbola when e is positive, null or negative.

2.1.3. Orbital and equinoctial elements

When the orbit is a non-perturbed ellipse, the movement is periodic and can be described by six parameters, the **classical orbital elements**. They correspond to the ellipse equation

$$r = \frac{p}{1 + e\cos(\theta)} = a(1 - e\cos E) \tag{2.6}$$

whose orbiting period is $T = 2\pi\sqrt{a^3/\mu}$.

- a is the *semi-major axis* of the ellipse, and characterises its size.
- e is the *eccentricity* defined as $e = \frac{\sqrt{a^2 - b^2}}{a}$, and characterises the orbit shape.
- i is the *inclination* of the orbit plane, and represents the angle between the orbit and the equatorial plane.
- Ω is the *right ascension of the ascending node*, and represents the position of the intersection of these two planes.
- ω is the *argument of the perigee*, and gives the position of the perigee of the orbit.
- E is the *eccentric anomaly*, and gives the position of the satellite at one time of its period. It can be replaced by M which is the *mean anomaly*, and is defined as $M = E - e\sin E$.

We have thus the six parameters $\{a,\ e,\ i,\ \Omega,\ \omega,\ M\}$ to define a satellite orbit. In the two body problem (*i.e.* Earth and satellite), it is possible to make a transformation between them and the components of the Cartesian state vector

$$\{a,e,i,\Omega,\omega,M\} \longleftrightarrow \{x,y,z,\dot{x},\dot{y},\dot{z}\} \qquad (2.7)$$

2.1.4. Perturbations

When the trajectory is perturbed, orbital parameters can also be defined as in **(2.7)**, but they vary with time and are called **osculatory elements**. One generally uses these parameters rather than the components of the state vector because of their intuitive meaning for orbit perturbation.

Perturbations of a geostationnary satellite trajectory are due to other forces than the point Earth attraction. It is thus necessary to include them into the equation of motion. The more forces are taken into account, the better the propagation model, *i.e.* the equation of motion. Nevertheless, it is not possible to model all forces acting on the satellite, and a last force term containing all unmodeled forces may be introduced. In that way, the second member of the Newton second order differential equation becomes

$$\sum \mathbf{F} = \mathbf{F}_{\text{Earth}} + \mathbf{F}_{\text{Sun}} + \mathbf{F}_{\text{Moon}} + \mathbf{F}_{\text{Sol.Rad}} + \mathbf{F}_{\text{Thrusts}} + \mathbf{F}_{\text{Unmodeled}} \qquad (2.8)$$

where the attraction of the Earth is more general than a simple two body attraction, and where the other main perturbing terms are due to the Sun and Moon attractions, the effects of atmosphere, the solar radiation pressure, and the unmodeled forces. These perturbations may be divided into two classes, the gravitational and non gravitational ones.

i. Gravitational perturbations

The gravitational accelerations acting on a satellite only depend on the mass distribution around it. They all have a common origin, the Newtonian force

$$f = \frac{GmM}{r^2} \qquad (2.9)$$

They may be written in the conservative potential form

$$\mathbf{f} = \nabla U \qquad (2.10)$$

where $U(r) = \dfrac{GM}{r}$.

- *Earth geopotential, gravity field of an aspherical body*

The first important force perturbing the orbit is due to the anisotropy of the main attracting body, i.e. the Earth here. This anisotropy may be reflected by the decomposition of its gravitational potential in a spherical harmonic form

$$U(\mathbf{r}) = \frac{GM}{\mathbf{r}} \sum_{n=0}^{\infty} \sum_{m=0}^{n} \frac{R^n}{r^n} P_{nm}(\sin\phi)\left[C_{nm}\cos(m\lambda) + S_{nm}\sin(m\lambda)\right] \tag{2.11}$$

where the two groups of numbers C_{nm} and S_{nm} are given by the integrals

$$\begin{cases} C_{nm} = \dfrac{2}{M}\dfrac{(n-m)!}{(n+m)!}\displaystyle\int \frac{s^n}{R^n} P_{nm}(\sin\phi')\cos(m\lambda')\rho(\mathbf{s})d^3\mathbf{s} \\ S_{nm} = \dfrac{2}{M}\dfrac{(n-m)!}{(n+m)!}\displaystyle\int \frac{s^n}{R^n} P_{nm}(\sin\phi')\sin(m\lambda')\rho(\mathbf{s})d^3\mathbf{s} \end{cases} \tag{2.12}$$

and are called *unnormalised geopotential coefficients*, r, ℓ and ϕ are the geocentric distance, longitude and latitude, R and M the mean equatorial radius and mass of the attracting body, P_{nm} are Legendre Functions, and $\rho(\mathbf{s})$ is the mass distribution function.

A perfect propagation model would contain all these terms. Nevertheless, a good model generally only includes a maximum of 30 terms.

The coefficients with $m = 0$ are the *zonal coefficients*, and describe the part of the potential which does not depend on the satellite's longitude. All the S_{n0} coefficients vanish whereas C_{n0} coefficients are renamed $-J_n$. The other coefficients are known as *tesseral coefficients* for $m < n$, and *sectorial coefficients* for $m = n$.

All these coefficients are usually time dependent because of the variations with time of the Earth mass distribution (oceanic and terrestrial tides). Nevertheless these variations are very small and Earth potential can be considered constant. One thus generally speaks about these coefficients in terms of their mean during a period T of observation, *i.e.* the stationary part of the terrestrial potential

$$\begin{cases} C_{nm} = \dfrac{1}{T}\displaystyle\int_0^T C_{nm}(t)dt \\ S_{nm} = \dfrac{1}{T}\displaystyle\int_0^T S_{nm}(t)dt \end{cases} \tag{2.13}$$

- *Sun and Moon potentials*

The second and third important perturbations are due to the Sun and Moon mass point attractions. The resulting accelerations in a frame centred on the Earth's centre of mass are given by

$$\begin{cases} \ddot{\mathbf{r}}_{Sun} = GM_{Sun}\left(\dfrac{\mathbf{r}_{Sun} - \mathbf{r}}{\left|\mathbf{r}_{Sun} - \mathbf{r}\right|^3} - \dfrac{\mathbf{r}_{Sun}}{\left|\mathbf{r}_{Sun}\right|^3}\right) \\ \ddot{\mathbf{r}}_{Moon} = GM_{Moon}\left(\dfrac{\mathbf{r}_{Moon} - \mathbf{r}}{\left|\mathbf{r}_{Moon} - \mathbf{r}\right|^3} - \dfrac{\mathbf{r}_{Moon}}{\left|\mathbf{r}_{Moon}\right|^3}\right) \end{cases} \tag{2.14}$$

where M_{Sun} , M_{moon} and $\mathbf{r}_{Sun}$, $\mathbf{r}_{Moon}$ are respectively the mass and position of the Sun and Moon.

Note : The effects of the other planets are negligible so their contribution is included in the unmodeled force term.

ii. Non gravitational perturbations

The non gravitational perturbation forces do not depend on the mass of the satellite, the corresponding accelerations are thus inversely proportional to this mass. The most important non gravitational accelerations of geostationnary satellites are due to the Solar Radiation Force (SRF) and the thrust forces imposed for manoeuvres. The SRF perturbations is due to an exchange of impulsion at the satellite surface and is called *surface force*. Moreover these perturbations are modelled in satellite local axes.

- *Solar radiation pressure*

This perturbation is due to the absorption and reflection by the satellite surface of photons coming directly from the Sun or reflected by the Earth, the albedo effect being generally neglected for geostationnary orbits .

For a surface receiving photons and absorbing them perpendicularly to the surface, there is a *variation of the momentum* given by

$$\frac{dP}{dt} = \frac{S}{d^2}\frac{\phi}{c} \qquad (2.15)$$

where d is the distance between the satellite and the Sun, c the speed of light, S is the surface of the satellite, and ϕ is the power of the incident photon flux. It represents a pressure of $p_{0,abs} = \frac{1}{S}\frac{dP}{dt} = \frac{\phi}{d^2 c} \cong 4.5 10^{-6}\, N/m^2$ for satellite at the distance $d = 1AU \cong 149.6 10^6\, km$ of the Sun, where $\phi \cong 3.02 10^{25}\, W/Steradian$.

A part of the absorbed photons are re-emitted and then produce a supplementary variation of the momentum. Indeed only a part of the incident photons are absorbed, the rest is reflected. For a completely reflecting surface, the impulse transferred in a direction perpendicular to the surface would be twice as large as for complete absorption, $\mathbf{F}_{tot\_refl} = 2\mathbf{F}_{tot\_abs} = 2p_{tot\_abs}A\mathbf{e}$ where $\mathbf{e}$ is a unit vector pointing from the Sun to the satellite.

In reality the surface only reflects a part ε of the incoming radiation, and the perturbation force is

$$\mathbf{F} = p_{abs}A(1+\varepsilon)\mathbf{e} \qquad (2.16)$$

where ε depends on the satellite surface material. So the perturbation acceleration due to the solar radiation pressure is

$$\ddot{\mathbf{r}}_{Sol.Rad} = -C_R\frac{S}{m}p_{abs,0}\frac{\mathbf{r}_{Sun}}{|\mathbf{r}_{Sun}|^3}.(AU)^2 \qquad (2.17)$$

where $C_R = 1+\varepsilon$ is the *radiation pressure coefficient*, $\mathbf{r}_{\mathrm{Sun}}$ is the Sun vector position, and AU is the astral unity.

Formula (**2.17**) applies to satellites with a constant Sun-facing surface. Alternatively for a surface facing the Sun perpendicularly to the equator, it is necessary to multiply the whole expression by $\cos(\theta_{Sun})$ who renders the variation of the cross-sectional area due to the variation of the Sun declination θ_{Sun}. In that case the solar radiation pressure model becomes

$$\ddot{\mathbf{r}}_{Sol.Rad} = -C_R \frac{S}{m}\cos(\theta_{Sun})p_{abs,0}\frac{\mathbf{r}_{Sun}}{|\mathbf{r}_{Sun}|^3}.(AU)^2 \qquad \textbf{(2.18)}$$

Both models account for the dependence of the solar radiation pressure from the Sun-Earth distance which leads to a seasonal variation of about 3%, and eclipses are treated assuming a conical shadow modelling with umbra and penumbra shadow regions.

In addition to the main radiation pressure component of these models, which always points away from the Sun, the perpendicular component can also be taken

$$\ddot{\mathbf{r}}_{Sol.Rad.\perp} = -C_{R_p}\frac{S}{m}p_{abs,0}\frac{\mathbf{e}_{Sun\perp}}{|\mathbf{r}_{Sun}|^2}.(AU)^2 \qquad \textbf{(2.19)}$$

where $\mathbf{e}_{Sun,\perp}$ is a unit vector perpendicular to $\mathbf{r}_{\mathrm{Sun}}$ in the equatorial plane, and C_{R_p} is used to model the corresponding solar radiation pressure component.

- *Manoeuvres*

The orbit corrections of the satellite are considered as perturbations which can be modelled as

$$\ddot{\mathbf{r}} = \mathbf{E}(t).\mathbf{u}(t) \qquad \textbf{(2.20)}$$

where **u**(t) is the *acceleration thrust vector* in the satellite frame, and **E**(t) is the rotation matrix from the satellite reference frame to the inertial reference frame and depends of its orientation in space, that is its *attitude*.

One generally considers constant thrusts between the times t_{start} and t_{end}, so the corresponding acceleration is given by

$$\ddot{\mathbf{r}} = \begin{cases} \mathbf{0} & \text{for } \mathrm{t} < \mathrm{t}_{\mathrm{start}} \\ \mathbf{E}(t).\mathbf{u} & \text{for } \mathrm{t}_{\mathrm{start}} \leq \mathrm{t} < \mathrm{t}_{\mathrm{end}} \\ \mathbf{0} & \text{for } \mathrm{t} \geq \mathrm{t}_{\mathrm{end}} \end{cases} \qquad \textbf{(2.21)}$$

If the manoeuvre duration is negligible compared to the orbital period, it is said *impulsive* and may be described in term of velocity increment $\Delta\mathbf{v}$ and burn centre time t_{man} as

$$\ddot{\mathbf{r}} = \delta(t - t_{man})\Delta\mathbf{v}\,. \qquad \textbf{(2.22)}$$

iii. **Unmodeled forces**

Various forces are not taken into account in the propagation model the solar radiation pressure on antenna reflectors and some satellite body components, the variations of solar activity, the gravitational attraction of the other planets, the effect of solar wind, the forgotten terms in the geopotential of the Earth, and the relativistic effects. All these forces cannot be completely modelled, because it is not possible to know all parameters of possible models, and because it would require a finite-elements modelling of the geometry of the satellite.

The effect of all these unmodeled forces can nevertheless be included in an additive process noise, whose statistics is supposed to be white Gaussian.

This white Gaussian force $\ddot{\mathbf{r}}_{\text{unmodeled}} = \ddot{\mathbf{r}}_{\text{noise}} \overset{def}{=} \mathbf{w}(t)$ is defined by its mean and correlation functions

$$\begin{cases} E[\mathbf{w}(t)] = \mathbf{0} \\ E[\mathbf{w}(t)\mathbf{w}^T(\tau)] = \mathbf{Q}(t)\delta(t-\tau) \end{cases} \tag{2.23}$$

where **Q**(t) is the *spectral density matrix* by analogy with the power spectral density function which is the Fourier Transform of the correlation function., $\mathbf{w}(t)$ and $\mathbf{w}(\tau)$, $t \neq \tau$, being supposed not correlated.

2.1.5. Propagation model

The Newton equation of motion may be written in the functional form **(1.1)** in term of the state vector **x**(t) in an inertial coordinate system, where **f** includes all deterministic modelled forces described in paragraphs 2.1.1 to 2.1.3, and **w**(t) is the white Gaussian process noise described in paragraph 2.1.4 and which is assumed not correlated with the state vector **x**(t)

$$E[\mathbf{x}(t)\mathbf{w}(\tau)] = \mathbf{0} \quad \forall t,\tau \ . \tag{2.24}$$

Furthermore, the equation **(1.1)** is a stochastic differential equation with an additive Gaussian forcing function, and is called the *Langevin equation*.

The coordinate system used refers generally to the *Earth's Mean Equator and Equinox of J2000* (EME2000), where J2000 denotes the epoch of *Universal Time* (UT), 12^h on January 1, 2000. The integration of this equation is realised in this system, whereas the independent variable used for dynamic calculations is the in *Terrestrial Time* (TT), see paragraph 2.2.

The state vector **x** is a random multivariable which is assumed to be Gaussian itself. It is thus known at a certain time *t* with an uncertainty defined by its covariance matrix **P**(t), the effect of the additional random acceleration **w**(t) being to increase the elements of this matrix **P**(t) when the state vector is propagated using the propagation equation.

2.2. Tracking data

The modelling of ground based measurements requires the knowledge of the satellite position with respect to a *local topocentric frame*. This frame is defined with respect to an origin point on the solid Earth, and the local East, North and Zenith directions.

It is thus necessary to have transformations from the inertial EME2000 coordinates, wherein the integration of the equation of motion is carried out, to the local topocentric coordinates. With these transformations, measurements realised in the local topocentric frame can be related to an observation model, and used thereafter for orbit determination.

2.2.1. Time notions

The transformation between terrestrial and celestial reference frame requires a detailed knowledge of the Earth's rotation. In view of know irregularities in the Earth's rotational motion, a continuous monitoring is coordinated by the *International Earth Rotation Service* (IERS) [McC 96], which regularly publishes information on change in the *Universal Time*, *Coordinated Universal Time* and polar motion.

The notion of time in physics and astronomy involves a variety of concepts which may be classified as *Dynamic Time*, *Atomic Time* and *Universal Time*.

- Dynamic time is the independent variable in equations of motion. Within the framework of Newtonian physics, the *Ephemeris Time* (ET) represents an uniform time scale that forms the basis for computation of solar system ephemeris[1]. A distinction is made between the independent time variable for geocentric description of ephemerides and for equations of motion with respect to the barycentre of the solar system. The former time scale is therefore called *Terrestrial Dynamical Time* (TDT) or simply *Terrestrial Time* (TT), and now replaces the ET.
- For scientists, the standard unit of measurement of time intervals is the SI (*Système International*) second, which is defined as a fixed value (from the frequency of the hyperfine ground-state transition of Cesium 133 atoms). The *International Atomic time* (or *Temps Atomique International*, TAI) is a practical time standard that conforms as closely as possible to the definition of the SI second. TAI corresponds closely to dynamic time and the difference between both time scales is considered constant within the measurement accuracy of atomic clocks

$$TT = TAI + 32.184s \qquad \textbf{(2.25)}$$

- Atomic and dynamic times represent a completely uniform measure of time. Civil time, in contrast, is based on the length of a solar day and thus depends both of the Earth's rotation around its axis and its revolution around the Sun. Both effects are taken into account in the definition of the *Universal Time* (UT1). This UT1 is

[1] The concept of global time is however partially abandoned in the general theory of Relativity which establishes a close relation between time and reference frame in which it is measured.

related to the *Greenwich Mean Sidereal Time* Θ (which is in fact the Greenwich hour angle of the mean vernal equinox) through a relation involving the number of Julian centuries of Universal time elapsed from the 1st January 2000. Since Sidereal time may be obtained from meridian transit observations, this relation provides a way to determine practically the UT1. Any irregularity in the Earth's rotation is then directly reflected into the UT1 through the changes in the Greenwich mean sidereal time. In order to take into account these changes in the mean solar day while maintaining the uniformity of the atomic clocks, the *Co-ordinated Universal Time* (UTC) is now used as standard measure for civil time-keeping. UTC is kept in close agreement with the UT1 scale by the introduction of leap seconds (one second step on January 1 and/or July 1). The difference between UTC and TAI is thus always an integer number of seconds, and the UTC may then be considered as an atomic time.

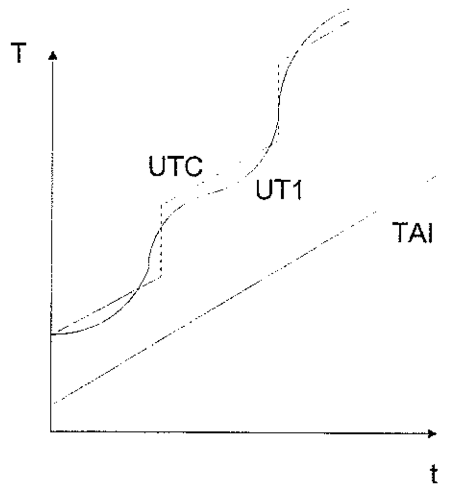


Figure 2.1

In addition to these different time scales, a practical way to have a continuous time scale which is always positive is to use the *Julian Dates* (JD). This scales indeed counts the number of days since January 1, 4712 BC at 12h including a fraction of days. Today, the Julian date number is well over two millions and it is more practical to start counting at 0h UT. The *Modified Julian Date* (MJD) is then defined as

$$MJD = JD - 2400000.5 \qquad (2.26)$$

The changes to civil calendar is realised throughout tables, or numerical algorithms.

2.2.2. Coordinate transformations

i. True of date, Earth fixed and Geometric topocentric coordinates

The rotational transformation from the Earth-Mean-Equator EME2000 frame to an Earth-Fixed frame is by convention split into *precession*, *nutation, polar motion* and *rotation around the pole-axis* [McC 96]. The rotation of the Earth is considered as an uniform rotation around its z axis, the angle $\theta(t)$ describing this rotation.

- The precession of the Earth is the secular effect of the gravitational attraction from the Sun and the planets on the equatorial bulge of the Earth. The main effect is a rotation of the mean-of-date system in the negative sense in the ecliptic plane by one turn in approximately 26000 years, which is equivalent to a rotation of $0.014°$ per year.
- The nutation is a short-periodic effect of the gravitational attraction of the Moon and to a lower degree the planets on the Earth's equatorial bulge. It has a certain periodicity with the contributions from the Moon's orbital period, and the maximum of any of the two nutation angles is $0.006°$.
- The polar motion is the motion of the Earth's pole relative to the fixed Earth. It moves in a path that looks like a spiral with a period slightly less than a year for an order of 20 meters on the Earth's surface.

The precession and nutation rotation matrices $\mathbf{P}(t)$ and $\mathbf{N}(t)$ transform the EME coordinates to the so-called *True-Of-Date coordinates* (TOD)

$$\mathbf{r}_{TOD} = \mathbf{N}(t).\mathbf{P}(t).\mathbf{r}_{EME}(t) \qquad (2.27)$$

where the precession and nutation matrices are given by

$$\begin{cases} \mathbf{P} := \mathbf{R}_z(-z).\mathbf{R}_y(\theta).\mathbf{R}_z(-\varsigma) \\ \mathbf{N} = \mathbf{R}_x(-\varepsilon-\Delta\varepsilon).\mathbf{R}_z(\Delta\psi).\mathbf{R}_x(\varepsilon) \end{cases} \qquad (2.28)$$

with $\mathbf{R}_x, \mathbf{R}_y$ and $\mathbf{R}_z$ denoting elementary rotations around the x, y and z axes, and where the parameters $z, \theta, \varsigma, \varepsilon, \Delta\varepsilon, \Delta\psi$ can be computed using the IAU1976 (International Astronomic Unity) theory of precession and the IAU1980 theory of nutation [McC 96].

The polar motion transformation $\Pi(t)$ and the Earth rotation $\Theta(t)$ transform the TOD coordinates into the *Earth-Fixed coordinates* (EF)

$$\mathbf{r}_{EF} = \Pi(t).\Theta(t).\mathbf{r}_{TOD}(t) \qquad (2.29)$$

where the two transformation matrices are given by

$$\begin{cases} \Theta = \mathbf{R}_z(\theta(t)) = \begin{pmatrix} \cos\theta(t) & \sin\theta(t) & 0 \\ -\sin\theta(t) & \cos\theta(t) & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ \Pi = \mathbf{R}_y(-x_p).\mathbf{R}_x(-y_p) = \begin{pmatrix} 1 & 0 & x_p \\ 0 & 1 & -y_p \\ -x_p & y_p & 1 \end{pmatrix} \end{cases} \qquad (2.30)$$

In these transformations, $\theta(t)$ is based on the IAU relation between sidereal time and UT1, the TT-UTC and UT1-UTC time differences as well as the values x_p, y_p can be extracted from IERS time and polar motion tables.

Ground station locations are given in this EF coordinates system through *geodetic latitude φ, longitude λ* and height h above the geoid. The EF station vector $\mathbf{R}_{EF}$ is thus given by

$$\mathbf{R}_{EF} = \left(1-e^2\sin^2\varphi\right)^{-\frac{1}{2}} R_{Earth}\begin{pmatrix}\cos\varphi.\cos\lambda\\ \cos\varphi.\sin\lambda\\ \left(1-e^2\right)\sin\varphi\end{pmatrix} + h\begin{pmatrix}\cos\varphi.\cos\lambda\\ \cos\varphi.\sin\lambda\\ \sin\varphi\end{pmatrix} \qquad (2.31)$$

where R_{Earth} is the Earth's radius and e is the Earth oblateness.

For a given ground station, the *Topocentric Geometric coordinates* (TG) of the satellite are thus obtained by rotation of the EF station-satellite vector into the local horizon frame

$$\begin{aligned}\mathbf{s}_{TG}(t) &= \Phi.\Lambda.\left[\mathbf{r}_{EF}(t)-\mathbf{R}_{EF}\right] = \Phi.\Lambda.\left[\Pi(t).\Theta(t).\mathbf{N}(t).\mathbf{P}(t).\mathbf{r}_{EME}(t)-\mathbf{R}_{EF}\right]\\ &= \Phi.\Lambda.\left[\mathbf{U}(t).\mathbf{r}_{EME}(t)-\mathbf{R}_{EF}\right]\end{aligned} \qquad (2.32)$$

where

$$\mathbf{U}(t) = \Pi(t).\Theta(t).\mathbf{N}(t).\mathbf{P}(t) \qquad (2.33)$$

and

$$\begin{cases}\Lambda = \begin{pmatrix}\cos\lambda & \sin\lambda & 0\\ -\sin\lambda & \cos\lambda & 0\\ 0 & 0 & 1\end{pmatrix}\\ \Phi = \begin{pmatrix}0 & 1 & 0\\ -\sin\varphi & 0 & \cos\varphi\\ \cos\varphi & 0 & \sin\varphi\end{pmatrix}\end{cases} \qquad (2.34)$$

ii. Astrometric topocentric coordinates

A rigorous modelling of radiometric tracking data requires to properly take into account the signal propagation time. A general measurement process should include the fact that a signal is sent to the satellite at a *ground transmit time* t_{g_t} and that the satellite transmits its signal at the *satellite transmit time* t_{s_t}, and that the ground receives it at the *ground receive time* t_{g_r}. The downlink signal path is thus

$$\mathbf{d}_{doxn} = \mathbf{r}_{EME}(t_{s\_t}) - \mathbf{U}^T(t_{g\_r}).\mathbf{R}_{EF} \qquad (2.35)$$

It links the position of the observer at the ground receive time t_{g_r} and the position of the satellite at the satellite transmit time t_{s_t}. Due to non-linear motion of the satellite and the ground station in the inertial frame, the satellite transmit time and the satellite state vector $\mathbf{x}(t_{s\_t})$ have to be computed iteratively, beginning from the ground receive time, so as to satisfy simultaneously equation **(2.35)** and the basic relation

$$\left\|\mathbf{d}_{down}\right\| = c.(t_{g\_r} - t_{s-t}) \qquad (2.36)$$

where the *light time* is $\tau = t_{g\_r} - t_{s\_t}$, and c is the speed of light in a vacuum.

Similarly, the uplink path should satisfy the two relations

$$\mathbf{d}_{up} = \mathbf{r}_{EME}(t_{s\_t}) - \mathbf{U}^T(t_{g\_r}).\mathbf{R}_{EF} \qquad (2.37)$$

$$\left\|\mathbf{d}_{up}\right\| = c.(t_{s\_t} - t_{g\_r}) \qquad (2.38)$$

The *Topocentric Astrometric coordinates* (TA) represents the direction, in a geometric inertial system, from which an observed signal reaches the station, and can be written as

$$\mathbf{s}_{TA} = \Phi.\Lambda.\left[\mathbf{U}(t_{g\_r}).\mathbf{r}_{EME}(t_{s\_t}) - \mathbf{R}_{EF}\right] \qquad (2.39)$$

The difference between astrometric and geometric coordinates is known as the *light time correction.*

iii. Apparent topocentric coordinates

Apparent Topocentric coordinates (AT) indicate the direction from which an incident signal is perceived by an observer on the rotating Earth. A rigorous description of this phenomenon is known as *stellar aberration* and requires the theory of Special Relativity. However, since the velocity of the observing stations in the geocentric inertial system is relatively small against the velocity of the electromagnetic waves, the correction

$$\begin{aligned}\mathbf{s}_{AT}(t_{g\_r}) &= \Phi.\Lambda.\left[\mathbf{r}_{EF}(t_{g\_r}) - \mathbf{R}_{EF}(t_{g\_r}) + \tau.\dot{\mathbf{R}}_{EME}(t_{g\_r})\right] \\ &= \mathbf{s}_{TA}(t_{g\_r}) + \tau.\Phi.\Lambda.\dot{\mathbf{R}}_{EME}(t_{g\_r}) = \mathbf{s}_{TA}(t_{g\_r}) + \tau.\Phi.\Lambda.\dot{\mathbf{U}}^T(t_{g\_r}).\mathbf{R}_{EF} \qquad (2.40)\end{aligned}$$

to the astrometric topocentric coordinates is sufficient.

Note :

From the above equations for astrometric and apparent topocentric coordinates, the partial derivatives of the apparent coordinates with respect to the geometric EME2000 position vector of the satellite can be approximated by

$$\frac{\partial \mathbf{s}_{AT}(t_{g\_r})}{\partial \mathbf{r}_{EME}(t_{g\_r})} \approx \Phi.\Lambda.\mathbf{U}(t_{g\_r}) \qquad (2.41)$$

2.2.3. Tracking systems

i. Range measurements

Range measurements always involve measurements of the propagation time of an electromagnetic signal path between the satellite and the ground station. They are given in terms of the uplink and downlink distances from the ground station up to the satellite and from the satellite back to the ground station

$$\begin{cases} \left\|\mathbf{d}_{down}\right\| = \left\|\mathbf{r}_{TOD}(t_{s-t}) - \mathbf{R}_{TOD}(t_{g\_r})\right\| = c.(t_{g\_r} - t_{s\_t}) \\ \left\|\mathbf{d}_{up}\right\| = \left\|\mathbf{r}_{TOD}(t_{s-t}) - \mathbf{R}_{TOD}(t_{g\_r})\right\| = c.(t_{s\_t} - t_{g\_r}) \end{cases} \quad (2.42)$$

- **2-way range**

This method uses the average uplink and downlink path from the ground station up to the satellite and back down to the station

$$\rho = \frac{1}{2}\left(\left\|\mathbf{d}_{down}\right\| + \left\|\mathbf{d}_{up}\right\|\right) \quad (2.43)$$

where partials derivatives with respect to the True of date position vector[2] are given by

$$\frac{\partial \rho_{2\_ways}}{\partial \mathbf{r}_{TOD}(t_{s\_t})} = \frac{1}{2}\left(\frac{\mathbf{d}^T{}_{up}}{\left\|\mathbf{d}^T{}_{up}\right\|} + \frac{\mathbf{d}^T{}_{down}}{\left\|\mathbf{d}^T{}_{down}\right\|}\right) \quad (2.44)$$

- **Trilateration**

The trilateration tracking system is similar to the navigational and surveying technique of triangulation. Both techniques determine the relative position between two points by using the geometry of triangles, trilateration using only distance measurements.

A simple example of trilateration is the location of a point in two dimensions relative to a coordinate system. Two points determine a two dimensional coordinate system, the line between the two reference points being defined as the X axis and its perpendicular at one of them, and in the plane of the three points, being the Y axis.

To determine the location of the third point, all three distances between the points must be known, the altitude of the third point being the Y coordinate. By applying Pythagoras's Theorem to each of the two triangles formed, two equations for Y are formed, and then give the relation for X

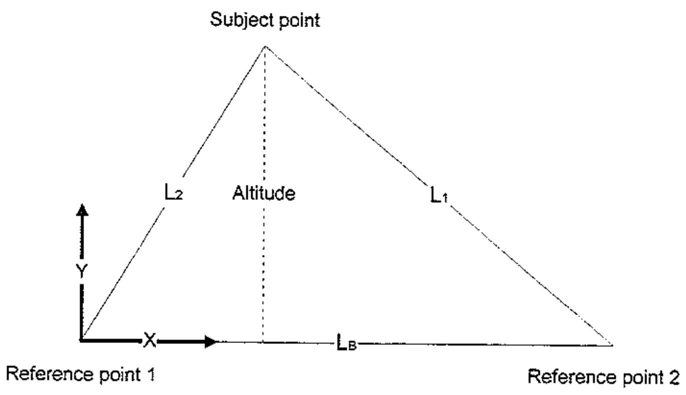
[2] These derivatives are used in the numerical implementations of both estimation methods.

$$L_2^{\,2} - X^2 = L_1^{\,2} - (L_B - X)^2 \Leftrightarrow X = \frac{(L_2^{\,2} - L_1^{\,2} + L_B^{\,2})}{2L_B} \qquad (2.45)$$

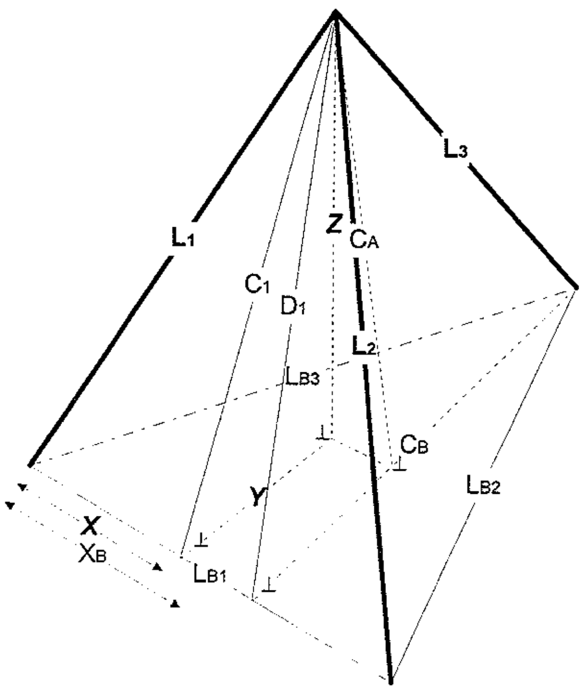
and then for Y

$$Y = (L_2^{\,2} - X^2)^{\frac{1}{2}} \qquad (2.46)$$

where L_1, L_2 and L_B are represented on the next figure



Therefore, the two coordinates of the subject point are found from the three length measurements. Extending this technique to three dimensions requires a third reference point to define the three dimensional coordinate system and the Z axis is defined by a right handed coordinate system between X and Y axes.



The four points are the summits of a tetrahedron whose sides are precisely measured. Locating the subject point very precisely can then be done in a way similar to the two-dimensional problem.

This method is implemented for satellite keeping with a network composed of ground stations measuring distances to the satellite with a range measurement method, and whose mutual distances are very well known. Moreover, such tracking systems are generally composed of more than three stations as basically required to a redundancy of data to improve orbit estimations and characterise the transponder delays of satellite.

Note :

For SES, the trilateration system is not exactly so simple, as measurements are not simultaneous. A propagation of measurements is thus required for OD, and the residuals are in terms of range instead of Cartesian coordinates.

ii. Angular measurements

Angular measurements track the satellite by controlling the receiving antenna to maximise the signal level. The antenna turn around two axes to follow the satellite movements, usually in an *azimuth-elevation* mounting[3]. A local topocentric coordinate system centred on the antenna, and pointing to North-East-Zenith, is generally introduced, where the apparent satellite position is denoted $\mathbf{s} = (s_e, s_n, s_z)$

The azimuth is the direction of the projected station-satellite line on the horizontal plane, counted clockwise from local North

$$\tan A = \frac{s_e}{s_n} \tag{2.47}$$

$$\frac{\partial A}{\partial \mathbf{r}_{EME}} = \frac{\partial A}{\partial \mathbf{s}} \cdot \frac{\partial \mathbf{s}}{\partial \mathbf{r}_{EME}} = \begin{pmatrix} \dfrac{s_n}{s_n^2 + s_e^2} & \dfrac{-s_{ne}}{s_n^2 + s_e^2} & 0 \end{pmatrix} \cdot \frac{\partial \mathbf{s}}{\partial \mathbf{r}_{EME}} \tag{2.48}$$

and the elevation is the angle between the station-satellite direction and the horizontal plane

$$\tan E = \frac{s_z}{\sqrt{s_n^2 + s_e^2}} \tag{2.49}$$

$$\frac{\partial E}{\partial \mathbf{r}_{EME}} = \frac{\partial E}{\partial \mathbf{s}} \cdot \frac{\partial \mathbf{s}}{\partial \mathbf{r}_{EME}} = \begin{pmatrix} \dfrac{-s_e s_z}{\|\mathbf{s}\|^2 \sqrt{s_n^2 + s_e^2}} & \dfrac{-s_n s_z}{\|\mathbf{s}\|^2 \sqrt{s_n^2 + s_e^2}} & \dfrac{\sqrt{s_n^2 + s_e^2}}{\|\mathbf{s}\|^2} \end{pmatrix} \cdot \frac{\partial \mathbf{s}}{\partial \mathbf{r}_{EME}} \tag{2.50}$$

The station keeping is then realised with two these angular measurements and a range measurement since the satellite position relatively to the ground station is determined with two angles and a distance.

[3]. Other kind of mounting also exists, like the XY-mounting and the equatorial mounting, but they are not used by SES.

iii. Refraction and bias

All radiometric measurements need to be corrected for the atmospheric refraction, as the effect of the refractive index of the atmosphere causes both a bending of the signal path and a change of the signal's group velocity within the atmosphere.

The effect of atmospheric refraction essentially contains two terms. The *tropospheric* term is mainly a function of the meteorological conditions (temperature, pressure and humidity) but is independent of the wavelength. The *ionospheric* refraction term, on the other hand, is proportional to the square of the signal's wavelength and is directly function of the free electron density in the atmosphere. The tropospheric term may thus be considered as a known bias perturbing measurements which is simply subtracted [Mon 96], but as the ionospheric term varies randomly it cannot be easily taken into account in the treatment of measurements data and constitute a random error source in the observation model.

Finally, all measurement types can be affected by other constant biases, conventionally subtracted from the values of the observations. These biases can be considered as parameters of the observation model to be estimated, in which case the needed measurements partial derivatives are given by

$$\frac{\partial h_i}{\partial q_j} = -\delta_{ij} \qquad (2.51)$$

where i and j are indices running over the different possible measurements types (i.e. azimuth, elevation and range), and over measurements collected at different sites (like for trilateration) or with different equipment.

2.2.4. Observation model

All measurement types can be written to the functional form (**1.2**) where the values $\mathbf{z}(t_k)$, $k = 1, \ldots, N$, are the measurements data, $\mathbf{h}\left[\mathbf{y}(t_k), t_k\right]$ is the observational model at time t_k which depends on the type of measurement (range or angle) and corresponds to the relations of paragraph 2.2.3, and $\mathbf{v}(t_k)$ represents an additive white Gaussian noise (like the ionospheric refraction effect) with zero-mean and covariance $\mathbf{R}(t_k)$

$$\begin{cases} E\left[\mathbf{v}(t_k)\right] = \mathbf{0} \\ E\left[\mathbf{v}(t_k)\mathbf{v}^T(t_k + t_l)\right] = \mathbf{R}(t_k)\delta(l) \end{cases} \qquad (2.52)$$

and is supposed not correlated with the state vector $\mathbf{x}(t)$

$$E\left[\mathbf{x}(t)\mathbf{v}(\tau)\right] = \mathbf{0} \quad \forall t, \tau \qquad (2.53)$$

Chapter 3.

Estimation and filtering methods

3.1. Introduction

3.1.1. Models

The satellite orbit is characterised the dynamic equation (**1.1**) which can be written

$$\dot{\mathbf{y}}(t) = \mathbf{f}\left[\mathbf{y}(t), t\right] + \mathbf{w}(t) \qquad (3.1)$$

where the parameters **p** and **q** and the state vector **x** are grouped in the *trajectory vector* (or *solve for*) $\mathbf{y}(t) = \begin{pmatrix} \mathbf{x}(t) \\ \mathbf{p} \\ \mathbf{q} \end{pmatrix}$ [Fra 98].

On the other hand, the set of measurements $\mathbf{z}(t_k),\ k = 1, \ldots, N$ are related to this trajectory vector through the measurement model (**1.2**) which can be written

$$\mathbf{z}(t_k) = \mathbf{h}\left[\mathbf{y}(t_k), t_k\right] + \mathbf{v}(t_k) \qquad (3.2)$$

3.1.2. Linearisation about a reference trajectory

Both observation and measurement modelling form a non-linear problem. If a reasonable reference trajectory is available and if the actual trajectory is sufficiently close to this reference throughout the time interval of interest, both above equations can be expanded in a Taylor's series about this reference trajectory $\mathbf{y}_{ref}$ to obtain linear equations

$$\dot{\mathbf{y}}(t) = \mathbf{f}\left[\mathbf{y}(t), t\right] + \mathbf{w}(t) = \mathbf{f}\left[\mathbf{y}_{ref}(t), t\right] + \mathbf{F}(t)\left[\mathbf{y}(t) - \mathbf{y}_{ref}(t)\right] + \ldots + \mathbf{w}(t)$$

$$\mathbf{z}(t_k) = \mathbf{h}\left[\mathbf{y}(t_k), t_k\right] + \mathbf{v}(t_k) = \mathbf{h}\left[\mathbf{y}_{ref}(t_k), t_k\right] + \mathbf{A}(t_k)\left[\mathbf{y}(t_k) - \mathbf{y}_{ref}(t_k)\right] + \ldots + \mathbf{v}(t_k)$$

$$\Leftrightarrow \begin{cases} \dot{\mathbf{y}}_{diff}(t) = \mathbf{F}(t)\mathbf{y}_{diff}(t) + \mathbf{w}(t) \\ \mathbf{z}_{diff}(t_k) = \mathbf{A}(t_k)\mathbf{y}_{diff}(t_k) + \mathbf{v}(t_k) \end{cases} \qquad (3.3)$$

where **F**(t) is the Jacobian of the vector function $\mathbf{f}\left[\mathbf{y}(t), t\right]$, $\mathbf{A}(t_k)$ is the partial derivatives matrix of the observational vector function $\mathbf{h}\left[\mathbf{y}(t_k), t_k\right]$ with respect to the **y** components, and where the differences between the true and reference trajectories

$\mathbf{y}_{diff}(t) \overset{def}{=} \mathbf{y}(t) - \mathbf{y}_{ref}(t)$, and $\mathbf{z}_{diff}(t) \overset{def}{=} \mathbf{z}(t) - \mathbf{z}_{ref}(t)$ have been introduced respectively with $\mathbf{y}_{ref}$ and $\mathbf{z}_{ref}$ the reference trajectory vector and the observations vector

$$\begin{cases} \mathbf{F}(t) = \left.\dfrac{\partial \mathbf{f}[\mathbf{y}(t), t]}{\partial \mathbf{y}(t)}\right|_{\mathbf{y}(t)=\mathbf{y}_{ref}(t)} \\ \mathbf{A}(t_k) = \left.\dfrac{\partial \mathbf{h}[\mathbf{y}(t_k), t_k]}{\partial \mathbf{y}(t_k)}\right|_{\mathbf{y}(t_k)=\mathbf{y}_{ref}(t_k)} \end{cases} \qquad (3.4)$$

3.1.3. Discretisation of the linearised equations

The solution of the linearised motion equations can be expressed between two consecutive observation time t_{k-1} and t_k as

$$\mathbf{y}_{diff}(t_k) = \Phi(t_k, t_{k-1})\mathbf{y}_{diff}(t_{k-1}) + \mathbf{w}'(t_k) \qquad (3.5)$$

where

$$\Phi = \frac{\partial \mathbf{y}(t)}{\partial \mathbf{y}(t_0)} \qquad (3.6)$$

with $\mathbf{y}(t_0)$ the initial state vector. The *state transition matrix* Φ for the linearised system is obtained from the matrix differential equations for $t_{k-1} \le t \le t_k$, called the *variational equation*

$$\frac{d}{dt}\Phi(t, t_{k-1}) = \mathbf{F}(t)\Phi(t, t_{k-1}) \qquad (3.7)$$

where $\Phi(t_{k-1}, t_{k-1}) = \mathbf{I}$ and $\mathbf{w}'(t) = \int_{t_{k-1}}^{t_k} \Phi(t_k, t)\mathbf{w}(t)dt$. This stochastic integral is defined in the Itô sense [Jaz 70], $\mathbf{w}'(t)$ being a discrete white noise sequence with zero mean and covariance $\mathbf{Q}'(t_k)$

$$\begin{cases} E[\mathbf{w}'(t)] = \mathbf{0} \\ E[\mathbf{w}'(t_k)\mathbf{w}'^T(t_{k+l})] = \mathbf{Q}'(t_k)\delta(l) \end{cases} \qquad (3.8)$$

and

$$\mathbf{Q}'(t_k) = \int_{t_{k-1}}^{t_k} \Phi(t_k, \tau)\mathbf{Q}(\tau)\Phi^T(t_k, \tau)d\tau \qquad (3.9)$$

The discrete noise covariance matrix $\mathbf{Q}'$ is computed from the differential equation

$$\begin{aligned} \frac{d}{dt}\mathbf{Q}'(t) &= \frac{d}{dt}\int_{t_{k-1}}^{t} \Phi(t,\tau)\mathbf{Q}(\tau)\Phi^T(t,\tau)d\tau = \Phi(t,\tau)\mathbf{Q}(\tau)\Phi^T(t,\tau) \\ &+ \int_{t_{k-1}}^{t} \mathbf{F}(t)\Phi(t,\tau)\mathbf{Q}(\tau)\Phi^T(t,\tau)d\tau + \int_{t_{k-1}}^{t} \Phi(t,\tau)\mathbf{Q}(\tau)\Phi^T(t,\tau)\mathbf{F}^T(t)d\tau \end{aligned}$$

$$\Leftrightarrow \quad \frac{d}{dt}\mathbf{Q}'(t) = \mathbf{Q}(\tau) + \mathbf{F}(t)\mathbf{Q}'(\tau) + \mathbf{Q}'(\tau)\mathbf{F}^T(t) \tag{3.10}$$

with the initial condition $\mathbf{Q}'(t_{k-1}) = \mathbf{0}$, and where $\mathbf{Q}(\tau)$ is the spectral density matrix of the original dynamic noise $\mathbf{w}(t)$. The discrete linearised system around the reference trajectory is thus represented by the set of two equations

$$\mathbf{y}_{diff}(t_k) = \Phi(t_k, t_{k-1})\mathbf{y}_{diff}(t_{k-1}) + \mathbf{w}'(t_k) \tag{3.11}$$

$$\mathbf{z}_{diff}(t_k) = \mathbf{A}_{diff}(t_k)\mathbf{y}_{diff}(t_k) + \mathbf{v}(t_k) \tag{3.12}$$

3.2. Batch Least Squares Estimation

3.2.1. Maximum likelihood estimator

The propagation model equations without process noise have here the form

$$\dot{\mathbf{y}} = \mathbf{f}[\mathbf{y}, t] \tag{3.13}$$

where $\mathbf{y}$ is an m-dimensional vector containing the state vector $\mathbf{x}$ at the initial time t_0 and the two parameter vectors $\mathbf{p}$ and $\mathbf{q}$ to be estimated. This vector thus simplifies into the only initial state vector when parameters $\mathbf{p}$ and $\mathbf{q}$ need not to be estimated.

In parallel, the measurement model equations can be represented as

$$\mathbf{Z} = \mathbf{H}[\mathbf{y}] + \mathbf{V} \tag{3.14}$$

with the hypothesis of additive zero-mean white Gaussian noise

$$\begin{cases} E[\mathbf{V}] = \mathbf{0} \\ E[\mathbf{V}\mathbf{V}^T] = \mathbf{G} \end{cases} \tag{3.15}$$

and where the three $n = M.N$ vectors

$$\begin{cases} \mathbf{Z}(t_N) = \{\mathbf{z}(t_1), \mathbf{z}(t_2), \ldots, \mathbf{z}(t_N)\}^T \\ \mathbf{H} = \{\mathbf{h}[\mathbf{y}(t_1), t_1], \mathbf{h}[\mathbf{y}(t_2), t_2], \ldots, \mathbf{h}[\mathbf{y}(t_N), t_N]\}^T \\ \mathbf{V} = \{\mathbf{v}(t_1), \mathbf{v}(t_2), \ldots, \mathbf{v}(t_N)\}^T \end{cases} \tag{3.16}$$

are respectively the set of measurements, the observational model estimated at times t_k ($k = 1, \ldots, N$), and the additive zero-mean white Gaussian observation noise.

The probability density of the data set $\mathbf{Z}$ is given [Pap 91] by

$$P(\mathbf{y}) = \frac{1}{\sqrt{(2\pi)^n \|\mathbf{G}\|}} \exp\left\{ -\frac{1}{2}[\mathbf{Z} - \mathbf{H}(\mathbf{y})]^T \mathbf{G}^{-1} [\mathbf{Z} - \mathbf{H}(\mathbf{y})] \right\} \tag{3.17}$$

So, in the absence of process noise, the value $\mathbf{y}$ corresponding to a *maximum likelihood estimator* $\hat{\mathbf{y}}$ is the one which maximise this density probability

$$\hat{\mathbf{y}} = \max_{\mathbf{y}} P(\mathbf{y}) = \max_{\mathbf{y}} \ln\left(P(\mathbf{y})\right) = \min_{\mathbf{u}}\left[-\ln\left(P(\mathbf{y})\right)\right] \qquad (3.18)$$

that is the one which minimise the square function

$$\chi^2 = \left[\mathbf{Z} - \mathbf{H}(\mathbf{y})\right]^T \mathbf{G}^{-1}\left[\mathbf{Z} - \mathbf{H}(\mathbf{y})\right] = \mathbf{r}^T\mathbf{G}^{-1}\mathbf{r} = \mathbf{r}^T\mathbf{W}\mathbf{r} \qquad (3.19)$$

where $\mathbf{W}$ is defined as the *weight matrix* and $\mathbf{r}$ is a vector containing the *residuals*

$$r_j = Z_j - H_j\left[\mathbf{y}(t_j), t_j\right] \qquad j = 1, \dots n \qquad (3.20)$$

Finding the maximum likelihood estimator thus corresponds to realise a parameter estimation which is an orbit determination as described in paragraph 1.2.

As the dependency upon the estimate $\mathbf{y}$ is non-linear, the problem is solved iteratively. At each iteration, linearisation is realised around the trajectory $\hat{\mathbf{y}}_{i-1}$ obtained from the estimation of the previous iteration and taken as the reference trajectory $\mathbf{y}_{ref}$ of paragraph 3.1.2. The residuals can then be written as

$$\mathbf{r}_i \approx \mathbf{Z} - \mathbf{H}\left[\hat{\mathbf{y}}_{i-1}\right] - \frac{\partial \mathbf{H}\left[\hat{\mathbf{y}}_{i-1}\right]}{\partial \hat{\mathbf{y}}_{i-1}} \mathbf{y}_{diff,i} \equiv \mathbf{z}_{diff,i} - \mathbf{A}_i \mathbf{y}_{diff,i} \qquad (3.21)$$

where $\mathbf{y}_{diff,i}$ is the correction to be applied at the i^{th} iteration

$$\mathbf{y}_{diff,i}(t) \overset{def}{=} \hat{\mathbf{y}}_i(t) - \hat{\mathbf{y}}_{i-1}(t) \qquad (3.22)$$

and

$$\begin{cases} \mathbf{z}_{diff,i} \overset{def}{=} \left[\mathbf{Z} - \mathbf{H}(\mathbf{y})\right]\Big|_{\mathbf{y}=\hat{\mathbf{y}}_{i-1}} \\ \mathbf{A}_i \overset{def}{=} \dfrac{\partial \mathbf{H}(\mathbf{y})}{\partial \mathbf{y}}\Bigg|_{\mathbf{y}=\hat{\mathbf{y}}_{i-1}} \end{cases} \qquad (3.23)$$

The criterion of minimising the square function $\chi^2 = \left\|\mathbf{r}_i\right\|^2$ then becomes for the unweighted problem ($\mathbf{W} = \mathbf{I}$)

$$\left\|\mathbf{A}_i \mathbf{y}_{diff,i} - \mathbf{z}_{diff,i}\right\|^2 = \text{minimum} \qquad (3.24)$$

The problem is thus reduced, at each iteration, to a linear Least Squares (LS) problem, iterations being stopped when the convergence condition is reached

$$\frac{\chi^2_{i+1} - \chi^2_i}{\chi^2_{i+1}} < \varepsilon_c \qquad (3.25)$$

3.2.2. Normal equations

At each iteration, the linear LS problem presented in paragraph 3.2.1. can be solved practically through the *normal equations* for difference between the reference and true trajectories

$$\begin{aligned}
&\left\|\mathbf{A}_i\mathbf{y}_{diff,i} - \mathbf{z}_{diff,i}\right\|^2 = \text{minimum} \\
&\Rightarrow \frac{\partial}{\partial \mathbf{y}_{diff,i}}\left[\left(\mathbf{A}_i\mathbf{y}_{diff,i} - \mathbf{z}_{diff,i}\right)^T\left(\mathbf{A}_i\mathbf{y}_{diff,i} - \mathbf{z}_{diff,i}\right)\right] = 0 \\
&\Rightarrow 2\mathbf{A}_i^T\left(\mathbf{A}_i\mathbf{y}_{diff,i} - \mathbf{z}_{diff,i}\right) = 0 \\
&\Rightarrow \mathbf{A}^T\mathbf{A}\mathbf{y}_{diff} = \mathbf{A}^T\mathbf{z}_{diff} \qquad (3.26)
\end{aligned}$$

where the iteration index i is omitted from now on.

The weighted problem resolution can be easily deduced from the unweighted one by means of the transformation

$$\chi^2 = \left\|\mathbf{r}_i\right\|^2 \quad \rightarrow \quad \chi^2 = \left\|\mathbf{S}.\mathbf{r}_i\right\|^2 \qquad (3.27)$$

where

$$\mathbf{S}^T.\mathbf{S} = \mathbf{G}^{-1} = \mathbf{W} \qquad (3.28)$$

One then gets

$$\mathbf{r}_i \approx \mathbf{z}_{diff,i} - \mathbf{A}_i\mathbf{y}_{diff,i} \quad \rightarrow \quad \mathbf{S}.\mathbf{r}_i \approx \mathbf{S}.\mathbf{z}_{diff,i} - \mathbf{S}.\mathbf{A}_i\mathbf{y}_{diff,i} \equiv \mathbf{z}'_{diff,i} - \mathbf{A}'_i\,\mathbf{y}_{diff,i} \qquad (3.29)$$

with the transformations

$$\mathbf{b}' = \mathbf{S}.\mathbf{b} \quad , \quad \mathbf{A}' = \mathbf{S}.\mathbf{A} \qquad (3.30)$$

The weighted normal equations are then

$$\begin{aligned}
&(\mathbf{A}'^T\,\mathbf{A}')\mathbf{y}_{diff} = \mathbf{A}'^T\,\mathbf{z}'_{diff} \\
&\Rightarrow (\mathbf{A}^T.\mathbf{S}^T.\mathbf{S}.\mathbf{A})\mathbf{y}_{diff} = \mathbf{A}^T.\mathbf{S}^T.\mathbf{S}.\mathbf{z}_{diff} \\
&\Rightarrow (\mathbf{A}^T.\mathbf{W}.\mathbf{A})\mathbf{y}_{diff} = \mathbf{A}^T.\mathbf{W}.\mathbf{z}_{diff} \qquad (3.31)
\end{aligned}$$

It can thus be reduced to the unweighted problem through the transformation matrix S, so it is sufficient to study the unweighted case.

Note :

When all measurements errors are not correlated and characterised by standard deviations σ_k, $k = 1,\ldots n$, the matrix S is diagonal

$$\mathbf{S} = diag\left(\sigma_1^{-1},\ldots,\sigma_n^{-1}\right) \qquad (3.32)$$

3.2.3. Least Squares estimate and Covariance matrix

From equation (3.26) one gets the LS estimate for the difference between the reference and true trajectory

$$\hat{\mathbf{y}}_{diff} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{z}_{diff} \qquad (3.33)$$

where

$$\mathbf{z}_{diff} = E(\mathbf{z}_{diff}) + \mathbf{V} \qquad (3.34)$$

is a multivariate random variable with properties $\begin{cases} E[\mathbf{V}] = \mathbf{0} \\ E\left[\mathbf{V}\mathbf{V}^T\right] = \mathbf{I} \end{cases}$, for the unweighted problem. From this estimate, the covariance matrix can be written, with the transformations

$$\left(\hat{\mathbf{y}}_{diff} - E\left(\hat{\mathbf{y}}_{diff}\right)\right) = \left(\mathbf{A}^T.\mathbf{A}\right)^{-1}\mathbf{A}^T.\mathbf{V} \quad \Leftrightarrow \quad \left(\mathbf{A}^T.\mathbf{A}\right).\left(\hat{\mathbf{y}}_{diff} - E\left(\hat{\mathbf{y}}_{diff}\right)\right) = \mathbf{A}^T.\mathbf{V}$$
$$\Rightarrow \left(\mathbf{A}^T.\mathbf{A}\right)^{-1} E\left[\left(\hat{\mathbf{y}}_{diff} - E\left(\hat{\mathbf{y}}_{diff}\right)\right)\left(\hat{\mathbf{y}}_{diff} - E\left(\hat{\mathbf{y}}_{diff}\right)\right)^T\right]\left(\mathbf{A}^T.\mathbf{A}\right) = \mathbf{A}^T.E\left(\mathbf{V}.\mathbf{V}^T\right).\mathbf{A}$$

as the inverse of the information matrix $\mathbf{A}^T.\mathbf{A}$

$$\mathbf{P} = E\left[\left(\hat{\mathbf{y}}_{diff} - E\left(\hat{\mathbf{y}}_{diff}\right)\right)\left(\hat{\mathbf{y}}_{diff} - E\left(\hat{\mathbf{y}}_{diff}\right)\right)^T\right] = \left(\mathbf{A}^T.\mathbf{A}\right)^{-1} \qquad (3.35)$$

3.2.4. Practical considerations

i. Computation of partial derivatives

To compute the partial derivatives of matrix $\mathbf{A}_i$, the vector **H** has to be decomposed into its components $\mathbf{h}\left[t_k\right]$. It is then possible to evaluate successively (where **y** represents $\mathbf{y}_{diff}$)

- $\partial\mathbf{h}/\partial\mathbf{y}$ which is obtained through the equations

$$\frac{\partial\mathbf{h}}{\partial\mathbf{y}(t_0)} = \frac{\partial\mathbf{h}}{\partial\mathbf{y}(t)}.\frac{\partial\mathbf{y}(t)}{\partial\mathbf{y}(t_0)} = \frac{\partial\mathbf{h}}{\partial\mathbf{y}(t)}.\Phi(t,t_0) \qquad (3.36)$$

where $\Phi = \dfrac{\partial\mathbf{y}(t)}{\partial\mathbf{y}(t_0)}$ is the state transition matrix computed throughout the *variational equation* introduced in paragraph 3.1.3.

$$\frac{d}{dt}\Phi = \frac{d}{dt}\frac{\partial\mathbf{y}}{\partial\mathbf{y}(t_0)} = \frac{\partial\dot{\mathbf{y}}}{\partial\mathbf{y}(t_0)} = \frac{\partial\mathbf{f}}{\partial\mathbf{y}(t_0)} = \frac{\partial\mathbf{f}}{\partial\mathbf{y}(t)}\frac{\partial\mathbf{y}(t)}{\partial\mathbf{y}(t_0)} = \frac{\partial\mathbf{f}}{\partial\mathbf{y}(t_0)}\Phi = \mathbf{F}(t).\Phi \qquad (3.37)$$

which are integrated along with the motion equations, starting with $\Phi(t_0,t_0) = \mathbf{I}$,

- $\partial \mathbf{f}/\partial \mathbf{y}(t)$, which is obtained by differentiating the force model components of the propagation model with respect to the instantaneous trajectory vector.

The vector $\mathbf{y}(t)$ may also be decomposed into its components $\mathbf{q}$, $\mathbf{p}$ and $\mathbf{x}_0 = \mathbf{x}_{ref}(t_0)$. The partial derivatives are then computed through variational equations of corresponding size.

The matrices Φ and $\mathbf{F}$ may be decomposed as

$$\Phi = \begin{pmatrix} \frac{\partial \mathbf{x}}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{x}}{\partial \mathbf{p}} & \frac{\partial \mathbf{x}}{\partial \mathbf{q}} \\ \frac{\partial \mathbf{p}}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{p}}{\partial \mathbf{p}} & \frac{\partial \mathbf{p}}{\partial \mathbf{q}} \\ \frac{\partial \mathbf{q}}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{q}}{\partial \mathbf{p}} & \frac{\partial \mathbf{q}}{\partial \mathbf{q}} \end{pmatrix} = \begin{pmatrix} \Phi' & \frac{\partial \mathbf{x}}{\partial \mathbf{p}} & \frac{\partial \mathbf{x}}{\partial \mathbf{q}} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{pmatrix} \tag{3.38}$$

and

$$\mathbf{F} = \begin{pmatrix} \frac{\partial \mathbf{f}}{\partial \mathbf{x}_0} \\ \frac{\partial \mathbf{f}}{\partial \mathbf{p}} \\ \frac{\partial \mathbf{f}}{\partial \mathbf{q}} \end{pmatrix} \tag{3.39}$$

The partial derivative then becomes

- $\partial \mathbf{h}/\partial \mathbf{q}$ which is obtained directly from the expression of the measurement model.

- $\partial \mathbf{h}/\partial \mathbf{x}_0$ which is obtained through variational equation

$$\frac{\partial \mathbf{h}}{\partial \mathbf{x}_0} = \frac{\partial \mathbf{h}}{\partial \mathbf{x}(t)} \cdot \frac{\partial \mathbf{x}(t)}{\partial \mathbf{x}_0} = \frac{\partial \mathbf{h}}{\partial \mathbf{x}(t)} . \Phi'(t,t_0) \tag{3.40}$$

where $\Phi' = \frac{\partial \mathbf{x}(t)}{\partial \mathbf{x}_0}$ is the *sub-matrix of state transition* computed throughout the appropriate variational equation

$$\frac{d}{dt}\Phi' = \frac{d}{dt}\frac{\partial \mathbf{x}(t)}{\partial \mathbf{x}_0} = \frac{\partial \dot{\mathbf{x}}(t)}{\partial \mathbf{x}_0} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}_0} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}(t)} \frac{\partial \mathbf{x}(t)}{\partial \mathbf{x}_0} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}(t)} \Phi' = \mathbf{F}(t).\Phi' \tag{3.41}$$

integrated along with the motion equations, starting with $\Phi(t_0,t_0) = \mathbf{I}$.

- $\partial \mathbf{h}/\partial \mathbf{p}$ which is obtained in a similar way through equations

$$\frac{\partial \mathbf{h}}{\partial \mathbf{p}} = \frac{\partial \mathbf{h}}{\partial \mathbf{x}(t)} . \frac{\partial \mathbf{x}(t)}{\partial \mathbf{p}} = \frac{\partial \mathbf{h}}{\partial \mathbf{x}(t)} . \mathrm{S}(t) \qquad (3.42)$$

where $\mathbf{S} = \frac{\partial \mathbf{x}(t)}{\partial \mathbf{p}}$ is the *sensitivity matrix* computed throughout the supplementary variational equation

$$\frac{d}{dt}\frac{\partial \mathbf{x}}{\partial \mathbf{p}} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}(t)} \frac{\partial \mathbf{x}(t)}{\partial \mathbf{p}} + \frac{\partial \mathbf{f}}{\partial \mathbf{p}} \qquad (3.43)$$

which again has to be integrated numerically along with the motion equation.

In practice, the three quantities $\mathbf{x}(t)$, $\Phi'(t,t_0)$ and $\mathrm{S}(t)$ are packed in a single vector, and appropriate procedures are used to perform packing and unpacking operations as required [Mon 96].

ii. QR Factorisation

The solution of the normal equations may be very sensitive to small errors in the information matrix $\mathbf{A}^T.\mathbf{A}$ that are inevitable, and renders it singular if these errors are smaller than the machine accuracy. Reducing this influence of individual measurement error with a large number of data may be insufficient if the measurement geometry and distribution does not provide enough information on all estimated parameters.

It may be helpful to use a different treatment of the LS problem that avoid the normal equations. It is based on a QR factorisation of the $n \times m$ matrix $\mathbf{A}$ into an orthogonal $n \times n$ matrix $\mathbf{Q}$ and an upper triangular $m \times m$ matrix $\mathbf{R}$, where $n = N.M$ is the total number of observations and m is the number of parameters to be estimated

$$\mathbf{A}_{n\times m} = \mathbf{Q}_{n\times n}\begin{pmatrix} \mathbf{R}_{m\times m} \\ \mathbf{0}_{(n-m)\times m} \end{pmatrix} \qquad (3.44)$$

Since $\mathbf{Q}^T.\mathbf{Q} = \mathbf{Q}.\mathbf{Q}^T = \mathbf{I}$, the function χ^2 may be written as

$$\begin{aligned} \chi^2 &= \left(\mathbf{A}\mathbf{y}_{diff} - \mathbf{z}_{diff}\right)^T\left(\mathbf{A}\mathbf{y}_{diff} - \mathbf{z}_{diff}\right) \\ \Leftrightarrow \chi^2 &= \left(\begin{pmatrix}\mathbf{R}\\ \mathbf{0}\end{pmatrix}\mathbf{y}_{diff} - \begin{pmatrix}\mathbf{d}\\ \mathbf{r}\end{pmatrix}\right)^T .\mathbf{Q}^T\mathbf{Q}.\left(\begin{pmatrix}\mathbf{R}\\ \mathbf{0}\end{pmatrix}\mathbf{y}_{diff} - \begin{pmatrix}\mathbf{d}\\ \mathbf{r}\end{pmatrix}\right) \\ \Leftrightarrow \chi^2 &= \left(\mathbf{R}\mathbf{y}_{diff} - \mathbf{d}\right)^T\left(\mathbf{R}\mathbf{y}_{diff} - \mathbf{d}\right) + \mathbf{r}^T\mathbf{r} \qquad (3.45) \end{aligned}$$

where $\mathbf{Q}^T\mathbf{z}_{diff}$ has been separated into vectors $\mathbf{d}$ and $\mathbf{r}$ of dimension m and $n-m$.

From this expression, one can see that the minimum is reached for

$$\mathbf{R}\mathbf{y}_{diff} = \mathbf{d} \qquad (3.46)$$

Normal equations are thus not required anymore when using the orthogonal transformation, and numerical problems arising from computation of $\mathbf{A}^T.\mathbf{A}$ are avoided.

Note : In case of weighted observations, this method may also be applied if $\mathbf{A}$ and $\mathbf{z}_{diff}$ are replaced by $\mathbf{S.A}$ and $\mathbf{S}.\mathbf{z}_{diff}$ respectively, where $\mathbf{S}$ is the square root of the weighting matrix $\mathbf{W} = \mathbf{S}^T\mathbf{S}$.

iii. Sequential accumulation

The QR factorisation can be performed using a type of orthogonal transformations known as Givens rotations [Giv 58]. Each individual rotation is used to eliminate a single element of the lower triangular part of the matrix $\mathbf{R}$, the global orthogonal matrix being formed by accumulation of individual rotation matrices. This method allows the transformation to be realised in a row-wise manner, where matrix $\mathbf{A}$ and vector $\mathbf{z}_{diff}$ need not to be given as a whole to compute the QR factorisation.

The measurement equations

$$a_k^{\ T}\mathbf{y}_{diff} = z_{k,diff} \qquad k = 1,\dots n \qquad (3.47)$$

where $a_k^{\ T}$ and $z_{k,diff}$ are respectively the k^{th} rows of $\mathbf{A}$ and $\mathbf{z}_{diff}$ can thus be processed one by one using the data sequential accumulation by Givens rotations algorithm illustrated in Figure 3.1, where underlined elements indicate at each step which values are affected by the transformation that annihilate the leading non-zero element of the data equation.

$$\begin{pmatrix} \underline{R_{1,1}} & \underline{R_{1,2}} & \underline{R_{1,3}} \\ 0 & R_{2,2} & R_{2,3} \\ 0 & 0 & R_{3,3} \\ \underline{a_1} & \underline{a_2} & \underline{a_3} \end{pmatrix} \mathbf{y}_{diff} = \begin{pmatrix} \underline{d_1} \\ d_2 \\ d_3 \\ \underline{z'_{diff}} \end{pmatrix} \rightarrow \begin{pmatrix} R_{1,1} & R_{1,2} & R_{1,3} \\ 0 & \underline{R_{2,2}} & \underline{R_{2,3}} \\ 0 & 0 & R_{3,3} \\ 0 & \underline{a_2} & \underline{a_3} \end{pmatrix} \mathbf{y}_{diff} = \begin{pmatrix} d_1 \\ \underline{d_2} \\ d_3 \\ \underline{z'_{diff}} \end{pmatrix}$$

$$\rightarrow \begin{pmatrix} R_{1,1} & R_{1,2} & R_{1,3} \\ 0 & R_{2,2} & R_{2,3} \\ 0 & 0 & \underline{R_{3,3}} \\ 0 & 0 & \underline{a_3} \end{pmatrix} \mathbf{y}_{diff} = \begin{pmatrix} d_1 \\ d_2 \\ \underline{d_3} \\ \underline{z'_{diff}} \end{pmatrix} \rightarrow \begin{pmatrix} R_{1,1} & R_{1,2} & R_{1,3} \\ 0 & R_{2,2} & R_{2,3} \\ 0 & 0 & R_{3,3} \\ 0 & 0 & 0 \end{pmatrix} \mathbf{y}_{diff} = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ \underline{z'_{diff}} \end{pmatrix}$$

Figure 3.1

For each measurement, the given upper triangular system $\mathbf{R}\mathbf{y}_{diff} = \mathbf{d}$ and the single data equation $a_k^{\ T}\mathbf{y}_{diff} = z_{k,diff}$ are transformed into an upper triangular system $\mathbf{R}'\mathbf{y}_{diff} = \mathbf{d}'$ and a scalar z'_{diff} through a sequence of m Givens rotations. The precedent square sum of residuals $\|\mathbf{r}\|^2$, known before processing the new measurement equation, is updated after performing the triangularisation by

$$\|\mathbf{r}'\|^2 = \|\mathbf{r}\|^2 + \left(z'_{diff}\right)^2 \tag{3.48}$$

The process starts from $\mathbf{R} = \mathbf{0}$ and $\mathbf{z}_{diff} = \mathbf{0}$ and may then be applied to include all data equations in a recursive way. The only quantities to be stored are the upper triangular matrix **R**, the vector **d**, and the Euclidean norm $\|\mathbf{r}\|^2$.

iv. Solution of the accumulated equations

After having accumulated all data equations, to determine the solution of the LS problem, the following linear system have to be solved for $\mathbf{y}_{diff}$

$$\begin{pmatrix} R_{1,1} & R_{1,2} & \cdots & R_{1,m-1} & R_{1,m} \\ 0 & R_{2,2} & \cdots & R_{2,m-1} & R_{2,m} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & R_{m-1,m-1} & R_{m-1,m} \\ 0 & 0 & 0 & 0 & R_{m,m} \end{pmatrix} \cdot \begin{pmatrix} y_{1,diff} \\ y_{2,diff} \\ \cdots \\ y_{m-1,diff} \\ y_{m,diff} \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ \cdots \\ d_{m-1} \\ d_m \end{pmatrix} \tag{3.49}$$

Due to its triangular structure, its solution is obtained by the back-substitution

$$\begin{cases} \hat{y}_{n,diff} = \dfrac{d_n}{R_{n,n}} \\ \hat{y}_{k,diff} = \dfrac{d_k - \sum_{j=k+1}^{n} R_{k,j} y_{j,diff}}{R_{k,k}} & \mathrm{k = n-1,\ldots,1} \end{cases} \tag{3.50}$$

where diagonal elements of **R** are non-zero provided its rank is m, the same as **A**, that is if the LS problem is non-singular.

Making use of the QR decomposition of **A**, the covariance matrix of the estimated parameters may be computed [Law 74] from the inverse of the upper triangular matrix **R** as

$$Cov\left(\hat{\mathbf{y}}_{diff}, \hat{\mathbf{y}}_{diff}\right) = \left(\mathbf{A}^T\mathbf{A}\right)^{-1} = \left(\mathbf{R}^T\mathbf{R}\right)^{-1} = \mathbf{R}^{-1}\mathbf{R}^{-T} \tag{3.51}$$

v. Sequential accumulation with a priori information

Assuming now that an a priori information on the estimated parameters is available in the form of an a priori estimate $\mathbf{y}_{diff}{}^a$ and an associated covariance matrix $Cov\left(\mathbf{y}_{diff}{}^a, \mathbf{y}_{diff}{}^a\right) = \mathbf{P}_0(t_0)$, the sequential accumulation may also be applied. To this end, the upper triangular matrix **R** has to be initialised with a square root factor $\mathbf{R}^a$ of the inverse a priori covariance matrix, this a priori covariance matrix is thus supposed to be not null.

The symmetric a priori covariance matrix being positive definite, it may be factorised into the product of an upper triangular matrix and its transposed, similarly to the Cholesky decomposition [Bie 77]

$$Cov\left(\mathbf{y}_{diff}{}^{a}, \mathbf{y}_{diff}{}^{a}\right) = \left(\mathbf{T}^{a}\right)\left(\mathbf{T}^{a}\right)^{T} \qquad (3.52)$$

Given the factor $\mathbf{T}^{a}$ of the a priori covariance matrix, the upper triangular square root factor $\mathbf{R}^{a}$ of the a priori information matrix

$$Cov\left(\hat{\mathbf{y}}_{diff}, \hat{\mathbf{y}}_{diff}\right)^{-1} = \left(\mathbf{T}^{a}\right)^{-T}\left(\mathbf{T}^{a}\right) = \left(\mathbf{R}^{a}\right)^{-T}\left(\mathbf{R}^{a}\right) \qquad (3.53)$$

is thus obtained by a simple inversion of $\mathbf{T}^{a}$

$$\mathbf{R}^{a} = \left(\mathbf{T}^{a}\right)^{-1} \qquad (3.54)$$

The extended loss function may then be written as

$$\chi^{2} = \left(\mathbf{A}\mathbf{y}_{diff} - \mathbf{z}_{diff}\right)^{T}\left(\mathbf{A}\mathbf{y}_{diff} - \mathbf{z}_{diff}\right) + \left(\mathbf{y}_{diff} - \mathbf{y}_{diff}{}^{a}\right)^{T}\left(Cov\left(\mathbf{y}_{diff}{}^{a}, \mathbf{y}_{diff}{}^{a}\right)\right)^{-1}\left(\mathbf{y}_{diff} - \mathbf{y}_{diff}{}^{a}\right)$$

$$\Leftrightarrow \chi^{2} = \left(\mathbf{A}\mathbf{y}_{diff} - \mathbf{z}_{diff}\right)^{T}\left(\mathbf{A}\mathbf{y}_{diff} - \mathbf{z}_{diff}\right) + \left(\mathbf{R}^{a}\mathbf{y}_{diff} - \mathbf{R}^{a}\mathbf{y}_{diff}{}^{a}\right)^{T}\left(\mathbf{R}^{a}\mathbf{y}_{diff} - \mathbf{R}^{a}\mathbf{y}_{diff}{}^{a}\right)$$

$$\Leftrightarrow \chi^{2} = \left(\begin{pmatrix}\mathbf{R}^{a}\\ \mathbf{A}\end{pmatrix}\mathbf{y}_{diff} - \begin{pmatrix}\mathbf{R}^{a}\mathbf{y}_{diff}{}^{a}\\ \mathbf{z}_{diff}\end{pmatrix}\right)^{T} \cdot \left(\begin{pmatrix}\mathbf{R}^{a}\\ \mathbf{A}\end{pmatrix}\mathbf{y}_{diff} - \begin{pmatrix}\mathbf{R}^{a}\mathbf{y}_{diff}{}^{a}\\ \mathbf{z}_{diff}\end{pmatrix}\right) \qquad (3.55)$$

The a priori information is therefore handle in the same way as the additional observations and is incorporated into the sequential accumulation algorithm by initialising the information matrix factor $\mathbf{R}$ with the a priori value $\mathbf{R}^{a}$ and the vector $\mathbf{d}$ with the a priori value $\mathbf{d}^{a} = \mathbf{R}^{a}\mathbf{y}_{diff}{}^{a}$.

In consequence, the LS with a priori information can be deduced from the standard formulation through the substitutions

$$\mathbf{A} \rightarrow \begin{pmatrix}\mathbf{R}^{a}\\ \mathbf{A}\end{pmatrix} \qquad \mathbf{z}_{diff} \rightarrow \begin{pmatrix}\mathbf{R}^{a}\mathbf{y}_{diff}{}^{a}\\ \mathbf{z}_{diff}\end{pmatrix} \qquad (3.56)$$

3.2.5. Recursive Least Squares estimation

i. Equations of Least Squares orbit determination

Using equations of paragraph 3.2.4, the initial trajectory estimation after having processed *N* measurements becomes

$$\hat{\mathbf{y}}_{diff} = \left\{\left(\left(\mathbf{R}^{a}\right)^{T} \quad \mathbf{A}^{T}\right).\begin{pmatrix}\mathbf{R}^{a}\\ \mathbf{A}\end{pmatrix}\right\}^{-1}\left\{\left(\left(\mathbf{R}^{a}\right)^{T} \quad \mathbf{A}^{T}\right).\begin{pmatrix}\mathbf{R}^{a}\mathbf{y}_{diff}{}^{a}\\ \mathbf{z}_{diff}\end{pmatrix}\right\}$$

$$\Leftrightarrow \hat{\mathbf{y}}_{diff} = \left[\mathbf{P}_{0}{}^{-1} + \mathbf{A}^{T}\mathbf{A}\right]^{-1}\left[\mathbf{P}_{0}{}^{-1}\mathbf{y}_{diff}{}^{a} + \mathbf{A}^{T}\mathbf{z}_{diff}\right] \qquad (3.57)$$

where $\mathbf{P}_0 = (\mathbf{R}^a)^T \mathbf{R}^a$ is the a priori covariance matrix at time t_0, and the different matrices implicitly refer to N measurements processed.

For the weighted case, the substitutions $\mathbf{A} \to \mathbf{S}.\mathbf{A}$ and $\mathbf{z}_{diff} \to \mathbf{S}.\mathbf{z}_{diff}$ yield

$$\hat{\mathbf{y}}_{diff} = \left[\mathbf{P}_0^{-1} + \mathbf{A}^T.\mathbf{W}.\mathbf{A}\right]^{-1}\left[\mathbf{P}_0^{-1}\mathbf{y}_{diff}{}^a + \mathbf{A}^T.\mathbf{W}.\mathbf{z}_{diff}\right] \qquad (3.58)$$

where $\mathbf{W} = \mathbf{G}^{-1}$.

And the covariance matrix becomes

$$\mathbf{P} = \left[\mathbf{P}_0^{-1} + \mathbf{A}^T.\mathbf{W}.\mathbf{A}\right]^{-1} \qquad (3.59)$$

where the right-hand side can be written as

$$\left[\mathbf{P}_0^{-1} + \mathbf{A}^T.\mathbf{W}.\mathbf{A}\right]^{-1} = \mathbf{P}_0 - \mathbf{P}_0\mathbf{A}^T\left[\mathbf{A}.\mathbf{P}_0.\mathbf{A}^T + \mathbf{G}\right]^{-1}\mathbf{A}.\mathbf{P}_0 \qquad (3.60)$$

where the Woodbury's matrix inversion lemma

$$(\mathbf{A} + \mathbf{BCD})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}\left(\mathbf{D}\mathbf{A}^{-1}\mathbf{B} + \mathbf{C}^{-1}\right)\mathbf{D}\mathbf{A}^{-1}$$

has been used.

Defining the matrix $\mathbf{K}$ as

$$\mathbf{K} \overset{def}{=} \mathbf{P}_0\mathbf{A}^T\left[\mathbf{A}.\mathbf{P}_0.\mathbf{A}^T + \mathbf{G}\right]^{-1} \qquad (3.61)$$

the estimate vector and the covariance matrix can thus be rewritten, after having processed N measurements, as

$$\begin{cases} \hat{\mathbf{y}}_{diff} = \mathbf{y}_{diff}{}^a + \mathbf{K}\left[\mathbf{z}_{diff} - \mathbf{A}\mathbf{y}_{diff}{}^a\right] \\ \mathbf{P} = [\mathbf{I} - \mathbf{KA}]\mathbf{P}_0 \end{cases} \qquad (3.62)$$

This formulation allows the LS to be implemented in a recursive way. Whenever the k^{th} new measurement is collected at time t_k, the a priori information is taken from the previous estimation at time t_{k-1} and the process is repeated.

Then, for a single measurement and a transition from t_{k-1} to t_k, the matrix $\mathbf{A}$ and the vector $\mathbf{z}_{diff}$ become $\mathbf{A}(t_k)$ and $\mathbf{z}_{diff}(t_k)$, and the $N.M \times N.M$ matrix $\mathbf{G}$ becomes the $M \times M$ matrix $\mathbf{G}(t_k) = \mathbf{W}^{-1}(t_k) = E\left[\mathbf{V}(t_k)\mathbf{V}(t_k)^T\right]$. The a priori estimate $\mathbf{y}_{diff}{}^a$ and covariance matrix $\mathbf{P}_0$ are then replaced by $\hat{\mathbf{y}}_{diff}\left(t_0 \middle| t_{k-1}\right)$ and $\mathbf{P}\left(t_0 \middle| t_{k-1}\right)$, while the new estimates $\hat{\mathbf{y}}_{diff}$ and $\mathbf{P}$ are denoted $\hat{\mathbf{y}}_{diff}\left(t_0 \middle| t_k\right)$ and $\mathbf{P}\left(t_0 \middle| t_k\right)$.

The *equations of recursive LS estimation* then become

$$\begin{cases} \hat{\mathbf{y}}_{diff}\left(t_0 \middle| t_k\right) = \hat{\mathbf{y}}_{diff}\left(t_0 \middle| t_{k-1}\right) + \mathbf{K}(t_k)\left[\mathbf{z}_{diff}(t_k) - \mathbf{A}(t_k)\hat{\mathbf{y}}_{diff}\left(t_0 \middle| t_{k-1}\right)\right] \\ \mathbf{P}\left(t_0 \middle| t_k\right) = \left[\mathbf{I} - \mathbf{K}(t_k)\mathbf{A}(t_k)\right]\mathbf{P}\left(t_0 \middle| t_{k-1}\right) \end{cases} \tag{3.63}$$

ii. Equations of orbit propagation

If for some time t_0 these estimations are known after having processed the N measurement data in the orbit determination (OD), the equations of motion can be integrated to determine $\hat{\mathbf{y}}_{diff}\left(t_l \middle| t_N\right)$ and $\mathbf{P}\left(t_l \middle| t_N\right)$ at any time $t_l > t_N$, that is realise an orbit propagation (OP) as described in paragraph 1.2.

This operation is practically realised for the covariance matrix with the discrete solution of the linearised motion equations of paragraph 3.1.3 without the process noise term, where Φ is obtained by integrating (**3.7**)

$$\mathbf{P}\left(t_l \middle| t_N\right) = \Phi(t_l, t_0).\mathbf{P}\left(t_0 \middle| t_N\right).\Phi^T(t_l, t_0) \tag{3.64}$$

iii. Iterated recursive Least Squares

In general the a priori estimate is not very accurate, and the reference trajectory based on this initial guess may not be close to the true trajectory over the whole interval of measurements. If the batch of N observation is sufficiently large, the final estimation of the first OD, $\hat{\mathbf{y}}_{diff,1}\left(t_0 \middle| t_N\right)$, is closer to the true initial vector than the initial guess.

Taking this estimate as the new reference initial condition, the whole batch of observations is reprocess, that is linearisation is realised around this new reference and the algorithm is rerun for the linearised system over the same batch of observations.

This process corresponds to the iterations described in paragraph 3.2.1, and is thus repeated until convergence has been reached. This algorithm is a *batch estimator* because an estimation can only be obtained after processing the whole batch of observations.

In practice, the non-linear observation function $\mathbf{h}$ is used to compute the residual vector $\mathbf{r}_i \equiv \mathbf{z}_{diff,i} - \mathbf{A}_i\mathbf{y}_{diff,i}$, and realise the orbit determination

$$\hat{\mathbf{y}}_{diff,i}\left(t_0 \middle| t_k\right) = \hat{\mathbf{y}}_{diff,i}\left(t_0 \middle| t_{k-1}\right) + \mathbf{K}(t_k)\left[\mathbf{z}_{diff}(t_k) - \mathbf{h}\left[\hat{\mathbf{y}}_{diff,i-1}\left(t_k \middle| t_N\right), t_k\right]\right] \tag{3.65}$$

while the orbit propagation to time t_l is realised with propagation function $\mathbf{f}$

$$\hat{\mathbf{y}}_{i-1}\left(t_l \middle| t_N\right) = \mathbf{y}_{ref,i}\left(t_l\right) = \hat{\mathbf{y}}_{i-1}\left(t_0 \middle| t_N\right) + \int_{t_0}^{t_l} \mathbf{f}\left[\hat{\mathbf{y}}_{i-1}\left(t \middle| t_N\right), t\right] dt \tag{3.66}$$

3.3. Kalman Filter

Using the equation (**3.61**) and (**3.63**), one can define the *Kalman gain* as

$$\mathbf{K}(t) \overset{def}{=} \mathbf{P}\left(t\middle|t_{k-1}\right)\mathbf{A}(t_k)^T\left[\mathbf{A}(t_k).\mathbf{P}\left(t\middle|t_{k-1}\right).\mathbf{A}(t_k)^T + \mathbf{G}(t_k)\right]^{-1} \qquad (3.67)$$

and the *filtering equations* as

$$\begin{cases} \hat{\mathbf{y}}_{diff}\left(t\middle|t_k\right) = \hat{\mathbf{y}}_{diff}\left(t\middle|t_{k-1}\right) + \mathbf{K}(t_N)\left[\mathbf{z}_{diff}(t_N) - \mathbf{A}(t_N)\hat{\mathbf{y}}_{diff}\left(t\middle|t_{k-1}\right)\right] \\ \mathbf{P}\left(t\middle|t_k\right) = \left[\mathbf{I} - \mathbf{K}(t_N)\mathbf{A}(t_N)\right]\mathbf{P}\left(t\middle|t_{k-1}\right) \end{cases} \qquad (3.68)$$

- For $t = t_0$, this formulation is equivalent to the initial LS parameter estimation problem, because parameters are always estimated at the same reference time t_0.
- By contrast, for $t = t_k$ the above equations lead to a filtering problem where parameters are estimated at each measurement time t_k. In that case, the values of $\hat{\mathbf{y}}_{diff}\left(t_k\middle|t_{k-1}\right)$ and $\mathbf{P}\left(t_k\middle|t_{k-1}\right)$ need first to be computed from the values $\hat{\mathbf{y}}_{diff}\left(t_{k-1}\middle|t_{k-1}\right)$ and $\mathbf{P}\left(t_{k-1}\middle|t_{k-1}\right)$ obtained at the previous time t_{k-1} through a *prediction step*, which thus requires to propagate the last estimate vector and error covariance matrix from time t_k to time t_{k-1}. The combination of the filtering equations with this step is the problem of *Kalman Filtering*.

3.3.1. Linearised discrete Kalman Filter

i. Time step prediction

As previously explained, the filtering problem allows the propagation model to be completed by a process noise term $\mathbf{w}(t)$. The propagation model equations can thus be here written as

$$\dot{\mathbf{y}}(t) = \mathbf{f}\left[\mathbf{y}(t), t\right] + \mathbf{w}(t) \qquad (3.69)$$

while the observation model keeps the same form as for the Least Squares problem.

This propagation model can be linearised as presented in paragraph 3.1.2, and its solution between observation times t_{k-1} and t_k can then be expressed as

$$\mathbf{y}_{diff}(t_k) = \Phi(t_k, t_{k-1})\mathbf{y}_{diff}(t_{k-1}) + \mathbf{w}'(t_k) \qquad (3.70)$$

where Φ and $\mathbf{w}'(t_k)$ are defined in paragraph 3.1.3. The values of $\hat{\mathbf{y}}_{diff}\left(t_k\middle|t_{k-1}\right)$ and $\mathbf{P}\left(t_k\middle|t_{k-1}\right)$ can then be computed from $\hat{\mathbf{y}}_{diff}\left(t_{k-1}\middle|t_{k-1}\right)$ and $\mathbf{P}\left(t_{k-1}\middle|t_{k-1}\right)$ through the *time step orbit prediction equations* which represents the expected values of a random process

$$\hat{\mathbf{y}}_{diff}\left(t_k \middle| t_{k-1}\right) = E\left[\Phi(t_k, t_{k-1}).\hat{\mathbf{y}}_{diff}\left(t_{k-1}\right) + \mathbf{w}'\left(t_k\right)\right] = \Phi(t_k, t_{k-1}).\hat{\mathbf{y}}_{diff}\left(t_{k-1} \middle| t_{k-1}\right) \quad (3.71)$$

$$\begin{aligned}
\mathbf{P}\left(t_k \middle| t_{k-1}\right) &= E\left[\left(\Phi(t_k, t_{k-1}).\hat{\mathbf{y}}_{diff}\left(t_{k-1}\right) + \mathbf{w}'\left(t_k\right)\right)\left(\Phi(t_k, t_{k-1}).\hat{\mathbf{y}}_{diff}\left(t_{k-1}\right) + \mathbf{w}'\left(t_k\right)\right)^T\right] \\
&= \Phi(t_k, t_{k-1}).E\left[\hat{\mathbf{y}}_{diff}\left(t_{k-1}\right).\hat{\mathbf{y}}_{diff}\left(t_{k-1}\right)^T\right].\Phi^T(t_k, t_{k-1}) \\
&\quad + \Phi(t_k, t_{k-1}).E\left[\hat{\mathbf{y}}_{diff}\left(t_{k-1}\right).\mathbf{w}'\left(t_k\right)^T\right] \\
&\quad + E\left[\mathbf{w}'\left(t_k\right).\hat{\mathbf{y}}_{diff}\left(t_{k-1}\right)^T\right].\Phi^T(t_k, t_{k-1}) \\
&\quad + E\left[\mathbf{w}'\left(t_k\right).\mathbf{w}'\left(t_k\right)^T\right] \\
&= \Phi(t_k, t_{k-1}).\mathbf{P}\left(t_{k-1} \middle| t_{k-1}\right).\Phi^T(t_k, t_{k-1}) + \mathbf{Q}'\left(t_k\right)
\end{aligned} \quad (3.72)$$

where the properties of the process noise given by equation **(3.8)** have been used.

ii. **Filtering equations**

The orbit determination is here realised through filtering equations from $\hat{\mathbf{y}}_{diff}\left(t \middle| t_{k-1}\right)$ to $\hat{\mathbf{y}}_{diff}\left(t \middle| t_k\right)$

$$\hat{\mathbf{y}}_{diff}\left(t_k \middle| t_k\right) = \hat{\mathbf{y}}_{diff}\left(t_k \middle| t_{k-1}\right) + \mathbf{K}(t_k)\left[\mathbf{z}_{diff}(t_k) - \mathbf{A}(t_k)\hat{\mathbf{y}}_{diff}\left(t_k \middle| t_{k-1}\right)\right] \quad (3.\ 73)$$

$$\mathbf{P}\left(t_k \middle| t_k\right) = \left[\mathbf{I} - \mathbf{K}(t_k)\mathbf{A}(t_k)\right]\mathbf{P}\left(t_k \middle| t_{k-1}\right) \quad (3.74)$$

where t has been replaced by t_k, and where the Kalman gain may be written as

$$\mathbf{K}(t_k) = \mathbf{P}\left(t_k \middle| t_{k-1}\right)\mathbf{A}(t_k)^T\left[\mathbf{A}(t_k).\mathbf{P}\left(t_k \middle| t_{k-1}\right).\mathbf{A}(t_k)^T + \mathbf{G}(t_k)\right]^{-1} \quad (3.75)$$

and with

$$\hat{\mathbf{y}}\left(t_k \middle| t_{k-1}\right) = \hat{\mathbf{y}}_{diff}\left(t_k \middle| t_{k-1}\right) + \mathbf{y}_{ref}\left(t_k\right) \quad (3.76)$$

$$\mathbf{z}_{diff}\left(t_k\right) = \mathbf{z}\left(t_k\right) - \mathbf{z}_{ref}\left(t_k\right) = \mathbf{z}\left(t_k\right) - \mathbf{h}\left[\mathbf{y}\left(t_k \middle| t_{k-1}\right), t_k\right] \quad (3.77)$$

3.3.2. Extended discrete Kalman Filter

The idea of the discrete Extended Kalman Filter (EKF) is to re-linearise around each new estimate as it becomes available

$$\mathbf{y}_{ref}(t) = \hat{\mathbf{y}}\left(t \middle| t_{k-1}\right) \quad (3.78)$$

$$\mathbf{y}_{diff}(t) = \mathbf{y}(t) - \mathbf{y}_{ref}(t) = \mathbf{y}(t) - \mathbf{y}\left(t \middle| t_{k-1}\right) \quad (3.79)$$

$$\mathbf{F}(t) = \left. \frac{\partial \mathbf{f}[\mathbf{y}(t),t]}{\partial \mathbf{y}(t)} \right|_{\mathbf{y}(t)=\mathbf{y}(t|t_{k-1})} \tag{3.80}$$

$$\mathbf{A}(t_k) = \left. \frac{\partial \mathbf{h}[\mathbf{y}(t),t]}{\partial \mathbf{y}(t)} \right|_{\mathbf{y}(t_k)=\mathbf{y}(t|t_{k-1})} \tag{3.81}$$

$$\hat{\mathbf{y}}_{diff}(t_k|t_{k-1}) = \mathbf{0} \tag{3.82}$$

The goal is thus to use a better reference trajectory as soon as it is available. As a consequence, large initial errors are not allowed to propagate through time and the linearity assumption is less likely to be violated, which allows to compensate for the fact that contrary to the LS formulation, the Kalman Filter is not iterated (the whole batch of observation is treated only once).

This permits to reformulate the filtering equations in terms of the original vector $\mathbf{y}$ instead of the difference vector $\mathbf{y}_{diff}$. Indeed, for the orbit prediction equations, $\hat{\mathbf{y}}(t_k|t_{k-1})$ may be computed through direct integration of the true motion equations, like for the iterative recursive Least Squares

$$\frac{d}{dt}\hat{\mathbf{y}}(t|t_{k-1}) = \mathbf{f}[\hat{\mathbf{y}}(t|t_{k-1}),t] \tag{3.83}$$

so the time step propagation equations are

$$\hat{\mathbf{y}}(t_k|t_{k-1}) = \hat{\mathbf{y}}(t_{k-1}|t_{k-1}) + \int_{t_{k-1}}^{t_k} \mathbf{f}[\hat{\mathbf{y}}(t|t_{k-1}),t]dt \tag{3.84}$$

$$\mathbf{P}(t_k|t_{k-1}) = \Phi(t_k,t_{k-1}).\mathbf{P}(t_{k-1}|t_{k-1}).\Phi^T(t_k,t_{k-1}) + \mathbf{Q}'(t_k) \tag{3.85}$$

The filtering equations then become

$$\mathbf{K}(t_k) = \mathbf{P}(t_k|t_{k-1})\mathbf{A}(t_k)^T\left[\mathbf{A}(t_k).\mathbf{P}(t_k|t_{k-1}).\mathbf{A}(t_k)^T + \mathbf{G}(t_k)\right]^{-1} \tag{3.86}$$

$$\hat{\mathbf{y}}(t_k|t_k) = \hat{\mathbf{y}}(t_k|t_{k-1}) + \mathbf{K}(t_k)\left[\mathbf{z}(t_k) - \mathbf{h}\left[\hat{\mathbf{y}}(t_k|t_{k-1}),t_k\right]\right] \tag{3.87}$$

$$\mathbf{P}(t_k|t_k) = [\mathbf{I} - \mathbf{K}(t_k)\mathbf{A}(t_k)]\mathbf{P}(t_k|t_{k-1}) \tag{3.88}$$

where equation (**3.77**) and (**3.82**)have been used, and $\hat{\mathbf{y}}(t_k|t_{k-1})$has been added to both sides of equation for the estimation vector $\mathbf{y}$.

3.3.3. Extended continuous-discrete Kalman Filter

In the discrete EKF, the prediction equations involve true equations of motion for orbit propagation, but the linearised form of these equations to propagate the error covariance matrix. Here, differential equations for covariance matrix are derived by differentiation with respect to t_k of this linearised propagation equation [Wel 97]

$$\begin{aligned}\frac{d}{dt}\mathbf{P}\left(t\middle|t_{k-1}\right) &= \left[\mathbf{F}(t)\Phi(t,t_{k-1})\right]\mathbf{P}\left(t_{k-1}\middle|t_{k-1}\right).\Phi^T(t,t_{k-1}) + \Phi(t,t_{k-1}).\mathbf{P}\left(t_{k-1}\middle|t_{k-1}\right).\left[\mathbf{F}(t)\Phi(t,t_{k-1})\right]^T \\ &\quad + \mathbf{Q}(t) + \mathbf{F}(t)\mathbf{Q}'(t) + \mathbf{Q}'(t)\mathbf{F}^T(t) \\ &= \mathbf{F}(t)\mathbf{P}\left(t\middle|t_{k-1}\right) - \mathbf{F}(t)\mathbf{Q}'(t) + \mathbf{P}\left(t\middle|t_{k-1}\right)\mathbf{F}^T(t) - \mathbf{Q}'(t)\mathbf{F}^T(t) \\ &\quad + \mathbf{Q}(t) + \mathbf{F}(t)\mathbf{Q}'(t) + \mathbf{Q}'(t)\mathbf{F}^T(t) \\ &= \mathbf{F}(t)\mathbf{P}\left(t\middle|t_{k-1}\right) + \mathbf{P}\left(t\middle|t_{k-1}\right)\mathbf{F}^T(t) + \mathbf{Q}(t)\end{aligned} \tag{3.89}$$

where terms in $\mathbf{Q}'$ cancel out.

Numerical integration of both equations **(3.83)** and **(3.88)** for estimated vector and covariance matrix then provides the desired propagation. These equations are integrated simultaneously in a way similar to the estimation vector and state transition matrix equations of the LS problem.

The prediction equations are now in a time-continuous differential form, whereas the filtering equations are still discrete at measurement times. This justifies the name of "continuous-discrete" extended Kalman Filter.

For each new observation, an orbit propagation is realised with the time-step continuous equations and an orbit determination is then realised with discrete extended filtering equations. Thus, unlike the LS algorithm, this process produces an estimation of $\mathbf{y}$ and its corresponding elements of the matrix $\mathbf{P}$ at each time of a new observation and each observation is processed only once. So this algorithm is a *sequential estimator*.

3.3.4. Interpretation of the position uncertainty

Due to the white Gaussian process noise, the vector $\mathbf{y}(t)$ is assumed to be a Gaussian multivariate. Its estimation $\hat{\mathbf{y}}(t)$ is the state vector which minimise the mean squared error such that

$$\begin{cases} E\left[\mathbf{y}\right] = \hat{\mathbf{y}} \\ E\left[(\mathbf{y}-\hat{\mathbf{y}})(\mathbf{y}-\hat{\mathbf{y}})^T\right] = \mathbf{P} \end{cases} \tag{3.90}$$

The probability density function of this normal distribution is thus

$$P(\mathbf{y}) = \frac{1}{\sqrt{(2\pi)^n \left\|\mathbf{P}\right\|}} \exp\left\{-\frac{1}{2}(\mathbf{y}-\hat{\mathbf{y}})^T \mathbf{P}^{-1}(\mathbf{y}-\hat{\mathbf{y}})\right\} \tag{3.91}$$

Moreover, as every symmetric positive definite matrix is diagonalisable with real positive eigenvalues (non-negative) and eigenvectors being pairwise orthogonal, and as $\mathbf{P}$ is non-negative definite symmetric, there exists an unitary matrix $\mathbf{U}$, with $\mathbf{U}\mathbf{U}^T = \mathbf{I}$, and a diagonal matrix $\mathbf{D}$ such that

$$\mathbf{P} = \mathbf{U}.\mathbf{D}.\mathbf{U}^T \qquad \text{with} \qquad \mathbf{D} = \begin{pmatrix} \sigma_1^{\ 2} & 0 & \dots & 0 \\ 0 & \sigma_2^{\ 2} & \dots & 0 \\ \dots & \dots & \dots & 0 \\ 0 & 0 & 0 & \sigma_m^{\ 2} \end{pmatrix} \tag{3.92}$$

This allows an interpretation of matrix **P** through its corresponding diagonal matrix **D**. The hyper-surfaces of constant probability are hyper-ellipsoids

$$(\mathbf{y} - \hat{\mathbf{y}})^T \mathbf{P}^{-1} (\mathbf{y} - \hat{\mathbf{y}}) = l^2 \tag{3.93}$$

where l is constant. The direction of the i^{th} principal axis of this hyper-ellipsoid is given by the i^{th} column of **U**, and its length is given by $l.\sigma_i$.

By a rotation of axes, these axes coincide with coordinate axes

$$\begin{cases} \mathbf{y}' = \mathbf{U}^T(\mathbf{y} - \hat{\mathbf{y}}) \\ E[\mathbf{y}'] = \mathbf{0} \\ E[\mathbf{y}'\mathbf{y}'^T] = \mathbf{U}^T.\mathbf{P}.\mathbf{U} = \mathbf{D} \end{cases} \tag{3.94}$$

such that components of **y'** are not correlated and statistically independent.

In practice, the interest is to know the probability η that the position **r** of the satellite lies inside an ellipsoid defined by the position of the estimate $\hat{\mathbf{r}}$ and the position error covariance matrix $\mathbf{P}_{rr}$

$$\mathbf{x} = \begin{pmatrix} \mathbf{r} \\ \dot{\mathbf{r}} \end{pmatrix} \qquad \text{and} \qquad \mathbf{P} = \begin{pmatrix} \mathbf{P}_{rr} & \mathbf{P}_{r\dot{r}} \\ \mathbf{P}_{\dot{r}r} & \mathbf{P}_{\dot{r}\dot{r}} \end{pmatrix} \tag{3.95}$$

where $\dot{\mathbf{r}}$ contains the velocity and other parameters components. This probability is given by $\Pr\left[(\mathbf{r} - \hat{\mathbf{r}})^T \mathbf{P}_{rr}^{\ -1} (\mathbf{r} - \hat{\mathbf{r}}) \leq l^2\right] = \eta$, and is obtained by integrating the probability density function $P(\mathbf{r})$ over the ellipsoid volume

$$\eta = \int_{(\mathbf{r}-\hat{\mathbf{r}})^T \mathbf{P}_{rr}^{\ -1}(\mathbf{r}-\hat{\mathbf{r}}) \leq l^2} P(\mathbf{r}) d\mathbf{r} \tag{3.96}$$

Using the above transformation $\mathbf{y}' = \mathbf{U}^T(\mathbf{y} - \hat{\mathbf{y}})$ and the other transformation $\mathbf{y}'' = \mathbf{D}^{-\frac{1}{2}}\mathbf{y}'$, the integration may be realised over an unitary sphere

$$\eta = \frac{1}{\sqrt{(2\pi)^6}} \int_0^l \exp\left(-\frac{1}{2}r^2\right) f(r) dr \tag{3.97}$$

where $r = \|\mathbf{r}''\|$ and $f(r)dr = 4\pi r^2 dr$ is the spherically symmetric volume element. Finally, the probability can be written as a function of l

$$\eta(l) = \sqrt{\frac{2}{\pi}} \int_0^l \exp\left(-\frac{1}{2}r^2\right) r^2 dr = erf\left(\frac{l}{\sqrt{2}}\right) - \sqrt{\frac{2}{\pi}}.l.\exp\left(-\frac{1}{2}l^2\right) \tag{3.98}$$

Inversely, for a given probability η, the extension l of the ellipsoid can be calculated. A common value for l is 3, and the probability for the satellite to be within the 3σ ellipsoid is 97.07 %. The following table shows some values of l versus η. For more details about this interpretation of the error covariance matrix, refer to the literature, e.g. [Bry 75].

L	η	η	l
1	0.1987480430988	0.9	2.50027771080941
2	0.73855358700509	0.99	3.36821417521873
3	0.97070911346511	0.999	4.03314222365619
4	0.99886601571021	0.9999	4.59429139978745
5	0.99998455950171	0.99999	5.08937616468373

Table 6.1

3.3.5. Factorisation of the filtering equations

As in the case of the LS method, numerical stability of the Kalman Filter may be improved by means of factorisation methods. In particular, these methods prevent the estimation covariance matrix from becoming asymmetric or negative definite. To this end, the filtering equations for the Kalman gain and the covariance update are rewritten as

$$\mathbf{K}(t_k)\mathbf{L}(t_k) = \mathbf{P}\left(t_k \middle| t_{k-1}\right)\mathbf{A}(t_k)^T \tag{3.99}$$

$$\mathbf{P}\left(t_k \middle| t_k\right) + \mathbf{K}(t_k)\mathbf{L}(t_k)\mathbf{K}^T(t_k) = \mathbf{P}\left(t_k \middle| t_{k-1}\right) \tag{3.100}$$

with the known matrices on the right-hand side and the matrices to be calculated on the left-hand side, and where

$$\mathbf{L}(t_k) \overset{def}{=} \mathbf{A}(t_k).\mathbf{P}\left(t_k \middle| t_{k-1}\right).\mathbf{A}(t_k)^T + \mathbf{G}(t_k) \tag{3.101}$$

These three equations may be written into a matrix form

$$\begin{pmatrix} \mathbf{L}(t_k) & \mathbf{L}(t_k)\mathbf{K}^T(t_k) \\ \mathbf{K}(t_k)\mathbf{L}(t_k) & \mathbf{P}\left(t_k \middle| t_k\right) + \mathbf{K}(t_k)\mathbf{L}(t_k)\mathbf{K}^T(t_k) \end{pmatrix} = \begin{pmatrix} \mathbf{A}(t_k).\mathbf{P}\left(t_k \middle| t_{k-1}\right).\mathbf{A}(t_k)^T + \mathbf{G}(t_k) & \mathbf{A}(t_k).\mathbf{P}\left(t_k \middle| t_{k-1}\right) \\ \mathbf{P}\left(t_k \middle| t_{k-1}\right).\mathbf{A}^T(t_k) & \mathbf{P}\left(t_k \middle| t_{k-1}\right) \end{pmatrix} \tag{3.102}$$

which becomes in terms of the square root matrices of both sides

$$\begin{pmatrix} \mathbf{L}^{\frac{1}{2}}(t_k) & \mathbf{0} \\ \mathbf{K}(t_k)\mathbf{L}^{\frac{1}{2}}(t_k) & \mathbf{P}^{\frac{1}{2}}\left(t_k \middle| t_k\right) \end{pmatrix} \begin{pmatrix} \mathbf{L}^{\frac{T}{2}}(t_k) & \mathbf{L}^{\frac{T}{2}}(t_k)\mathbf{K}^T(t_k) \\ \mathbf{0} & \mathbf{P}^{\frac{T}{2}}\left(t_k \middle| t_k\right) \end{pmatrix} = \begin{pmatrix} \mathbf{G}^{\frac{1}{2}}(t_k) & \mathbf{A}(t_k).\mathbf{P}^{\frac{1}{2}}\left(t_k \middle| t_{k-1}\right) \\ \mathbf{0} & \mathbf{P}^{\frac{1}{2}}\left(t_k \middle| t_{k-1}\right) \end{pmatrix} \begin{pmatrix} \mathbf{G}^{\frac{T}{2}}(t_k) & \mathbf{0} \\ \mathbf{P}^{\frac{T}{2}}\left(t_k \middle| t_{k-1}\right).\mathbf{A}^T(t_k) & \mathbf{P}^{\frac{T}{2}}\left(t_k \middle| t_{k-1}\right) \end{pmatrix} \tag{3.103}$$

If $\mathbf{S}$ and $\mathbf{T}$ are square roots of a same positive definite matrix $\mathbf{M}$, $\mathbf{M} = \mathbf{S}\mathbf{S}^T = \mathbf{T}\mathbf{T}^T$, there exists an orthogonal matrix $\mathbf{Q}$ such that $\mathbf{S} = \mathbf{Q}\mathbf{T}$, and

$$\begin{pmatrix} \mathbf{L}^{\frac{1}{2}}(t_k) & \mathbf{0} \\ \mathbf{K}(t_k)\mathbf{L}^{\frac{1}{2}}(t_k) & \mathbf{P}^{\frac{1}{2}}(t_k|t_k) \end{pmatrix} = \mathbf{Q}.\begin{pmatrix} \mathbf{G}^{\frac{1}{2}}(t_k) & \mathbf{A}(t_k).\mathbf{P}^{\frac{1}{2}}(t_k|t_{k-1}) \\ \mathbf{0} & \mathbf{P}^{\frac{1}{2}}(t_k|t_{k-1}) \end{pmatrix} \tag{3.104}$$

Then a sequence of Givens rotations may be used to rend to zero the upper right sub-matrix on the left-hand side. The estimation vector is thus updated with the sub-matrices of the left-hand side.

$$\hat{\mathbf{y}}(t_k|t_k) = \hat{\mathbf{y}}(t_k|t_{k-1}) + \left(\mathbf{K}(t_k)\mathbf{L}^{\frac{1}{2}}(t_k)\right)\mathbf{L}^{-\frac{1}{2}}(t_k)\left[\mathbf{z}(t_k) - \mathbf{h}\left[\hat{\mathbf{y}}(t_k|t_{k-1}), t_k\right]\right] \tag{3.105}$$

and the error covariance matrix becomes

$$\mathbf{P}(t_k|t_k) = \mathbf{P}^{\frac{1}{2}}(t_k|t_{k-1})\mathbf{P}^{\frac{T}{2}}(t_k|t_{k-1}) \tag{3.106}$$

3.3.6. Filter divergence

A problem of the Kalman Filter, known as *filter divergence* ought to be mentioned here. Its cause is the *a posteriori* covariance matrix $\mathbf{P}(t_k|t_k)$ which becomes smaller than the *a priori* one $\mathbf{P}(t_k|t_{k-1})$ at each processing of a new data, as shown by equation **(3.83)**.

When operating over many data as it is the case in satellite orbit determination, the error covariance matrix and the filter gain $\mathbf{K}(t_k)$, which depends directly on $\mathbf{P}(t_k|t_{k-1})$, may become very small and subsequent observations have a little effect on the estimate. If the dynamic model in the filter is different from the real dynamic behaviour of the system, the estimations of the trajectory vector may then diverge from the true state of the satellite. The onset of such a divergence manifests itself by the inconsistency of the residuals with their predicted statistics : they become biased and larger in magnitude.

A way to increase the error covariance matrix between measurements and thus compensate for the unmodeled accelerations is to increase the input spectral density matrix of the propagation noise $\mathbf{Q}(t)$.

Chapter 4.

The simulator software design and description

4.1. Estimations and artificial tracking data

4.1.1. The Orbit and Kalman processes

The orbit determination and propagation program, *ORBIT* [Mon 96] is used by SES for station-keeping of geostationnary satellites. Some years ago, SES has developed a Quasi-Real Time Orbit Determination application (QRTOD) based on this software. This application acts as a scheduler for ORBIT, which processes tracking data arcs over a moving time window, the previous estimate being usually taken as the initial guess in order to guarantee convergence of LS in only a few iterations.

The EKF implementation, KALMAN [Wel 97], realised upon the program ORBIT does not have the same level of sophistication as the QRTOD, as it is essentially used for evaluation purposes.

Moreover, this implementation is limited in its capability to evaluate manoeuvres ; once a new manoeuvre takes place, the filter's covariance matrix needs to be updated, practically done by stopping the filter once the estimation of the previous manoeuvre has converged, and restarting it with the new manoeuvre prediction. Furthermore, it cannot estimate measurement biases on the contrary of the LS implementation.

Nevertheless, these two programs run in a similar way, as they receive an IERS file [McC 96] (which contains data for time and position transformations) and a tracking measurements file in GEOS-CX format [Mon 96] as inputs, and yield various outputs like the summary of orbit determination, the residual plot of estimation, station keeping and acquisition plots, the residual list and artificial tracking data.

They are both run through an interface, the SETUP file [Mon 96], which contains a predefined sequence of input data (initial epoch of estimation, satellite and stations data, force model parameters, orbit propagation and prediction information) needed for OD and OP.

4.1.2. The Tracking Data Simulator process

The TDS program [Fra 96] has been developed at SES, and consists in four parts :

- The starting part produces a reference OD and OP with the ORBIT program.
- The *TRACK_S1* process produces a fitting of the residuals outgoing of ORBIT.
- The *TRACK_S2* process simulates tracking data from an error free artificial tracking data by adding a white Gaussian noise term values and an *error model*[1] to the reference tracking data, using a random generator to avoid systematic bias.
- The *TRACK_S3* process produces a statistical analysis based on an OD of the simulated data and on the reference OP.

4.2. Simulator process

The goal of this new software procedure, named *MCSIM*, is to compare in a systematic way the performances of the implemented LS and EKF methods. It runs following the Monte-Carlo Simulation principle to assess through a statistical analysis of the results which method is best suited to a trilateration network [Wau 95].

Previous work has been realised on this subject. A DCL (Dec Command Language) command file named *GO.COM* [Wei 97] was developed to compare the two estimation methods by calling successively three processes. The first one estimates the state vectors at successive epochs with the program KALMAN. The second one computes a Reference Trajectory at times corresponding with the program ORBIT. And the third one estimates the initial state vector, and to propagate it to the same corresponding times with the same program ORBIT. The input measurements for these two estimations come from the TDS application. The whole procedure is used as a black box whose outputs are the differences between the reference trajectory and each of the two estimations at the successive epochs of EKF estimations.

The problem of such an implementation is the lack of flexibility in the simulations. A more systematic and integrated way of working implies to redesign the comparison and review the different black boxes to reconstruct a new process. This new process has been realised so that the user acts easier on the different parts of the process. Moreover, it can run with or without real trilateration tracking data.

The new simulator compares the errors of both LS and EKF estimations of an artificial trajectory over a tracking window of typically two days, the artificial trajectory including the effects of random perturbations and being subsequently observed via the observation function **h** plus a white Gaussian observation noise **v**, and an input process noise spectral density matrix being added to the propagation model of the EKF. The results of both estimations are finally compared to the error free reference trajectory at eight successive estimation epochs. The best method is the one which gives estimations nearest to this reference.

[1] This error model is described in details in Chapter 6.

As the two computer programs ORBIT and KALMAN are written in the Ada language and run on the AXP/VMS operating system, the development of the MCSIM procedure has been done in the same environment to reuse the existing programs.

The new process is implemented in six different phases. The first two correspond to an **initial phase** of the process run only once, the others constitute the **running phase** run for each simulation attempts

a. The process starts with the creation of an error free 'reference' artificial tracking data set and a noise model, in a way similar to the starting part of the TDS. At the same time, a corresponding dense reference ephemeris is created. This initial part can run with or without real tracking data.

b. The dense reference ephemeris is used to build a cubic-spline interpolation table of these data [Fra 95].

c. The stage TRACK_S2 of the TDS uses an input *error model*[2], the 'reference' error free artificial tracking data and the noise model to produce simulated tracking data.

d. The two procedures ORBIT and KALMAN are then run with these simulated data, and both create an estimation file[3]. The EKF file contains all estimations and their successive epochs of estimation, whereas the LS one contains propagated estimations at eight chosen epochs[4].

e. Next, from the interpolation table and the two estimation files, reference state vectors are constructed at respective epochs corresponding to LS and EKF estimations.

f. Finally the differences between the corresponding reference and estimation files are computed. From the EKF differences, only the eight ones corresponding to LS epochs are needed, and are thus extracted from the EKF differences file. These two sets of differences are the final outputs of the simulator.

The whole new process can be decomposed in three main parts. The first one is composed of points *a* and *b* ; it is called **Stage A** and initiates the process. The second part is composed of points *c* and *d* ; it is called **Stage B** and makes the two estimations. The last part *e* is called **Stage C** and realises the comparison between reference and estimations.

Note :

There are two possible initial phases of running, depending on the availability or not of real trilateration tracking data.

- When there is no real data, the point *a* only makes an OP of an initial state vector to build the dense ephemeris and the error free simulated tracking data.
- When real data are available, the point *a* processes them through an OD, before OP, to produce the dense reference ephemeris, and TRACK_S1 produces the fit of corresponding residuals

[2] The detailed description of this error model is presented in Chapter 6.

[3] EKF gives estimated state vector at increasing times. Least Square Fit gives only an initial state vector, it is then necessary to propagate it at different epochs.

[4] Each epoch correspond to the end of a measurement interval, i.e. 3, 6, 9, 12, 18, 24, 36 and 48 hours.

4.3. Process analysis

The methodology of the process design is based on three phases of information analysis which allow a better understanding of the implementation [Fra 94].

- The first phase is a structured analysis of all data flows and processes. It is represented by means of *Data Flow Diagrams* (DFD). These diagrams show the interactions between the different parts of the process. Such diagrams are generally presented according to a *top-down* approach. Starting from the most general scheme (top of the tree), the analyst can reach more and more refined diagrams, showing in turn more and more details of the process. In the present work, three levels of this top-down approach are presented. These diagrams are completed by a *Data Dictionary* identifying the properties of main data flow, files and internal processes.
- The second one consists in the presentation of the whole process and of all the procedures, programs and subroutines. This is done in a *Call Tree Diagram* which shows their succession and links. This diagram is completed by a *Command Procedure Description* and *Environment descriptions*.
- The third one, called *Hierarchical Design*, consists in a complete description of the different procedures and programs of the process. They are presented through their summary, logical structure and meaning, interfaces, input and output files, and local data (in the '*Subroutines Specifications*').

After this study, the whole process was implemented and a test phase was realised to ensure the validity of the different parts of the process.

Note : The hierarchical design is presented in the **Annexe A1**.

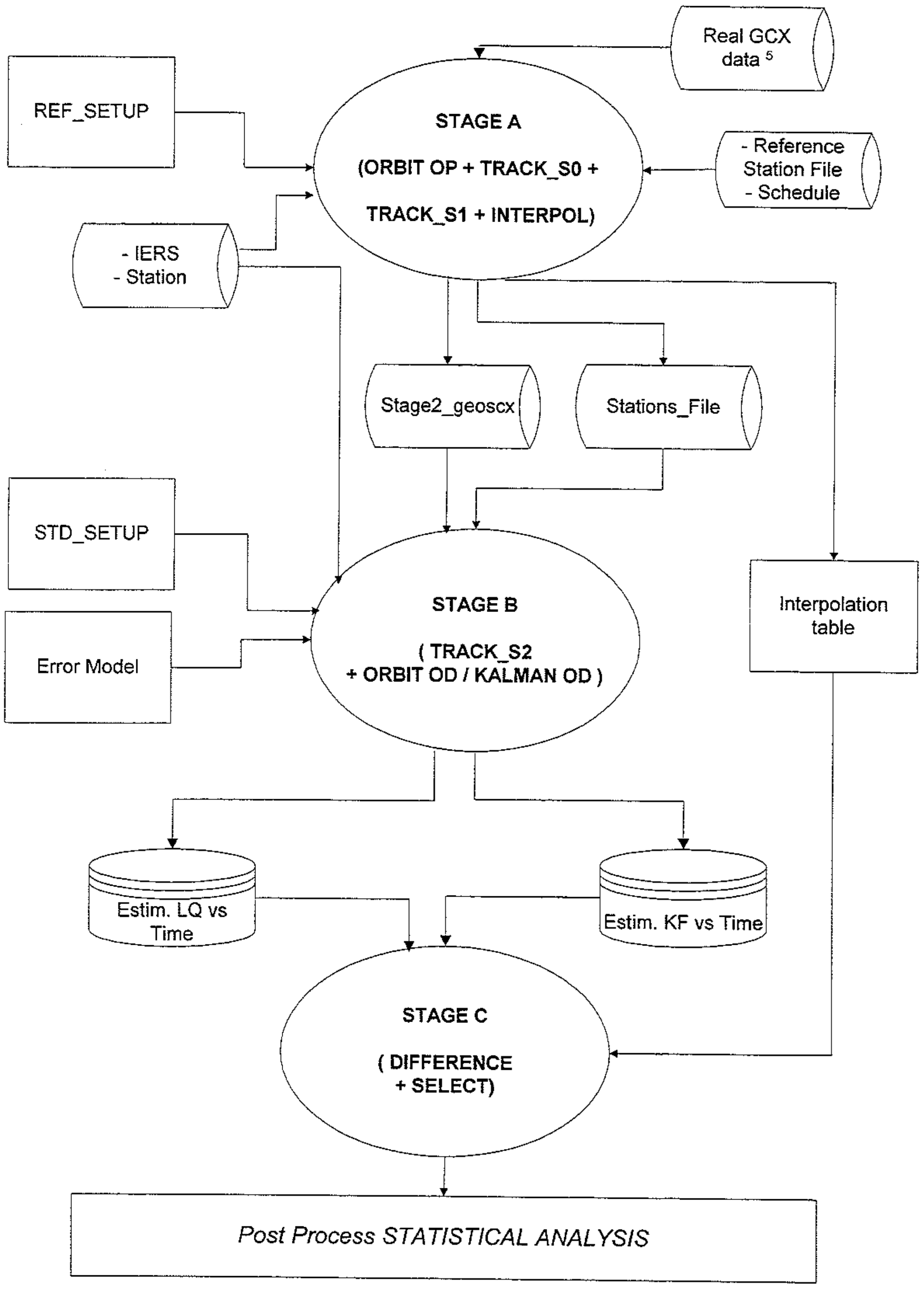
4.4. General software design and description

4.4.1. Data Flow Diagram

The different DFD are presented here in the following scheme according to the top-down approach :

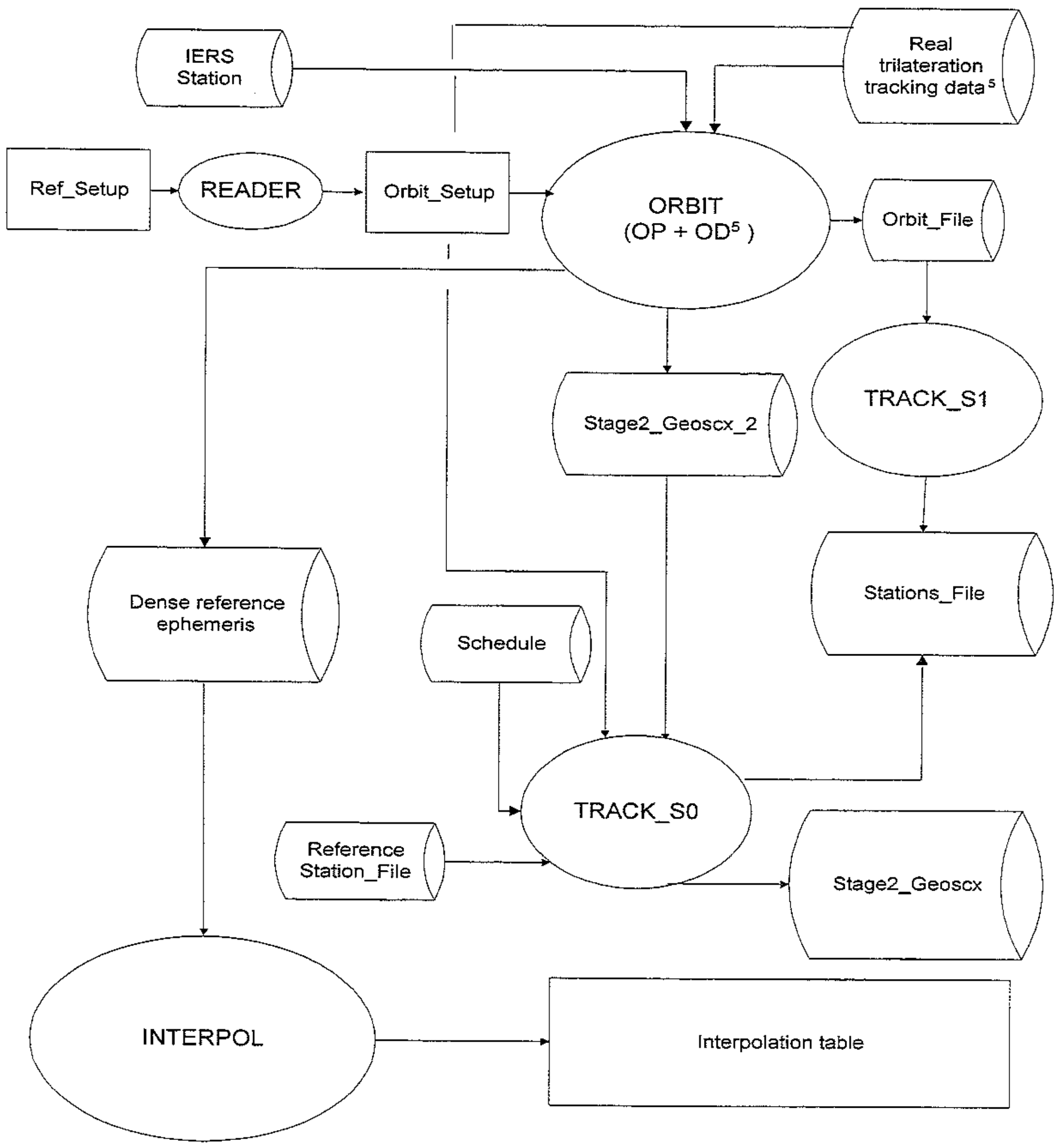
- The first diagram shows the three main stages of the process and their interactions in a global presentation (first level, or context diagram).
- The next three diagrams show these different stages (second level). They are completed by two other detailed little diagrams (third level).
- The last diagram is a complete one, like the first diagram, but with all interactions and data flows, and with numeration for the Data Dictionary.

i. Overall process (first level)



ii. Stage A (second level)

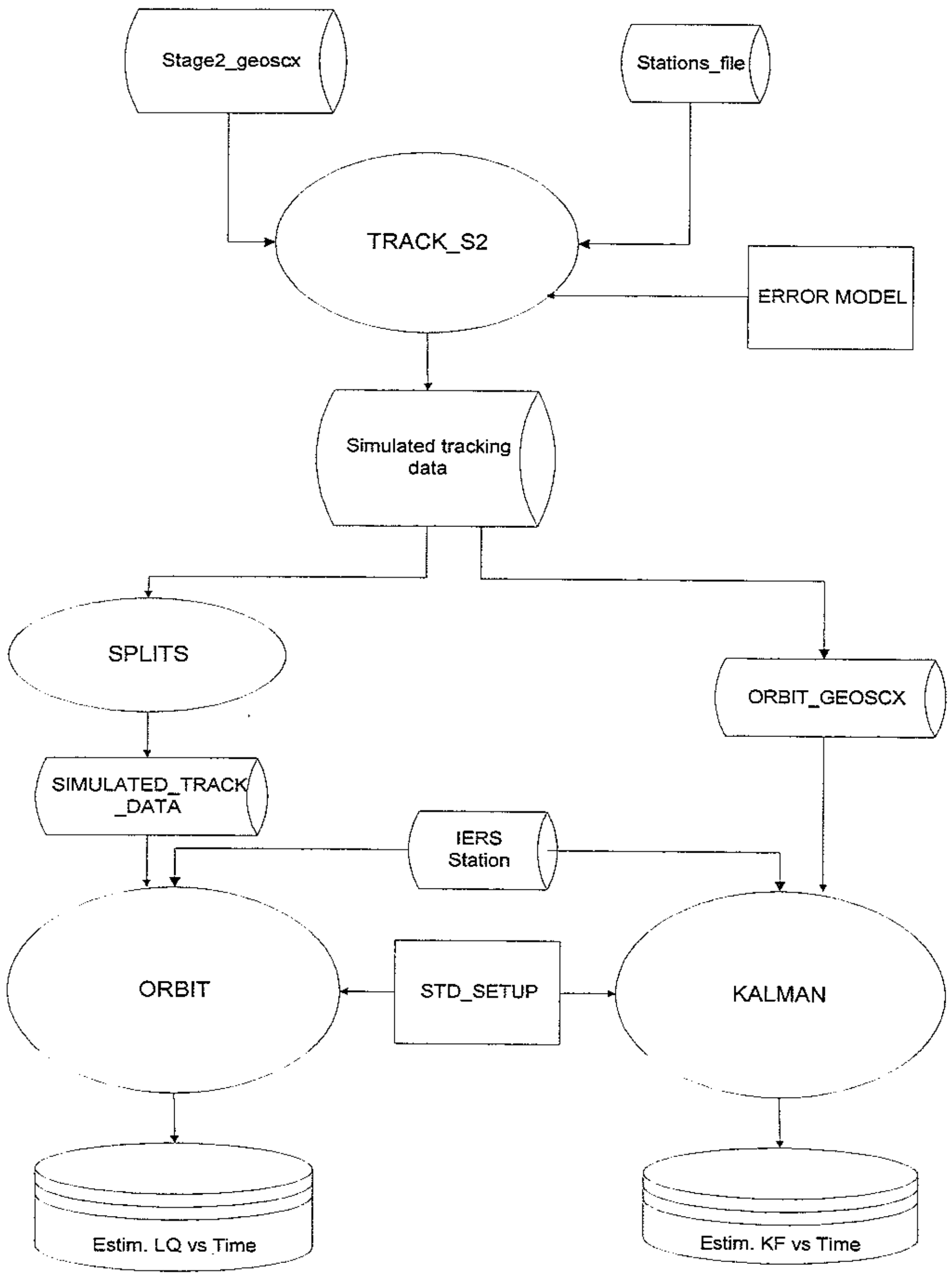
In this DFD the first stage is developed throughout its three main parts: the ORBIT, INTERPOL, TRACK_S0 and TRACK_S1[3] programs. The inputs of this stage are the IERS and Station files, the SETUP and Real Tracking Data[5] files, the Schedule and Reference_Station_File, while the outputs are the Stage2_geoscx, Stations_file[6], and the Interpolation Table.



[5] Only if real trilateration tracking data are available at the beginning of the procedure.
[6] The Stations_File is created by ORBIT when real data are available, and by TRACK_S0 in the other case

iii. **Stage B**

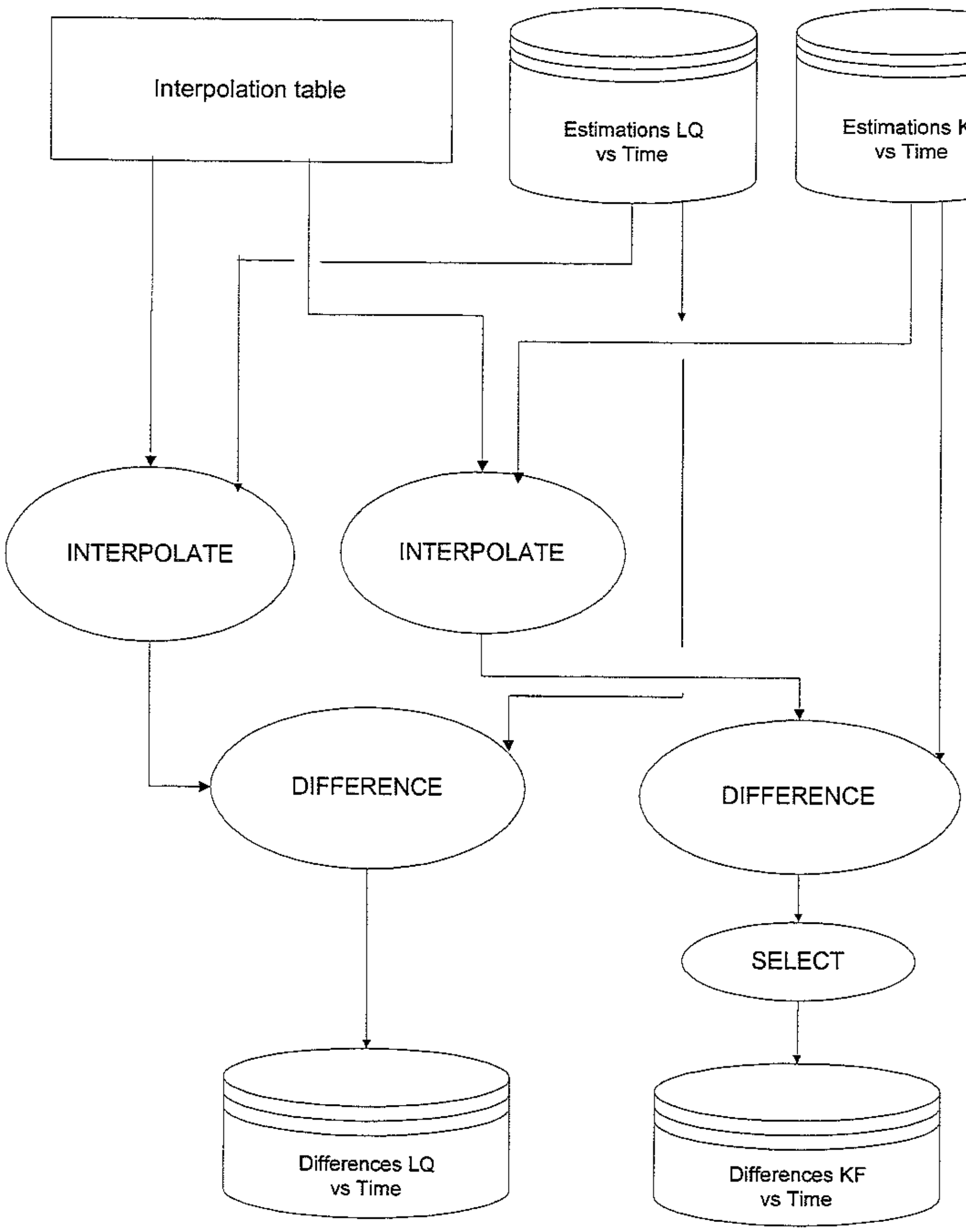
In this DFD the second stage is developed throughout its two main parts : the TRACK_S2 and the two estimation programs ORBIT and KALMAN. The inputs of this stage are the Error model file, the Stage2_geoscx and the Station_file, while the outputs are the two Estimation versus Time files (made and stored[7] at each run of the process).



[7] Not necessary.

iv. Stage C

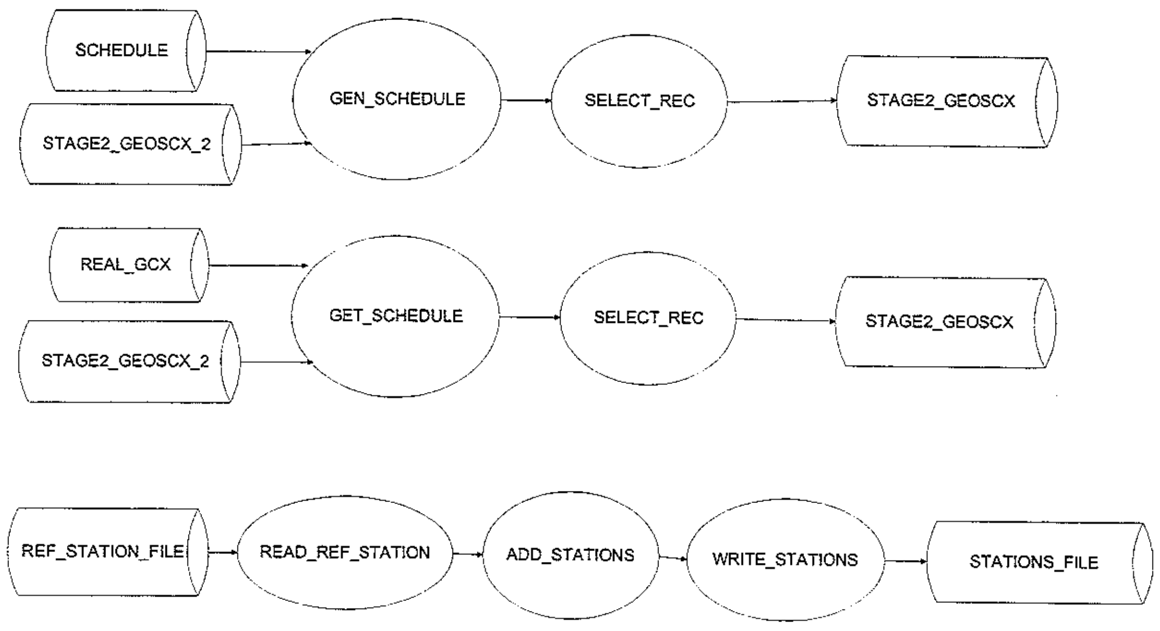
In this DFD the third stage is developed throughout its two main parts : the INTERPOLATE subroutine and the difference part (DIFFERENCE and SELECT). The inputs are the Interpolation Table and the two Estimation versus Time files, while the outputs are the two Difference versus Time files (stored at each run of the process).



V. Detailed parts (third level)

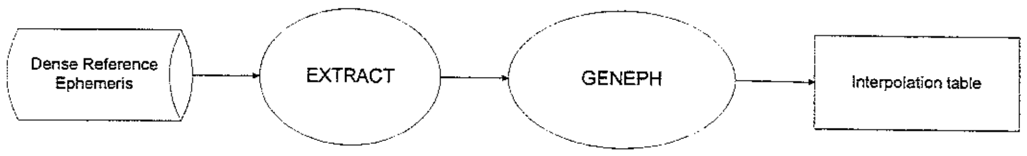
• TRACK_S0

First, the TRACK_S0 program of Stage A is presented at this third level of the DFD with its five subroutines WRITE_STATIONS, READ_STATION, GET_SCHEDULE or GEN_SCHEDULE and SELECT_REC, its inputs are the Schedule, Real.Gcx file, Ref_Stations_File and Stage2_Geoscx_2 file, while its outputs are the Stations_File and Stage2_Geoscx file.

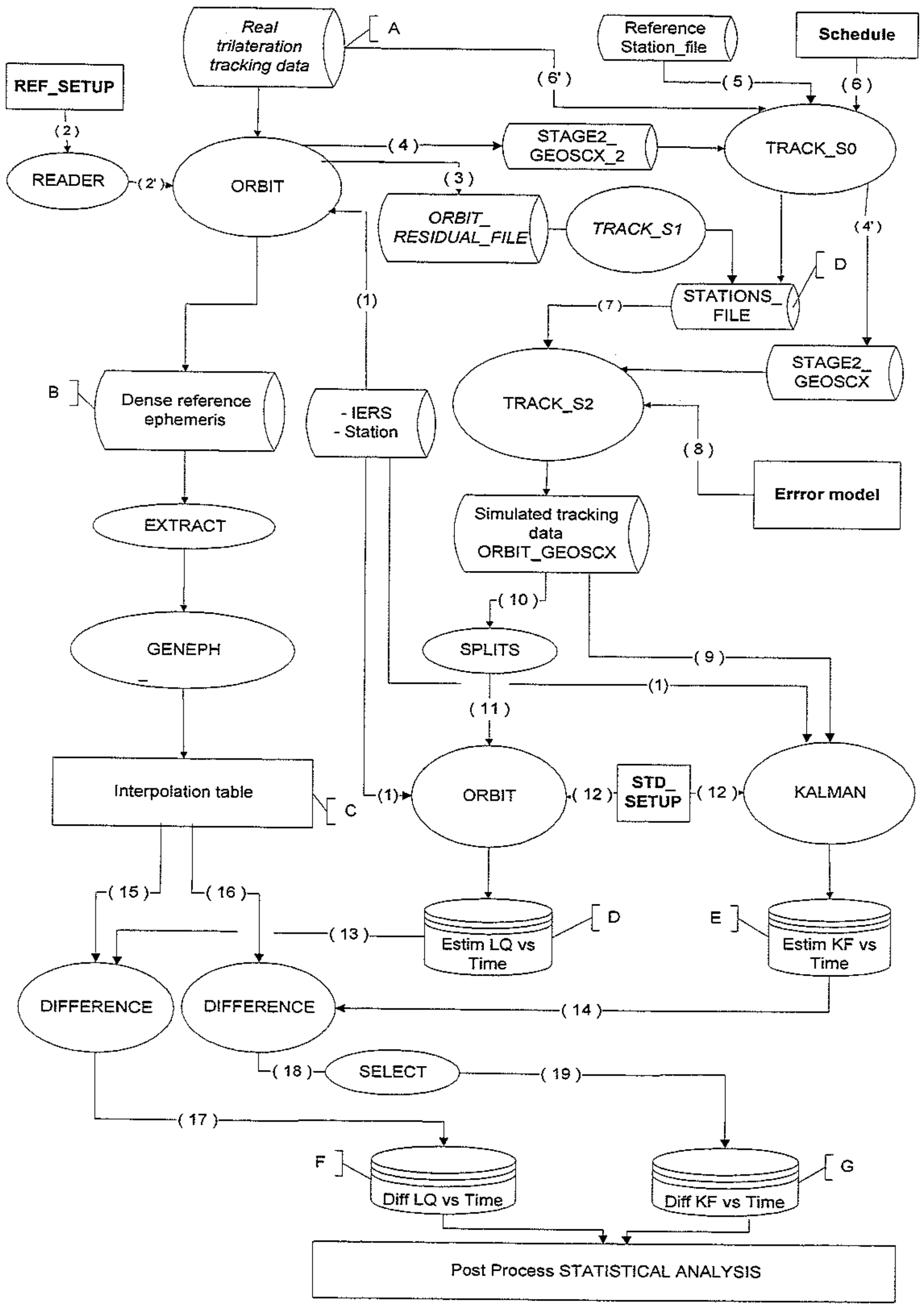


• INTERPOL

Secondly, the INTERPOL program of Stage A is here presented with its two main subroutines EXTRACT and GENEPH, its Dense Reference Ephemeris input and its Interpolation Table output.



vi. The global D.F.D[8]



[8] In this DFD, names in bold are user interfaces while those in italic are used (or exists) with real data on input.

4.4.2. Data Dictionary

i. Processes

Programs and subroutines	Description and using
READER	When there is no real tracking data, this program reads the REF_SETUP and create the ORBIT_SETUP containing the same information except the OD part which is not needed for the reference trajectory creation (by ORBIT) in this case.
ORBIT	1. The first application of this program uses the REF_SETUP (or the ORBIT_SETUP) and the IERS-Station to make an OD of real tracking data (with REF_SETUP, and when data are available) and an OP to a specified epoch (with a initial state vector if no real data are available and ORBIT_SETUP). Its three outputs are the Dense Reference Ephemeris REF.OUT, the Error Free Artificial Tracking Data STAGE2_GEOSCX_2, and (when real data are available) the ORBIT_RESIDUAL_FILE. 2. The second application of this program is done at the same level that the EKF program (Stage B). It creates the estimation file LSQ_SV.LQ with making an OD of the different Simulated Tracking Data file (the files TDS.GCX) and an OP, with the estimation set-up STD_SETUP.
TRACK_S0	• When real trilateration measurements are available, this program writes in the STAGE2_GEOSCX file the data of STAGE2_GEOSCX_2 at times of the real GCX file. • When no real data are available this program first does as above for the ATD file, with constructed times (from the Schedule) instead of real GCX ones. Secondly it creates the missing STATIONS_FILE from a Reference Station File of one station (whose noise is considered to be the same for the different stations).
GET_SCHEDULE	When real data are available, this subroutine of TRACK_S0 creates a measurement time structure from times of the real GCX file.
GEN_SCHEDULE	When no real data are available, this subroutine of TRACK_S0 creates a measurement time structure with the information of the Schedule.
READ_STATION / WRITE_STATIONS	These two subroutines (of TRACKLIB), used in TRACK_S0 when there is no real data, create the missing STATIONS_FILE using a Reference Station File. The first one read the REF_STATION_FILE and the second one copy, as many times as number of stations, these data in the STATIONS_FILE.
SELECT_REC	This last subroutine of TRACK_S0 forms the STAGE2_GEOSCX with data of the STAGE2_GEOSCX_2 at times of the above constructed structure.
TRACK_S1	This program is used when real trilateration tracking data are available. It creates the STATIONS_FILE with the ORBIT_RESIDUAL_FILE as input.
TRACK_S2	This program uses the STAGE2_GEOSCX and the STATIONS_FILE to simulate tracking data (form an ORBIT_GEOSCX file) including the measurement errors introduced in the error model file MODEL_FILE .
INTERPOL	This program constructs the Interpolation Table from the Dense Reference Ephemeris file.
EXTRACT	This subroutine of INTERPOL reads the REF.OUT file, and extracts out of it the Dense Reference Ephemeris needed in the subroutine GENEPH.
GENEPH	This other subroutine of INTERPOL builds a table containing second derivatives of the cubic spline interpolated function of the points contained in the Dense Reference Ephemeris.
SPLITS	This program splits the ORBIT_GEOSCX file into different files (TDS _'I'.GCX) used by the second application of the program ORBIT.

WRITE_LAST_SV_LQ	This subroutine is used at the end of the second application of ORBIT. It only computes and stores in the LSQ_SV.LQ the Least Squares estimation at the last epoch of simulated tracking data.
KALMAN	This program is the implementation of the EKF Filtering. It uses the ORBIT_GEOSCX data as input and creates the EKF_SV.KF estimation file, with the STD_SETUP file and the IERS table as inputs.
DIFFERENCE	This program realises the difference between corresponding Cartesian interpolated reference ephemeris and estimation state vectors of each method.
INTERPOLATE	This subroutine (of PANLIB), used in DIFFERENCE, creates Cartesian interpolation files of the Dense reference Ephemeris at EKF and LS estimation times : it reads the Interpolation Table and time(s)[9] in the estimations file (LSQ_SV.LQ or EKF_SV.KF) and create the SVI vector(s).
SELECT	This program selects the eight EKF differences and interpolated vectors corresponding to the epochs of the Least Squares differences file.

ii. Flows

N°	Name	Type	Using
1	• STATION.DAT (STATION_FILE) • NEWMAS.DAT (IERS_TABLE)	See STATION_.ADA See IERS_.ADA	• Data of the stations. • Data used to make time transformations.
2	REF.SETUP (REF_SETUP)	See SETUP_.ADA	Set-up of the reference trajectory. Input of READER.
2'	ORBIT.SETUP (ORBIT_SETUP)	See SETUP_.ADA	Output of READER, set-up of first ORBIT application.
3	REF.OUT (ORBIT_RESIDUAL_FILE)	Y(6), MJD : REAL*8	Reference ephemeris data, output of ORBIT and input of TRACK_S1. Does exists only if real tracking data.
4	REF.ATD (ORBIT_ATD ⇔ STAGE2_GEOSCX_2)	See GCX structure of the observations in GEOSLIB	Dense error free artificial tracking data : output of ORBIT, input of TRACK_S0
4'	REF2.ATD (STAGE2_GEOSCX)	See GCX structure of the observations in GEOSLIB	Output of TRACK_S0, input of TRACK_S2. Error free artificial tracking data at real or schedule times.
5	REF_STATION.DAT (REF_STATION_FILE)	ASCII free format sequential file (see T.D.S. software description of Stations_File)	Input of TRACK_S0, used to create the Stations_File when there is no real data.
6	SCH.DAT (SCHEDULE)	ASCII free format sequential file	Contains information of stations measurement and interval times.
6'	REAL.GCX (ORBIT_GEOSCX)	See GCX structure of the observations in GEOSLIB : %MJD	Its times are used when real data are available by TRACK_S0 to produce the Stage2_Geoscx.
7	STATIONS.DAT (STATIONS_FILE)	ASCII free format sequential file (see T.D.S. software description)	Data of the different stations, produced by TRACK_S1 or TRACK_S0 depending on availability of real data.

[9] In LS file there is only last epoch estimation, in KF file there are estimations at successive epochs.

8	SIM.MOD (MODEL_FILE)	See TRACKLIB.F90	Data with the different parameters of the perturbation model .
9	SIM.GCX (ORBIT_GEOSCX)	See structure of the observations in GEOSLIB	GCX output data of the TRACK_S2, contains simulated tracking data, input of KALMAN.
10	INPUT.GCX (INPUT)	See structure of the observations in GEOSLIB	Same than 9, but input of SPLITS.
11	TDS_'I'.GCX (ORBIT_GEOSCX)	See structure of the observations in GEOSLIB	Splited GCX tracking data, input of the second ORBIT.
12	STD.SETUP (STD_SETUP)	See SETUP_.ADA	Common set-up data of KALMAN and second ORBIT application.
13 / 14	LSQ_SV.LQ / EKF_SV.KF (LQ_ESTIM / KF_ESTIM)	MJD : REAL*8 VECTOR 6D : REAL*8	Epochs and corresponding estimations (LS & EKF), used to form interpolated reference files and make the differences.
15 / 16	SPLINE.TBL (TABLE)	EPH%NSD : REAL*8	Second derivatives of the interpolated function of the dense reference data.
17 / 18	DIF_'SIMUL'.LQ / DIFF_'SIMUL'.KF [10] (LQ_DIFF / KF_DIFF)	MJD : REAL*8 VECTOR 6D : REAL*8 VECTOR 6D : REAL*8	Differences between estimations and interpolated reference ephemeris at corresponding epochs, and interpolated reference ephemeris.
19	DIF_'SIMUL'.KF (DIF_MC)	MJD : REAL*8 VECTOR 6D : REAL*8 VECTOR 6D : REAL*8	EKF differences at the eight epochs corresponding to those of the LS, and interpolated reference ephemeris at these epochs.

iii. Storage

N°	Data file	Format	Using
A	REAL.GCX	See OBSERVATION structure in GEOSLIB	External file of real trilateration measurement.
B	REF.OUT	ASCII file Y(6), MJD : REAL*8	Dense reference ephemeris, produces by ORBIT with a small time step.
C	SPLINE.TABLE	Binary file	Interpolation table.
D/E	LSQ_SV.LQ / EKF_SV.KF	Y(6), MJD REAL*8	Estimated state vectors file for LS and EKF method, and corresponding epochs[11].
F/G	DIF_MC_'SIMUL'.LQ / DIF_ MC_'SIMUL'.KF	MJD : REAL*8 VECTOR 6D : REAL*8 VECTOR 6D : REAL*8	Contains the N x 8 LS / EKF differences and interpolated vectors at corresponding times [12].

[10] The 'SIMUL' part is composed of 6 digits XXXYYZ. For each tested case, XXX indicates the number of times the process has been run, YY and Z the number of set-up and model files used.
[11] This storage is updated at each new run of the process.
[12] This storage is appended at each new run of the process for a given set of input data, N being the number of times the « running phase » is run.

4.4.3. Command procedure description

The master process implementation is in agreement with the following Call Tree. This process is implemented in a DCL language as a command file. It contains the three main parts described before, each part being composed of different programs and subroutines written in Ada or Fortran-90, and linked together following the DFD and in agreement with the upper dictionary table.

Thus come the three procedures STAGE_A, STAGE_B and STAGE_C, each one composed of the following programs

The STAGE_A runs READER, ORBIT, INTERPOL, TRACK_S0 and TRACK_S1.
The STAGE_B runs TRACK_S2, KALMAN, SPLITS and ORBIT.
The STAGE_C runs DIFFERENCE and SELECT.

These programs call the main following subroutines

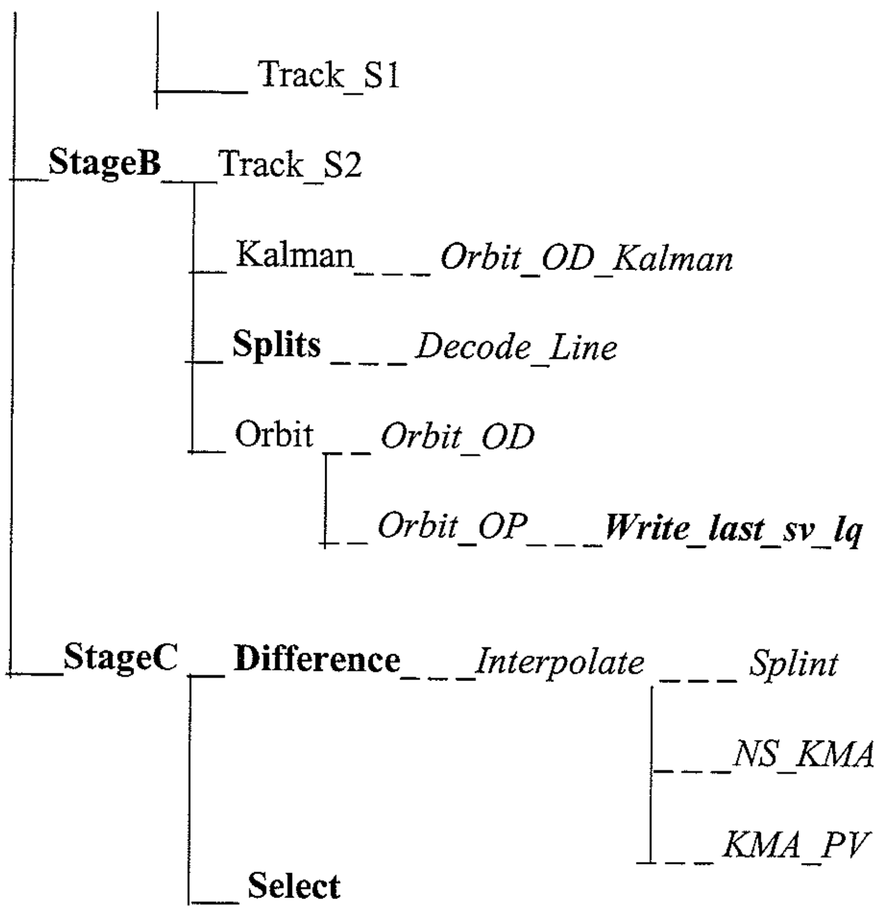
- *ORBIT* calls ORBIT_OP and ORBIT_OD subroutines.
- *TRACK_S0* calls GEN(GET)_SCHEDULE, DECODE_LINE, READ_STATION, ADD_STATIONS and WRITE_STATIONS subroutines.
- *INTERPOL* calls EXTRACT and GENEPH subroutines.
- *KALMAN* calls ORBIT_OD_KALMAN subroutine developed by Tomas Welter.
- *DIFFERENCE* calls INTERPOLATE which calls SPLINT, NS_KMA and KMA_PV subroutines.

Procedure, program and subroutines	Language of implementation
STAGE A, B, C	DCL
READER	Fortran-90
ORBIT	Ada
INTERPOL	Fortran-90
EXTRACT	Fortran-90
GENEPH	Fortran-90
TRACK_S0	Fortran-90
DECODE_LINE	Fortran-90
GEN_SCEDULE / GET_SCHEDULE	Fortran-90
ADD_STATIONS	Fortran-90
READ_STATION / WRITE_STATIONS	Fortran-90
TRACK_S1 / _S2	Fortran-90
SPLITS	Fortran-90
KALMAN	Ada
WRITE_LAST_SV_LQ	Ada
DIFFERENCE	Fortran-90
INTERPOLATE	Fortran-90
SELECT	Fortran-90

4.4.4. Call tree[13]

```
MCSIM______StageA____Reader___ Get_Word
            |        |       |
            |        |       |___ Get_Line
            |        |
            |        |__ Orbit ___ Orbit _OD
            |        |          |
            |        |          |___ Orbit_OP
            |        |
            |        |__ Interpol ___ Extract ___ Pv_Kma
            |        |             |           |
            |        |             |           |___ Kma_Ns
            |        |             |           |
            |        |             |           |___ Cal_Mjd
            |        |             |
            |        |             |___ Geneph ___ Spline
            |        |                          |
            |        |                          |___ Signal
            |        |                          |
            |        |                          |___ Pv_Kma
            |        |                          |
            |        |                          |___ Kma_Mns
            |        |
            |        |__Track_S0 __ Decode_line
            |        |           |
            |        |           |__ Select_Rec
            |        |           |
            |        |           |__ Get_Schedule _Decode_Line
            |        |           |
            |        |           |__ Gen_Schedule[14]_ Sort
            |        |           |
            |        |           |__Read_Stations[3]
            |        |           |
            |        |           |__ Add_Stations
            |        |           |
            |        |           |__ Write_Stations[3]
```

[13] New programs are in bold. Some old subroutines not essential here are not written.
[14] When no real data are available.



The three main stages constitute the first level, the programs READER, ORBIT, INTERPOL, TRACK_S0, TRACK_S1, TRACK_S2, KALMAN, DIFFERENCE and SELECT form the second level, and the subroutines are the third level of the DFD.

4.4.5. Environment

The *MCSIM* procedure is the master process. It is implemented in the DCL language of the AXP/VMS and is named MCSIM.COM. This process uses the different programs written in Fortran-90 or in Ada which themselves uses other subroutines and structure definitions of standard packages of SES : *PANLIB, GENERAL, GEOSLIB, TRACKLIB* and *SKSPEC* which are part of the Fortran-90 application, *ORBIT* and *KALMAN* which are part of the Ada application. In the following table, the corresponding package of the old programs and subroutines are summarised.

Existing program and subroutines	Package
ORBIT	Ada
DECODE_LINE	PANLIB
READ_STATION / WRITE_STATIONS	TRACKLIB
TRACK_S1 / _S2	Fortran-90
KALMAN	ORBIT
INTERPOLATE	SKSPEC

Chapter 5.

Statistical analysis.

The MCSIM simulator is a process which generates simulated observations corresponding to artificial trajectories including the effect of random perturbations, treats them by the LS and the EKF algorithms, and makes differences between the resulting estimations and the reference error free trajectory estimation. The results of the process are two files containing differences between the reference trajectory and estimations produced by both methods as a function of estimation epochs.

This simulator is run for each set of input parameters, *i.e.* each *Simulation case*. They are run to compare the LS and EKF methods, and are presented in the *Analysis plan* of **Chapter 6**. Furthermore, they are designed to characterise the behaviour of the EKF versus LS, and determine statistically if the decrease of the observational error due to the trilateration tracking system, which increases the relative importance of the unmodeled forces, allows to conclude that one methods is better suited for OD.

The outputs of the simulator are written in the specific form ***DIF_MC_'xxxyyz'.LQ*** for LS and ***DIF_MC_'xxxyyz'.KF*** for EKF, as previously described in the User Manual of **Appendix A3**, which links each simulation case with its result. These files contain eight groups of $N(=xxx)$ differences, that is the N realisations of the simulator process at each of the eight chosen epochs. To correctly analyse them, it is necessary to decide which quantity best discriminates the two methods.

The comparison is based here on the propagated position differences in the satellite axes, propagation being realised with the *Euler-Hill Algorithm* [Clo 60] from the estimation epochs coming from the estimator. This propagation of the differences in the satellite axes over one orbital period is done to avoid a bias which could happens due to the periodical variation of the local coordinates. The best method is thus the one which gives the differences nearest to zero, the eight estimation epochs being used to observe the convergence.

As the TRACK_S2 part of MCSIM generates random numbers to simulate the Gaussian measurement noise and models physical errors in a probabilistic way, estimated differences are random variables. The analysis of such quantities must therefore be statistical. Stage B and C of the process are then run several times to form samples of N realisations for each simulation case. They are statistically analysed in the post processor described in this chapter.

5.1. Statistical analysis of results

5.1.1. Summary statistics

To deal with a large number of data, it is necessary to pick out their important features in order to choose an appropriate model. One thus has first to reduce all data into some values called *summary statistics* and representing the data, and secondly to realise data analysis with these values [Spi 61].

The summary statistics consists of two groups. The first one is based on *moments, i.e.* mainly mean and standard deviation. The other one is based on the *order statistics*, that is principally the minimum, maximum and median. All these statistics can be represented using frequency tables and diagrams, box and whisker box, histograms, and scatter plots.

5.1.2. Elementary sampling theory

Sampling theory is a study of relationships existing between a population and samples drawn from it. This theory is used to estimate unknown population quantities, as mean and variance (also called population parameters or *parameters*), from sample quantities (also called sample parameters, or *statistics*).

In order to draw conclusions about the population from samples, these samples must be representative of the population. One way in which a representative sample may be obtained is by a *random sampling* process, according to which each member of a population has equal chance to be included in the sample.

Each of all possible samples of size N which can be obtained from a given population can be characterised by statistics which vary from sample to sample. This gives a distribution for each statistics, called *sampling distribution*[1] of the considered statistics. For each sampling distribution one can also compute the mean and standard deviation, *i.e.* the summary statistics of the sample statistics distributions.

Sampling distribution of mean

If one denotes by $\mu_{\bar{x}}$ and $\sigma_{\bar{x}}$ the mean and standard deviation of the sampling distribution of mean, and by μ and σ the population mean and standard deviation respectively, then

$$\mu_{\bar{x}} = \mu \qquad \sigma_{\bar{x}} = \frac{\sigma}{\sqrt{N}}\sqrt{\frac{N_p - N}{N_p - 1}} \approx \frac{\sigma}{\sqrt{N}} \tag{5.1}$$

[1] These sampling distribution may be computed for mean and standard deviation, but also for other statistics such as proportions, differences, sums, maximum, minimum, etc.

where N_p is the size of the population, considered very large in comparison to N. The sampling distribution of mean is very nearly normal for $N \geq 30$ even when the population is non-normal.

5.1.3. Central limit theorem

If $\{X_k\}$ are N independent and identically distributed (i.i.d.) realisations of the random variable X and if $E[X_i] = \mu$ and $Var[X_i] = \sigma$ exist, then for constant values a and b ($a < b$)

$$\lim_{N\to\infty} P\left[a < \frac{S_N - N\mu}{\sigma\sqrt{N}} < b\right] = F(b) - F(a) \qquad (5.2)$$

where $S_N = \sum_{i=1}^{N} X_i$, and F(x) is the normal distribution function N(0,1). [Ban 95]

This theorem states that a sum of independent random variables tends to be Gaussian when the number of variables in the sum is large. In practice, the sample mean $\overline{X_N}$ may therefore be considered as a normal random variable $N(\mu, \frac{\sigma}{\sqrt{N}})$

1. When $N > 30$, for any distribution of the random variable X in the population
2. For any value of *N*, if the population of *X* is normally distributed.

This theorem is the fundamental basis of many statistical analyses, like Monte Carlo Methods. Its meaning is very important when one deals with samples of random variables because it allows to consider that the sum of random variables is normal when there are enough realisations of them, and therefore helps in practical statistical analysis of stochastic systems.

5.1.4. Statistical estimation

Sampling theory deals with samples of a known population to obtain information about these samples. From a practical viewpoint, it is more often useful to find information about a population from samples of it. This is called *statistical inference.* One goal of this is to estimate population parameters (such as mean and standard deviation) from corresponding sample statistics.

One can estimate a parameter γ of the population from data of one sample in two manners : with a *point estimate* or with an *interval estimate.*

- The point estimate is a number which estimates the unknown parameter.
- The interval estimate is an interval containing the estimation of the unknown parameter with a certain accuracy level. It indicates the precision or accuracy of the estimate. One generally denotes the error or precision of an estimate by its *reliability.* A relation exists between the interval estimate number, *i.e.* the size of this interval, and the probability of finding the estimate inside it. As a consequence, it is often

called the *confidence interval.* There is a certain confidence of finding the estimate inside the interval, and this with a certain confidence level 1-α, where α is the error.

Note : The statistics is called an *unbiased estimator* when the mean of the sample distribution equals the corresponding population parameter. For example, $\overline{X}$ and $s_{NC}^2 = \frac{N}{N-1}s^2$ are unbiased estimates, since $E\left[\overline{X}\right] = \mu$ and $E\left[s_{NC}^2\right] = \sigma^2$. Moreover, if one considers all possible statistics whose sampling distribution have the same mean, the one with the smallest variance is called the most efficient or *best estimator* of this mean.

i. Confidence interval for mean

The confidence limits at the confidence level 1-α, for a normal population whose variance is known, are given by

$$CI_{1-\alpha}(\mu) = \overline{X} \pm \frac{\sigma}{\sqrt{N}} z_{1-\frac{\alpha}{2}}, \tag{5.3}$$

where $z_{1-\frac{\alpha}{2}}$ is the abscissa (also called *normal centile*) of the reduced normal centred variable of probability density $f(u)$, such that

$$\int_{-\infty}^{z_{1-\frac{\alpha}{2}}} f(u)du = 1 - \frac{\alpha}{2} \tag{5.4}$$

and where σ is the known standard deviation of the population.

When the standard deviation of the population is unknown, the confidence limits at the confidence level 1-α, for a normal population mean, are given by

$$CI_{1-\alpha}(\mu) = \overline{X} \pm \frac{s}{\sqrt{N}} t_{N-1,1-\frac{\alpha}{2}} \tag{5.5}$$

where s is the estimated value of σ, calculated with the sample, and $t_{N-1,1-\frac{\alpha}{2}}$ (also called Student centile) is the abscise of the Student-Fisher variable, with $N-1$ degrees of freedom, with a probability density $f_{t_{N-1}}(u)$, such that

$$\int_{-\infty}^{z_{1-\frac{\alpha}{2}}} f_{t_{N-1}}(u)du = 1 - \frac{\alpha}{2} \tag{5.6}$$

Furthermore, it is possible to relate the confidence interval to the needed number of measurements in the sample to have a certain level of precision. The risk is given by

$$\alpha = P\left(\left|\overline{X_N} - \mu\right| \geq z_{1-\frac{\alpha}{2}} \frac{\sigma}{\sqrt{N}}\right) \quad \text{or} \quad \alpha = P\left(\left|\overline{X_N} - \mu\right| \geq t_{N-1,1-\frac{\alpha}{2}} \frac{s}{\sqrt{N}}\right) \tag{5.7}$$

depending of the knowledge of the population's variance.

The corresponding number N is then given either by

$$N = \left(z_{1-\frac{\alpha}{2}} \right)^2 \frac{\sigma^2}{z_{1-\frac{\alpha}{2}} \frac{\sigma}{\sqrt{N}}} \quad \text{or} \quad N = \left(t_{N-1,1-\frac{\alpha}{2}} \right)^2 \frac{s^2}{t_{N-1,1-\frac{\alpha}{2}} \frac{s}{\sqrt{N}}} \tag{5.8}$$

where successive approximations need to be done.

Note : For $N \geq 30$, the asymptotic normality of mean is given by the *Central Limit Theorem*, and the results are still valid whether the population is normal or not. For $N \leq 30$, the approximation is not valid and small sampling theory must be employed.

ii. **Confidence interval for standard deviation**

The confidence limits for the standard deviation σ of a normally distributed population as estimated from sample with standard deviation s are given by

$$CI_{1-\alpha}(\sigma) = s \pm z_{1-\frac{\alpha}{2}} \sigma_s = s \pm z_{1-\frac{\alpha}{2}} \frac{\sigma}{\sqrt{2N}} \tag{5.9}$$

where σ is the population standard deviation supposed known.

For $N \geq 30$, one can consider that the variable s is normal. But for $N \leq 30$ this is not true anymore, and one has to consider the following confidence interval

$$CI_{1-\alpha}(\sigma) = \left[s - \sqrt{\frac{\sum \left(X_i - \overline{X} \right)}{b}}, s + \sqrt{\frac{\sum \left(X_i - \overline{X} \right)}{a}} \right] \tag{5.10}$$

where a and b are the values $a = \chi_{\frac{\alpha}{2}}^{\ 2}$ and $b = \chi_{1-\frac{\alpha}{2}}^{\ 2}$.

5.1.5. Statistical decision, test of hypotheses

The above statistical notions are often based on the normality of the population which is generally unknown. This requires a decision about population distribution based on sample information. Assumption about this population distribution may be true or not, and is called *statistical hypothesis*. In many instances, one formulates a statistical hypothesis for the sole purpose of rejecting or nullifying it, thus named *null hypothesis* and denoted H_0. The hypothesis which differs from H_0 is said *alternative* and denoted by H_1.

If the results observed in a random sample differ markedly from those expected under a particular hypothesis, the observed differences are said to be *significant* and the hypothesis is rejected. Such tests are called *tests of hypotheses* or *test of significance* or *rules of decision*. Thus when H_0 is supposed true, one can construct a confidence interval at the confidence level 1-α. If this interval contains the observed value, H_0 is accepted, if it does not, H_0 is rejected. This interval thus defines the critical region where the hypothesis is rejected.

Type I and II errors

Two kinds of errors are possible in a test of hypotheses, the Type I and Type II errors. The first one is the probability of rejecting an hypothesis H_0 when it should be accepted $\alpha = P(RH_0|H_0)$, that is the level of confidence. The second one is the probability of accepting the hypothesis H_0 when it should be rejected $\beta = P(AH_0|H_1)$. In both cases, a wrong decision has occurred. Any test of hypotheses must be designed to minimise errors of decision. This is generally difficult because α and β vary in opposite directions. The only way to reduce both errors is to increase the size of the sample.

Note : The level of confidence α is equal to the area, under the probability density curve, outside the confidence interval. The *power* of an hypotheses test is the probability of a correct reject of the hypothesis H_0, that is $P(RH_0|H_1) = 1 - \beta$. This power is greater and greater when the hypothesis H_0 is more and more false.

5.1.6. Adjustment tests, normality tests

Depending on the normality or non-normality of the population and on the knowledge or not of its variance, one edicts a rule of decision expressed in function of the confidence interval and statistics of the sample. This rule is used to determine some properties of the population, such as its mean. When the normality of the population is assumed, these rules are easier to formulate. It is thus necessary to first determine if the population is normal, and thereafter realise hypotheses tests.

The purpose of an *adjustment test* is to adjust an unknown theoretical distribution from the knowledge of an observed distribution (a sample). This step is sometimes necessary before realising an hypotheses test.

Two different situations occur

1. The theoretical distribution of the population is known, (mean and variance). The test can then be realised directly.
2. The theoretical distribution is not known, but is defined with some parameters. These parameters first have to be estimated with the sample data. There are two phases :
   - The adjustment, that is the search for the probability distribution best adjusted with the observed distribution.
   - The test itself, that is the comparison of this theoretical distribution and the observed distribution.

i. **The χ^2 adjustment test**

This test checks if a theoretical population is defined by a given probability distribution $(p_1, p_2, \dots p_k)$ when the observed absolute frequencies of a sample are $(n_1, n_2, \dots n_k)$, where k is the number of categories of the population and

$$N = \sum_{i=1}^{k} n_i = \sum_{i=1}^{k} Np_i \qquad (5.11)$$

is the size of the sample. It starts with calculating the value

$$\chi_{obs}^{\ 2} = \sum_{i=1}^{k} \frac{(n_i - Np_i)^2}{Np_i} \qquad (5.12)$$

which is a good measure of the difference between the theoretical and observed values. Each of the k frequencies observed n_i is then considered as the realisation of a binomial variable, asymptotically normal of mean Np_i. The hypotheses H_0 is that $\chi_{obs}^{\ 2}$ is a χ^2 random variable of $(k-1)$ degrees of freedom.

If $\chi_{obs}^{\ 2} = 0$, there is concordance between the theoretical and observed frequencies and H_0 is accepted, and if $\chi_{obs}^{\ 2} \geq \chi_{1-\alpha}^{\ 2}$ then H_0 is rejected at the level $1-\alpha$.

Note : This test is valid if $np_i \geq 5$, so it may be necessary to group classes to satisfy it, that is reduce the number of degrees of freedom υ of $\chi_{1-\alpha}^{\ 2}$. When the theoretical distribution is not completely defined, one must first estimate its parameters with the *χ^2 minimum method* [2] and computes the probabilities p_i. In that case, υ is reduced by the number m of estimated parameters $\nu = k - 1 - m$.

ii. **The Kolmogorov-Smirnov test**

Here the comparison is based on the cumulative function of frequencies

$$I_N = \frac{\left\| \{x_i : x_i \leq x, 1 \leq i \leq N\} \right\|}{N} \qquad (5.13)$$

of a sample $\{x_1, x_2, \dots x_N\}$ of size N, with the distribution function $F(x)$ of the population. The test is then based on the determination of the maximum difference

$$|I_N(x) - F(x)| \qquad (5.14)$$

and the comparison of this difference with particular critical values.

The I_N function is in formed of vertical steps at abscises x_i, and one generally defines two values which are random values following the *Miller-Owen distribution*[3]:

[2] This method consists in minimising the quantity $\chi_{obs}^{\ 2}$.

[3] This distribution is such that $\lim_{n\to\infty} P\left(\sqrt{N} D_N^{\ +} \leq d\right) = 1 - e^{-2d^2}$, $0 \leq d \leq \infty$.

$$D_N^{+} = \max_{1\le i\le N}\left[\frac{i}{N} - F(x_i)\right] \text{ and } D_N^{-} = \max_{1\le i\le N}\left[F(x_i) - \frac{i-1}{N}\right] \qquad \textbf{(5.15)}$$

If one chooses a level confidence $1-\alpha$, the value $d_{N,\alpha}$ is such that $P(D_N > d_{N,\alpha}) = \alpha$ and the test consist in rejecting H_0 at the level α if the observed value D_N is greater than $d_{N,\alpha}$.

Note : $d_{N,\alpha} \approx \sqrt{-\frac{1}{2N}\ln\frac{\alpha}{2}}$ when N is large enough (*i.e.* $N > 35$). The reject condition therefore becomes $Z_N = \sqrt{N}D_N \ge \sqrt{-\frac{1}{2}\ln\frac{\alpha}{2}}$.

iii. The normality test

To test the normality of a population which is observed through a sample, the two explained methods may be used under some restrictions :

- The χ^2 test may be used when $N > 30$ for a number of observation ≥ 5 in each class.
- The Kolmogorov-Smirnov test may be used $\forall N$. Nevertheless, the Kolmogorov-Smirnov test may only be used for continuous distributions.

The Kolmogorov-Smirnov test is therefore more general, when the distribution is continuous, because the χ^2 test requires 5 realisations in each classes and is not useful for small samples.

When the normality can be assumed, one directly realises tests of hypotheses on mean and variance of the population. If it cannot be assumed from the test of normality, the Central Limit Theorem still gives the possibility to deals with sample of $N > 30$ realisations and realise test of hypotheses on the mean. Nevertheless, tests on population variance are difficult to realise in that case due to the great sensibility of such test of comparison to population normality.

5.1.7. Monte Carlo Methods

When a deterministic system is submitted to random variations (coming for example from external perturbations) it is possible to estimate the statistical distribution of its response through a technique involving stochastic simulations. From N realisations of the probabilistic system, throughout random variations of its parameters, its possible to simulate the real operational mode of the system, and then produce N responses which form a sample of the possible responses (finite or infinite population). Thanks to the Central Limit Theorem, the mean of this sample is a random variable normally distributed.

Monte Carlo Simulations (MCS) is the general designation for such stochastic simulations of a probabilistic system. It means the use of random number and a statistical analysis of results with the Central Limit Theorem implications.

The statistical analysis of the realisations is based on an estimation of the system parameters (mean, variance, along with their confidence intervals if possible). The confidence intervals depending on the level of confidence, the analysis consists in choosing a level of confidence and running the process a number N of times great enough to reach the corresponding confidence interval for the system parameter estimations, the precision of results increasing as $\sqrt{N}$. The process is thus stopped when there are enough realisations in the sample to reach the level of confidence in results. In comparison with analytical methods, where no stochastic simulations are realised, the MCS has the great advantage to provide the shape of the unknown distribution.

5.2. The analysis post process

In this work, MCS are run for each simulation case. It is necessary to run the simulator presented in **Chapter 4**, and treat its outputs in a post process, which is described in this chapter. Due to randomness in the simulator, the N realisations of each sample created by the simulator may be considered as statistically *independent and identically distributed* (i.i.d.), and the different statistical notions presented before may be applied to realise the statistical analysis of the results in this post process, which thus realise the second part of MCS.

The two processes, simulator and post processor, are run for each simulation case, the corresponding sets of parameters being related to the indexes 'YYZ' used in the MCSIM result files and spanning all different interesting cases needed to compare the two methods efficiently, and the number of realisation being 'XXX', as explained in the User Manual of **Annexe A3**.

5.2.1. Analysis of the MCSIM results

The post processor performs statistical operations on the propagated differences to obtain significant final results and compare the two estimation methods. It also deals with corresponding Keplerian elements to provide another view of these results.

i. Monte Carlo Simulations

Each simulation includes several steps involving the simulator and the post processor, and provides four groups of results for the two methods LS and EKF.

Simulator

N runs (here 50) of the simulator generate two samples composed of i.i.d. realisations, for LS and EKF.

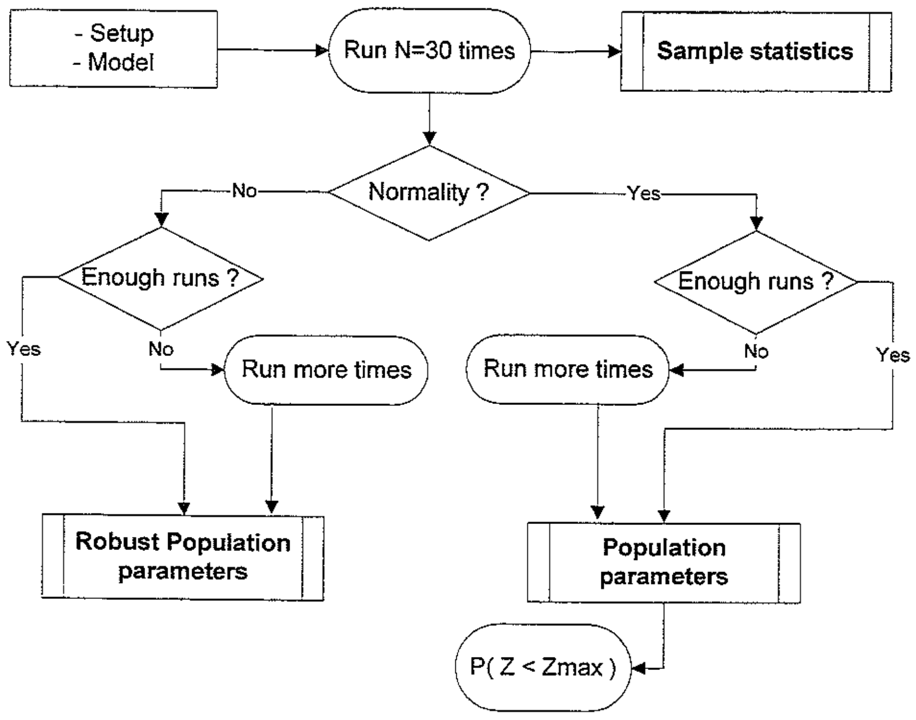
Post processor

- First, the input differences of the samples are converted in the satellite local axes and then propagated over one orbital period, for each hours of the day , with the Euler-Hill algorithm in the program *MAXP*, as explained at the beginning of this chapter. The maximum and minimum of these propagated differences along each axis form two samples of 2N realisations, each one containing the maximum and minimum of the 24 differences.

- Secondly, the input differences are transformed into Keplerian elements to have another representation of the results.
- Summary statistics are computed from the two samples, minimum and maximum of Cartesian and Keplerian propagated differences are then obtained. Their graphical representations are finally drawn.
- A Kolmogorov-Smirnov normality test is realised to test normality of populations.
- Mean and standard deviation of the populations with confidence intervals of mean are computed for Cartesian differences, the confidence intervals being determined with the Student centile because the population variances are unknown. Normality is not needed here to compute the confidence limits, because N is greater or equal to 30 and the Central Limit Theorem is thus applicable[4].
- The number of supplementary required runs is then computed, the population parameters estimations being finally computed and stored.
- If populations are both normal, the comparison probabilities for the two methods are computed using the final population parameter estimations

ii. Flow chart

This flow chart includes the simulator and the post processor. It shows how they are used together and how the comparison results are obtained.



[4] Parameter estimations realised in that case are called *robust estimations* [Dag 92]

iii. Results

The results for each simulation case are divided in four groups :

1. The min. and max. propagated differences are characterised by their **mean** and **standard deviation** at the eight epochs, and for the three position components in the local frame. These summary statistics are computed by the program *STATISTICS* and the application *EXCEL©*. Their **graphical representations** are then realised with *EXCEL©*.
2. The Keplerian differences are characterised by their **frequency distribution** at the eight epochs, and for the three position components. These statistics are computed by the program *STATISTICS* and the application *EXCEL©* which also draws their **histograms**.
3. The program *ANALYSIS* then tests normality of the population using the **Kolmogorov-Smirnov Test**. It also computes, in parallel with *EXCEL©*, the population parameters with **confidence interval for mean**[5], and gives the **number of runs** needed to reach the confidence interval corresponding to a constant level of confidence.
4. If normality can be assumed, *ANALYSIS* computes the normal **probability** that the global random variable is lower than a constant difference[6].

5.2.2. MCSTAT design and description

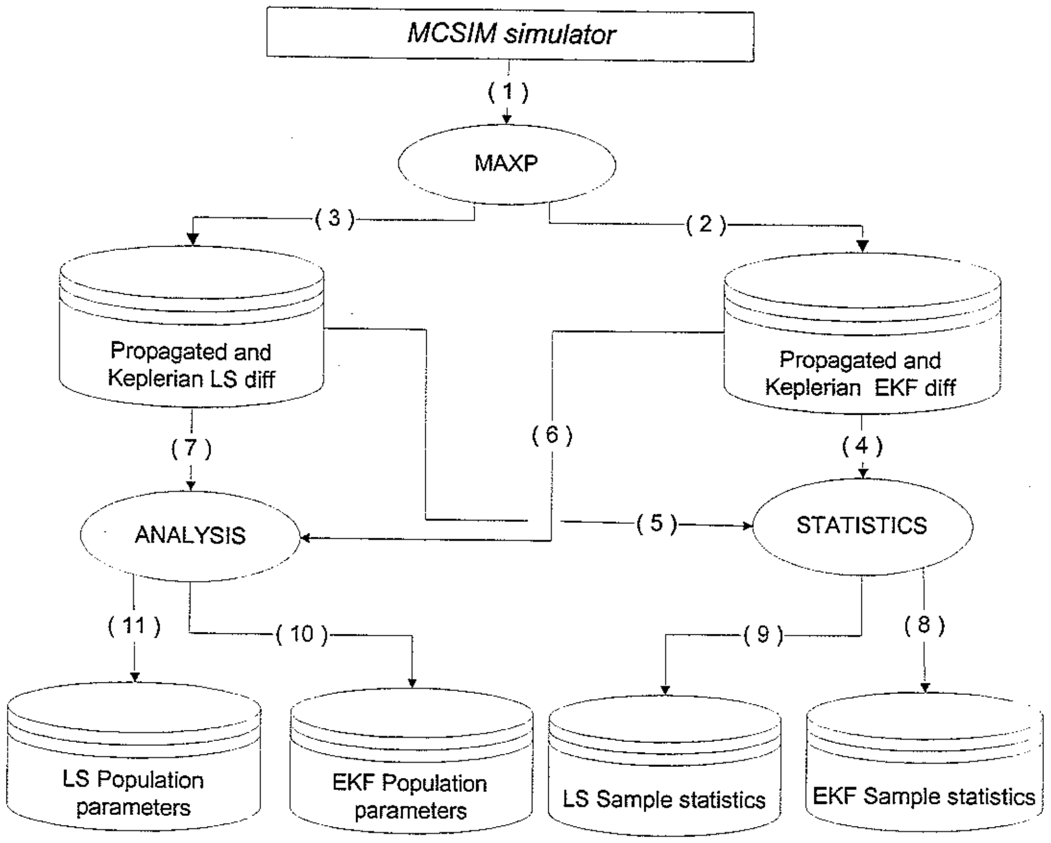
The three Fortran-90 programs used in the post processor are MAXP, STATISTICS and ANALYSIS. They constitute the application *MCSTAT* which runs on the AXP/VMS system as the simulator MCSIM. On the other hand the application *EXCEL©* is runs on PC simultaneously with this post processor.

The design and description of the MCSTAT part is here presented in a similar way than MCSIM, while the use of EXCEL© is briefly presented thereafter. Theses two applications are complementary, and allow a systematic analysis of the simulator results.

i. Data Flow Diagram

The post processor is presented in a one level DFD with flows between the simulator and the F90 programs, with its hierarchical description in **Annexe A8**. The inputs of this application are the outputs of the process MCSIM, *i.e.* the state vector differences versus eight estimation epochs. The outputs are sample statistics and population parameters for propagated positions.

[5] The parameter σ is not studied as far as μ, because it is very sensitive to normality.
[6] That is the position module lower than 100m.



ii. **Data Dictionary**

Flows

N°	Name	Type	Using
1	DIF_MC_'SIMUL'.LQ / DIF_ MC_'SIMUL'.KF	MJD : REAL*8 VECTOR 6D : REAL*8	Contains LS and EKF differences and corresponding times.
4 and 5 / 6 and 7	DIF_MAXP_LQ and _KF / DIF_EQ_LQ and _KF	MJD : REAL*8 VECTOR 6D : REAL*8	Contain LS and EKF 24 propagated differences / equinoctial differences, and corresponding times, for the N realisations.
8 / 9	STAT_ , FREQ_ , PLOT _ KF / LQ	See types definitions and programs descriptions.	Statistics of the LS and EKF differences contained in samples.
10 and 11	POP_ , PROBA_ MAXP KF / LQ	See types definitions and programs descriptions.	Values created to decide whether population are normal (1) or not (0) and probabilities values (if normal).

Processes

Programs and subroutines	Description and using
MAXP	• This program use the Euler-Hill algorithm to propagate the differences over two days, and the subroutine ECI_EHF to translate them in the satellite axes. It then construct a sample containing 24 propagated state vector for each of the N realisations of the input sample. • This program also uses the subroutine PV_KMA to construct a sample containing the N realisations in equinoctial elements.

STATISTICS	This program treat the propagated difference LS and KF as samples of N realisations. It runs the subroutines G01ABF,G01AEF and G01ASF to produce files containing statistics of the two samples for the eight epochs.
ANALYSIS	This program test normality of the population and compute its mean, variance and confidence interval of mean. It determine if there are enough run for this confidence intervals to be small enough for a constant confidence level. If population is normal, it also computes the probability value that the variable is lower than a constant difference.
G01 subroutines : **-G01ABF** **- G01AEF** **- G01ASF**	This group of NAG FORTRAN [Nag 90] subroutines compute simple calculations on statistical data : plots and descriptive statistics, statistical distributions, deviations from distributions, etc. • The subroutines G01ABF computes means, standard deviation, correct sums of squares and products, minimum and maximum values, and the product moment correlation coefficient of both LS and EKF sample. • The subroutine G01AEF constructs a frequency distribution with calculated class boundary values for LS sample and for EKF sample. • The subroutine G01ASF produces boxes and whisker plots for the 8 epochs.
G08 subroutines : **G08CBF**	This group of NAG FORTRAN [Nag 90]subroutines compute Nonparametric Statistics, *i.e.* location tests, dispersion tests, tests of fit, association and correlation tests, tests of randomness, etc. In this process, the subroutines - The subroutine G08CBF realises here the Kolmogorov-Smirnov tests of fit.
ECI_EHF	Given the inertial coordinates of a reference s/c, *i.e.* the interpolated vector here, convert the position and velocity components of a reference object to local coordinates. This subroutine is used to intriduce correct state vector in the Euler-Hill subroutine.
EULER_HILL	Analytical propagation of the relative state vector over the time span delta using the Euler-Hill equations. This subroutine is then used to propagate the position over a time interval and obtain a maximum position.
PV_KMA	This subroutine transforms differences coming from the simulator into equinoctial elements.

iii. Command Procedure Description

The procedure implementation is in agreement with the following Call Tree, and is processed by the command file MCSTAT written in the DCL language. It contains the different programs and subroutines written in Fortran-90 presented above, and link them following the DFD.

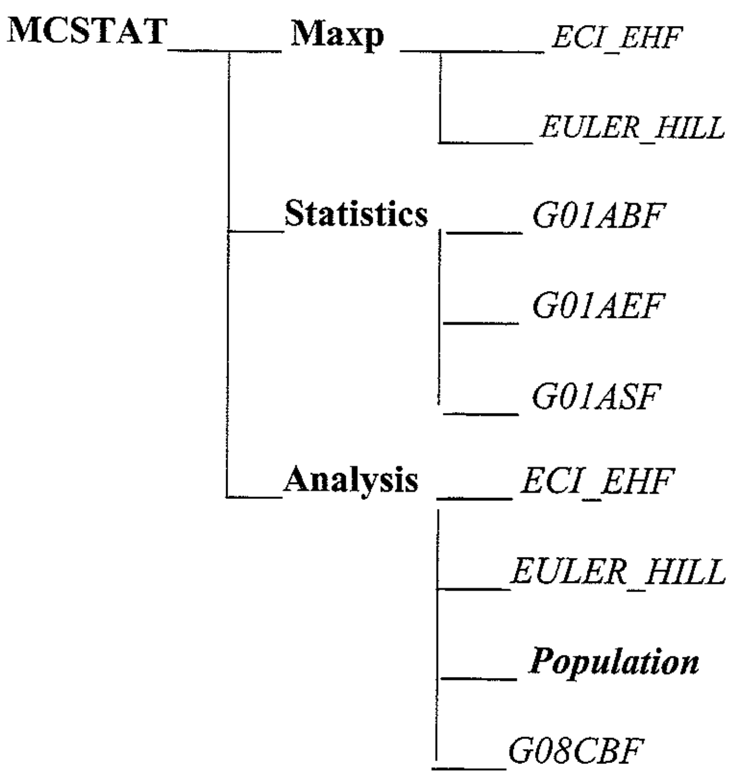
The MCSTAT procedure is composed of the programs MAXP, STATISTICS and ANALYSIS. These programs call the following subroutines :

- *MAXP* calls ECI_EHF and EULER_HILL subroutines.
- *STATISTICS* calls G01ABF, G01AEF, G01ASF and G01FAF subroutines.
- *ANALYSIS* calls PARAMETER and G08CBF subroutines.

In the following table the different languages of implementation are presented for programs and subroutines used in the MCSTAT post processor.

Procedure, program and subroutines	Language of implementation
MCSTAT	DCL
MAXP	Fortran-90
ECI_EHF	Fortran-90
EULER_HILL	Fortran-90
STATISTICS	Fortran-90
ANALYSIS	Fortran-90
NAG G01 and G08 subroutines	Fortran-90
POPULATION	Fortran-90

iv. **Call Tree**



v. **Environment**

The MCSTAT procedure is implemented in the DCL language of the AXP/VMS and is named MCSTAT.COM. This process uses the Fortran-90 programs presented before, which themselves uses other subroutines part of the NAG Fortran-77 routines package [Nag 90], and of the PANLIB package.

Existing program and subroutines	Package
G01-G07-G08	NAG Fortran-77
ECI_EHF	PANLIB
EULER_HILL	PANLIB

5.2.3. EXCEL© application description

This application is used together with the procedure MCSTAT to draw statistics of the different samples coming from simulator, with parameters of the corresponding populations and graphic representations.

i. Data treatment

Several EXCEL© sheets contain the input data and the computed values, and more specifically

1. The input data DIF_MAXP_'xxxyyz' and DIF_EQ_'xxxyyz' for LS and EKF, coming from program MAXP of MCSTAT.
2. Mean, standard deviation, maximum, minimum and mean confidence intervals of the two sets of input data for the eight epochs and the two methods.
3. Histograms and frequency tables built from the Keplerian differences for the two methods at the last of the eight epochs.
4. Graphical representations of mean, standard deviation and confidence intervals evolution over the eight tracking intervals, for the three components and for the two methods.

ii. Data representation

EXCEL© is used in parallel with the MCSTAT post processor to treat in a more visual way the data coming from the simulator and being passed into the MAXP program of this post processor. This application indeed allows easy graphical representations of results.

In addition to the above described data treatment, two concepts of data representations are used from the basis information contained in mean, standard deviation and confidence interval results to draw graphics

The first concept of *global evolution* represents on the same figure the mean, confidence interval, and 3-sigma curves of the propagated differences (the three position components) versus the tracking time intervals (*i.e.* 3, 6, 9, 12, 18, 24, 36, 48 hours). The evolution over time of these random variables is thus statistically entirely characterised by these figures, the mean and its confidence interval giving the interval where there is a probability of $\alpha\%$ that the real variable is outside of it, and the 3-sigma interval where there are 99.95 % of chance to find the real variable.

Figure 5.1 shows an example of this diagram for one of the three components in the satellite axes, where the horizontal axis has hours units and the vertical axis has meters units. Moreover both vertical scales are the same for comparison of EKF and LS errors.

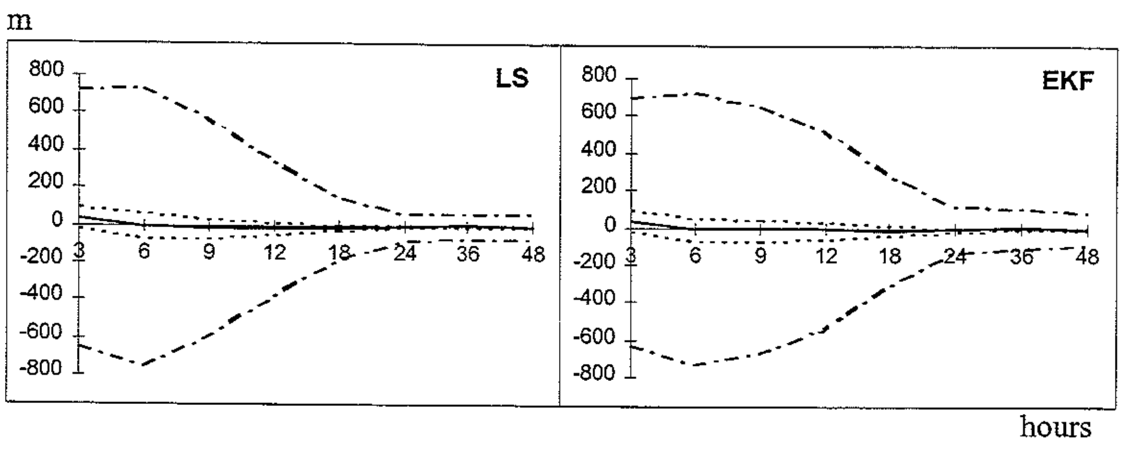


Figure 5.1

The second concept of *worst case* represents the addition of the 3-sigma band to the confidence interval. The corresponding curve is generally presented for the three components in the satellite axes as shown on Figure 5.2 for example, where the horizontal axis is scaled in tracking intervals of 3, 6, 9, 12, 18, 24, 36, 48 hours and the vertical axis has meter units and represent the evolution over time of variables representing the worst possible case of estimation, see equation (**5.16**) where s is the estimation of the standard deviation σ. These diagrams are used to compare the different simulation case results, so their vertical scale are again the same for the corresponding components of cases detailed in each paragraph of **Chapter 6.**

$$\overline{X} \pm \frac{s}{\sqrt{N}} t_{N-1,1-\frac{\alpha}{2}} + 3.s \qquad (5.16)$$

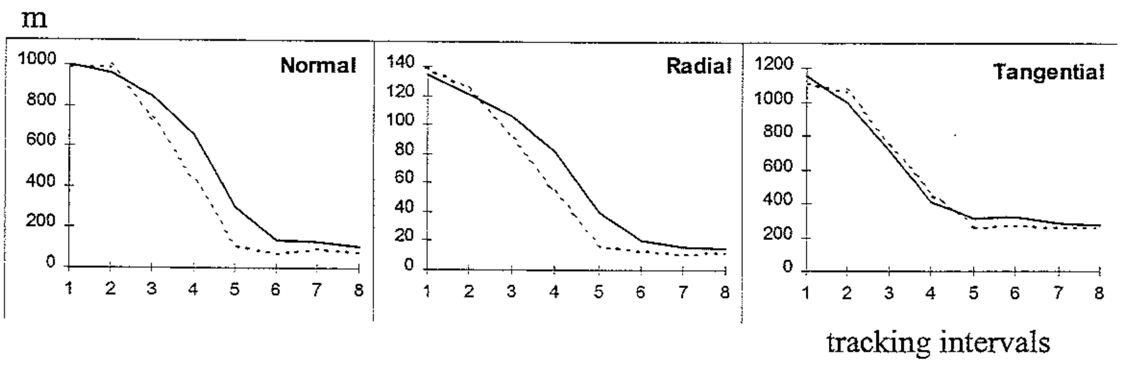


Figure 5.2

Note :

On worst case graphics like Figure 5.2, the horizontal axis is labelled with the numbers 1 to 8 instead of the 3, 6, 9, 12, 18, 24, 36, 48 hours. These numbers represent the eight tracking intervals corresponding to these eight hours.

Chapter 6.

Monte Carlo Simulations

The simulations realised to compare the LS and EKF are presented in this chapter through a *Simulation Plan* which is detailed in **Annexe A7**. For all cases of this plan, the simulator and the post processor described in **Chapters 5** have been run. Conclusions and interpretations of the results are detailed in **Chapter** 7.

During simulations it nevertheless appeared that these *main simulations* needed to be completed by *further investigations*, and that normality cannot be assumed in general. As a consequence, the main simulations are presented with their statistical results in paragraph 6.3, while the supplementary cases are given with their results in paragraph 6.4. Furthermore, as the simulator was designed to run with or without real data, and that real data were available before simulations were started, more cases than originally expected in the simulation plan have been realised with real data[1].

As it is not possible to detail all simulation cases and all their statistical properties, the results are presented through the most important cases. They are given for propagated differences in Cartesian and/or Keplerian elements throughout four groups of outcomes

- **Normality test** results for different simulation cases.
- **Histograms** of samples at the last epochs for Keplerian components of some simulation cases.
- **Population parameters** like mean, standard deviation, confidence intervals of mean at the eight epochs and for the three propagated position components.
- **Graphical representations** of these parameters through a worst case and a global *form*.

Other results, like normal probability that the differences are lower than a set value and variance analysis, cannot be presented because the normality for the populations could not be assumed. The criterion of comparison is thus based principally on populations parameters computed with the Student's distribution (as the variances of populations are unknown) and their graphical representations.

Note : The number of needed runs is different for each simulation case, so it has been fixed to 50, greater than all to guarantee the confidence limits.

[1] These real data are used to create the reference trajectory, as explained in Chapter 4.

6.1. Analysis plan

The simulator process and the statistical analysis post processor are first run in two ways, correspondingly to the two kinds of error which influence the orbit estimation :

1. *The measurement errors* in the observation model, which always exists in real tracking data.
2. *The dynamic errors* in the dynamic model, to model inaccuracies in the physical values.

These two kinds of simulations respectively correspond to a set-up as best as possible with a representation set of error models, and a null error model with different set-up files. The parameters of both files are inputs of the simulator, their variations thus giving the different simulation cases to be tested. However, it is of course not possible to span the two parameters spaces. So, simulations are limited to a relatively small number of cases[2]. Nevertheless, it would be difficult to choose which tests to realise if simulations were only based on these inputs. The *simulation plan* presented hereafter is therefore built on these two kinds of errors in a way allowing to test the different possibilities and behaviours of the EKF relatively to the LS method. The simulation plan is constructed in relation to the different set-up and error model files with their corresponding indexes to the tested cases and the number of runs realised as presented in **Annexe A6.**

Before presenting this simulations plan, both observation and dynamic errors are described as they are implemented in the TDS and the ORBIT software with the order of variation of their parameters.

6.1.1. Analysis with measurement errors

Here we consider range measurements from the trilateration tracking system with their measurement perturbations. The error model parameters are presented here and numbered from 1 to 9, to ease the relation between results and corresponding error model.

There are two sources of measurement errors : *random noise* and *physical effects* which are not modelled in the ORBIT OD software.

- ***The random noise errors*** represent all effects that are not taken into account in usual model. They are introduced either as simple *white noise* or by *fitting range residuals* with superimposed white noise. The former is produced by TRACK_S2 , while the latter is realised by the TRACK_S1 and is added to white noise by TRACK_S2 which again produces it. Both errors are produced for each tracking data realisation and each measurement time independently.
- ***The unmodeled physical effects*** represent different random errors that are generated and superimposed to the random noise by TRACK_S2, in a conservative approach, following the physical information given by the user in the error model file.

[2] Each run last more than 3', so a simulation of 50 runs needs 150'.

All realisations of the process are statistically independent, and the random values of the model parameters describing these effects are kept constant throughout the tracking interval for each realisation.

Note : In this section, the unmodeled physical effects and their implementation in the program TRACK_S2 through the error model file are particularly considered, the random additive noise being constant in the Ref_Stations file[3] when no real data are available and given in the Stations file by the program TRACK_S1 from real data in the other case.

In addition to these two error components, it is interesting to test different combinations of stations. The four stations of the SES trilateration system have been chosen here and are presented in Table 6.1. Nevertheless, in order to reduce the number of simulations and because it is not necessary to try all combinations, only two combination cases have been tested, four stations and two stations systems. On the other hand, simulations using real data have been done with all four stations

Station	**Index**	**Status**
Betzdorf	8000	master
Seville	8001	slave
Bergen	8002	slave
Rome	8003	slave

Table 6.1

At last, the Schedule File may also be changed to test different combinations of schedule measurements. To again restrict the number of simulations, only three cases have been be considered : regular measurement intervals, measurements concentrated at the beginning of the tracking interval, and measurements concentrated at its end.

Note : The different model files are numbered from 1 to 9 in the simulation plan, nevertheless they correspond to only two different model files : a null error model, and a complete one with reasonable values. The indexes distinguish the different combinations of both models with stations combinations, random noise possibilities (white or fitted) and schedules.

i. Measurement errors model parameters

The model file contains different values needed in Track_S2 to simulate physical effects which are unmodeled in the ORBIT software of SES. These effect are specified through a mean and standard deviation in order to realise random errors in TRACK_S2.

The following measurement error sources may be considered among others to improve the measurement model by adding different physical errors to measurements

[3] This random error is a white noise, or a fitted noise of one station measurements during some days.

- The *tropospheric refraction* effect.
- The *ionospheric refraction* with quiet or intense solar activity, and daily effects.
- A *short term noise* effect.
- The *station positioning errors and calibrations* effects.
- The *spacecraft delays* effect.

Nevertheless tropospheric refraction is generally well defined in measurement models as a known bias, while the ionospheric refraction is not. It is thus more interesting to only consider the latter. In the same way, station positions are generally very well known but their calibration and the spacecraft delays are not. In the TRACK_S2 error model, only *ionospheric refraction, station calibrations* and *spacecraft delays* thus have been considered.

The TRACK_S2 error model, see [Fra 96] and [Wau 95], uses seven couples of parameters, three for the ionospheric refraction, and four corresponding to the sum of the station calibration and the spacecraft delay effects. The global measurement error is given by

$$\delta = C_1 + \left[C_2 \Delta t + C_3 \sin(2\pi\Delta t - C_4)\right] + \left[C_{5\ or\ 6} \sin(2\pi\Delta t - C_7)\right] \qquad \textbf{(6.1)}$$

where C_i $(i = 1,...7)$ are the seven random values computed from the corresponding couples of the error model file, Δt is the time interval between measurements.
Note : C_5 is used when $(2\pi\Delta t - C_7) < \pi$, and C_6 when $(2\pi\Delta t - C_7) \geq \pi$.

In what follows, these three measurement errors are exposed with their model and parameters. Before running the process, the user introduces the seven couples of mean and standard deviation (m_i and σ_i, i = 1, 7) corresponding to the C_i in the model file. From them, Track_S2 generate the Gaussian random value C_i as

$$C_i = m_i + random(\sigma_i) \qquad i = 1,...7 \qquad \textbf{(6.2)}$$

- *Ionospheric refraction*

The ionospheric effect is specified for each station individually as

$$error^k{}_{ion} = \frac{\left\{\left|G(m2^k{}_{ion}, s2^k{}_{ion})\right| + \left|G(m1^k{}_{ion}, s1^k{}_{ion})\right|.\sin\left[\Psi(t_k) - G\left(m3^k{}_{ion}, s3^k{}_{ion}\right)\right]\right\}}{\sin(el_k)}$$

$$= \left[C^k{}_5 \sin\left(2\pi\Delta t - C^k{}_7\right)\right] \qquad \textbf{(6.3)}$$

when $\Psi(t_k) - G(m3^k{}_{ion}, s3^k{}_{ion}) < Pl(day_{time})$, or

$$error^k{}_{ion} = \frac{\left\{\left|G(m2^k{}_{ion}, s2^k{}_{ion})\right| + \left|G(m2^k{}_{ion}, s2^k{}_{ion})\right|.\sin\left[\Psi(t_k) - G\left(m3^k{}_{ion}, s3^k{}_{ion}\right)\right]\right\}}{\sin(el_k)}$$

$$= \left[C^k{}_6 \sin(2\pi\Delta t - C^k{}_7)\right] \qquad \textbf{(6.4)}$$

if $\Psi(t_k) - G(m3^k{}_{ion}, s3^k{}_{ion}) \geq Pl(night_{time})$, where t_k is the local time of the k^{th} station, Ψ is the Earth's angular rate, $m1^k{}_{ion}\,(m2^k{}_{ion})$ and $s1^k{}_{ion}\,(s2^k{}_{ion})$ are the means and standard deviations associated to the day (night) oscillation amplitude of the k^{th} station, $m3^k{}_{ion}$ and $s3^k{}_{ion}$ characterise the phase of this oscillation of the k^{th} station, the term $|G(m2\_ion, s2\_ion)|$ is introduced to prevent the refraction correction to become negative during night time, el_k is the elevation of the k^{th} station, and both Pl are constants representing the phase for the beginning and end of the day.

Model parameter	**Mean range**	**Standard deviation range**
$\frac{1}{\sin(el_i)}\left\|G(m1_{ion}, s1_{ion})\right\| = C_5$	0 - 1 m	0 - 10 m
$\frac{1}{\sin(el_i)}\left\|G(m2_{ion}, s2_{ion})\right\| = C_6$	0 - 0.5 m	0 - 5 m
$\frac{1}{\sin(el_i)}\left\|G(m3_{ion}, s3_{ion})\right\| = C_7$	0 - 3.14 rad	0 - 1 rad

Table 6.2

On Table 6.2, the reasonable mean and standard deviation values are presented.

- *Stations calibration*

The uncertainty on each station's ranging path calibration is assumed to result in a range error characterised by the Gaussian distribution

$$G(m_{stacal}, s_{stacal}) = C_1 \tag{6.5}$$

of constant mean and standard deviation[4] whose limits are presented in Table 6.3.

Model parameter	**Mean range**	**Standard deviation range**
$G(m_{stacal}, s_{stacal}) = C_1$	0 - 1 m	0 - 10 m

Table 6.3

- *Spacecraft delays*

The uncertainty on the spacecraft delays is assumed to result in a range error characterised by the following model and acceptable values of Table 6.4.

$$\begin{aligned} error_{delay} &= G(m0_{delay}, s0_{delay}).t + G(m1_{delay}, s1_{delay}).\sin\left[Psi.t - G(m2_{delay}, s2_{delay})\right] \\ &= C_2\Delta t + \left[C_3 \sin(2\pi\Delta t - C_4)\right] \end{aligned} \tag{6.6}$$

Model parameter	**Mean range**	**Standard deviation range**
$G(m0_{delay}, s0_{delay}) = C_2$	0 - 1 m/hour	0 - 10 m/hour
$G(m1_{delay}, s1_{delay}) = C_3$	0 - 1 m	0 - 10 m
$G(m2_{delay}, s2_{delay}) = C_4$	0 - 3.14 rad	0 - 1 rad

Table 6.4

[4] It is assumed that regular station calibration guarantee the absence of any periodic effect.

ii. Schedule parameters

The schedule file contains information needed in TRACK_S0, when no real data are available, to recreate the lacking stations file from the Ref_Stations File. This last file contains fitted residual data of one station measurements for one satellite OD performed over several days and is produced before these simulations by TRACK_S1 from Output file of Orbit estimation with real data. TRACK_S0 then produces a fit of residuals for the number of stations and at measurement time intervals specified in the Schedule file.

This last file is thus composed of different reproductions of a pattern, for each desired station (here 2 or 4 stations), with the *initial epoch* of measurements, the *period* of measurements (supposed the same for the different stations) and the *number of stations* considered at the beginning of the file. The pattern is composed of four informations

- The *station identity*.
- The *offset* (in seconds), which represents the beginning epoch of the first measurement interval from the initial epoch.
- The *number of* measurement *points* during each measurement period.
- The *cycle duration*, which represents the time interval during which measurements are collected by the stations.

station id	offset	3	150 s

Table 6.5

Here, the period chosen is 3600 seconds, the number of measurement points is 3, and the cycle duration is 150 seconds for all stations. The different schedules therefore change in their offset in order to concentrate measurements at the beginning or the end of the tracking interval. The pattern is presented in Table 6.5 and the beginning information in Table 6.6.

initial epoch (MJD)	3600 s	number of stations

Table 6.6

iii. Random noise errors

This error component is introduced in the process in Stations file when real data are available, or in the Ref_Stations file when no real data are avilable. So the additive noise is either *white* or *fitted* when no real data are available, and necessarily fitted in the other case.

- The white noise is created with zero mean and constant standard deviations for the different epochs of the tracking interval, in the Ref_Stations file.
- The fitted noise represents a fitting of the residuals coming from the LS treatment of real data (before running the process and independent of it). When no real data is available, the Ref_Stations file contains fitted means and standard deviations at successive epochs, and replace of the lacking Stations file.

6.1.2. Analysis with dynamic errors

Dynamic error means that the dynamic model is not perfect and contains a limited modelling of the realistic system. The more precise the dynamic model, the smaller are the dynamic errors. To represent such errors, two groups of dynamic parameters are used and spanned in the set-up file.

- The number of *terms in the geopotential development.*
- The *solar radiation pressure* coefficients.

In the present simulations, for simulations cases realised to observe the effect of propagation (dynamic) noise, the set-up file of the reference trajectory specifies a more accurate dynamic model than the one of LS and EKF estimations of the corresponding artificial tracking data.

Nevertheless for all other simulations, where the behaviour of LS and EKF methods is observed, both set-ups are the same and the dynamic errors come from parameters of methods which influence the estimations

- The *process noise* of the EKF representing the unmodeled forces
- The *initial state vector* and *initial guess error* (initial covariance matrix) for both methods.

Before running the process, the user introduces the dynamic parameters values in the Ref_Setup (reference trajectory) and Std_Setup (EKF and LS estimations) files.

i. Dynamic model parameters

- *Earth potential development*

As presented in Chapter 2, the first and most important force perturbing the orbit is due to the anisotropy of the Earth and is computed from relation (**2.12**).

In the set-ups, the number of geopotential coefficient taken into account may be introduced. The more coefficient taken, the best accurate is the model. It is thus interesting to realise simulations with varying number of geopotential coefficients in the limits given in Table 6.7.

Model parameter	**Possible value**
Potential_order	1 - 35

Table 6.7

- *Solar radiation pressure*

Two different models may be selected in the set-up file to take into account the solar radiation pressure in the force computation, from relations (**2.19**) to (**2.21**).

In the set-up, the two parameters C_R and C_{R_p} may be changed. It is also possible to modify the mass of the satellite and the surface (in m^2) exposed to Sun, with $\delta = 0$ or not. Nevertheless as all these parameters appear in the different relations through the global form $\frac{S}{m}C_R$ (or $\frac{S}{m}C_{R_p}$), their variations may be reduced to the only variation of C_R (or C_{R_p}). So, in simulations, only these parameters are modified in the range presented in Table 6.8.

Model parameter	Reasonable value
C_R	1.2 - 1.6 (min 0 - max 2)
C_{R_p}	0.01 - 0.1

Table 6.8

ii. **Parameters of methods**

- *Initial guess error*

The two initial state vector and the initial state variance[5] parameters influence the results of estimations for both methods. The initial variance is composed of the six position and velocity values in diagonal matrix, and the initial guess error is composed of errors on the six state vector components whose limit are presented in Table 6.9.

Model parameter	Reasonable values
Initial state variance (standard deviation)[6]	10000 - 10000 - 10000 (m) 0.1 - 0.1 - 0.1 (m/s)
Initial guess error	1000 - 1000 - 1000 (m) 0.01 - 0.01 - 0.01 (m/s)

Table 6.9

Note : The default value is ∞ for the six diagonal elements, and the others by default, the program Orbit begins its estimation with an infinite uncertainty on the initial state vector. A null standard deviation would indeed mean that uncertainty on this initial vector is null, *i.e.* the LS estimation is no necessary.

- *Process noise of the EKF*

The EKF may accept another parameter, the process noise spectral density matrix. It can be included in the propagation of the EKF covariance matrix, in the form of an additive process noise whose statistics is supposed to be known and white Gaussian.

[5] In the OD, the number of iterations and the different supplemented estimated parameters (like C_R, C_{RP}, C_D) may also be introduced for LS, while time step value may be introduced for OP.

[6] One has to chose large values. Indeed, small values would influence the estimators in a wrong way, giving them an initial state vector very well known ; the estimation would then be not necessary.

This white Gaussian force term may be written as $\ddot{\mathbf{r}}_{unmodeled} = \mathbf{w}(t)$ with properties of relation **(2.25)**, and where the covariance functions is defined through the 3×3 diagonal spectral density matrix $\mathbf{Q}(t)$ introduced in the set-up. Here the diagonal elements are supposed identical and limited as in Table 6.10.

Model parameter	Reasonable value
Q(t)	$(10^{-15} m/s^2)^2$ to $(10^{-3} m/s^2)^2$ Default value : $(10^{-5} m/s^2)^2$

Table 6.10

6.1.3. Simulation plan

As described in the two preceding sub-sections, there are two important group of parameters that can be spanned : the dynamic and the measurement ones. The dynamic parameters of the set-up file may be divided into three sub-groups : the process noise of the EKF, the initial guess and the dynamic model parameters. The measurement parameters may be divided into four sub-groups : the unmodeled physical parameters of the model file, the random noise in the stations file, the number of stations and the scheduling.

The simulations are organised in seven classes a, b, c, d, e, f, g around these seven sub-groups of parameters ; for each class, the selected parameters are represented by a number (1, 2, 3, 4) as follows

- *Process noise* **a** : Low (1) - Standard (2) - High (3)
- *Initial guess error* **b** : Null (1) - Standard (2) - Large (3)
- *Dynamic model* **c** : Full model (1) - Simplified model (2) - Wrong CR and CRP for estimators (3) - Wrong SRF model (4)
- *Error model* **d** : Full (1) - None (2)
- *Noise* **e** : White noise (1) - Fitted noise (2)
- *Schedule* **f** : Middle (1) - Beginning (2) - End (3)
- *Station combination* **g** : Four stations (1) - Two stations (2)

i. Simulations

The simulations are thus divided into four main groups :

- Simulations checking that the estimators are unbiased : **a(2), b(2), c(1), d(2), f(2), g(1),** for **e(1,2)**, where there is no error model (the seven couples of parameters are nulls), for white and fitted noise.
- Simulations which characterise the EKF and LS behaviour : **c(1), d(1), e(2),** for **a(1,2,3), b(1,2,3), f(1,2,3)** and **g(1,2).**
- Simulation to analyse the effect of the dynamic model : **b(2), d(1), f(2), g(1),** for **a(2,3)** and **c(2,3,4).**
- Simulations with real input data : **b(2), e(2), g(2)** for **a(2,3), c(1,2)** and **d(1,2)** (the schedule file being not used).

Note :

The simulation plan then requires 2 + 36 + 6 + 8 = 52 simulation cases. This number is reasonable and allows to run the process a number N of times sufficient to reach the Monte Carlo goals in a feasible time (approximately one week of CPU time).

With the information on the different parameters of the models and set-up files given before, it is now possible to use tables where the different parameters values and corresponding index of model and set-up files are presented. These tables are presented in **Annexe 6**, and are constructed to characterise the measurement parameters with their corresponding model file in the form "Sim_z", and the dynamic parameters characteristics with their corresponding set-up in form "Std_yy". The last table of Annexe 6 characterises the corresponding parameters level values of the simulation tables, *i.e.* the numbers **(1)**, **(2)**,...

6.2. Simulation results

In this part, main simulations, which are detailed in the **Annexe 6**, are presented. They have been realised for 50 runs and statistically analysed with a confidence level of $\alpha = 0.05$. Furthermore, as minimum and maximum of propagated differences are used, each of these samples contains 100 realisations. This number of realisations guarantees the assumptions of the Central Limit Theorem, which is implicitly applied, and allow computation of mean confidence intervals described in **Chapter 5**.

The results are presented in paragraph 6.2.3 to 6.2.3 through the evolution[7] over two days of the worst case estimation errors along the normal, radial and tangential directions (in meters) as a function of the tracking interval (in hours), and one corresponding global evolution[8] for both methods.

Nevertheless, before presenting these main results, paragraph 6.2.1 shows the different information that can be used during a *typical simulation* to observe the correctness and convergence of estimation at each run of the process, and paragraph 6.2.2 details first simulations realised to adjust the error model parameters.

6.2.1. Results of a typical simulation

i. Graphical results

Two graphical representations may be used to observe the accuracy of estimations. They are the residuals plots, given directly by the program ORBIT [Mon 96], and the EKF covariance matrix, which need to be drawn from an output file of the program KALMAN [Wel 97] containing the covariance matrix evolution.

[7] Where solid line represents the EKF evolution and dashed line the LS evolution

[8] Where solid line represents the mean position error, dotted lines represents the associated confidence interval and dash-dotted lines represents the 3-sigma error band.

- *The plots of residuals*

These plots are obtained when an orbit determination is realised, that is for a reference LS estimation of real tracking data, and for LS and EKF estimations of corresponding simulated tracking data. A first comparison of both methods could be realised from these plots, through the quality of estimation which is represented in these plots of residuals. Nevertheless no statistical information is available from these plots, and furthermore there is a striking resemblance between the residuals of the EKF in steady state and the last iteration of LS algorithm [Fra 98], [Wel 97]. For these two reasons, they are not used in main simulations, and residual plots for reference estimation with both single- and multi-stations are here presented. However, in paragraph 6.3.3, these plots are employed to compare LS and EKF adaptive behaviour

Figure 6.1 shows the residual plots for reference LS estimation of trilateration measurements with three stations.

C

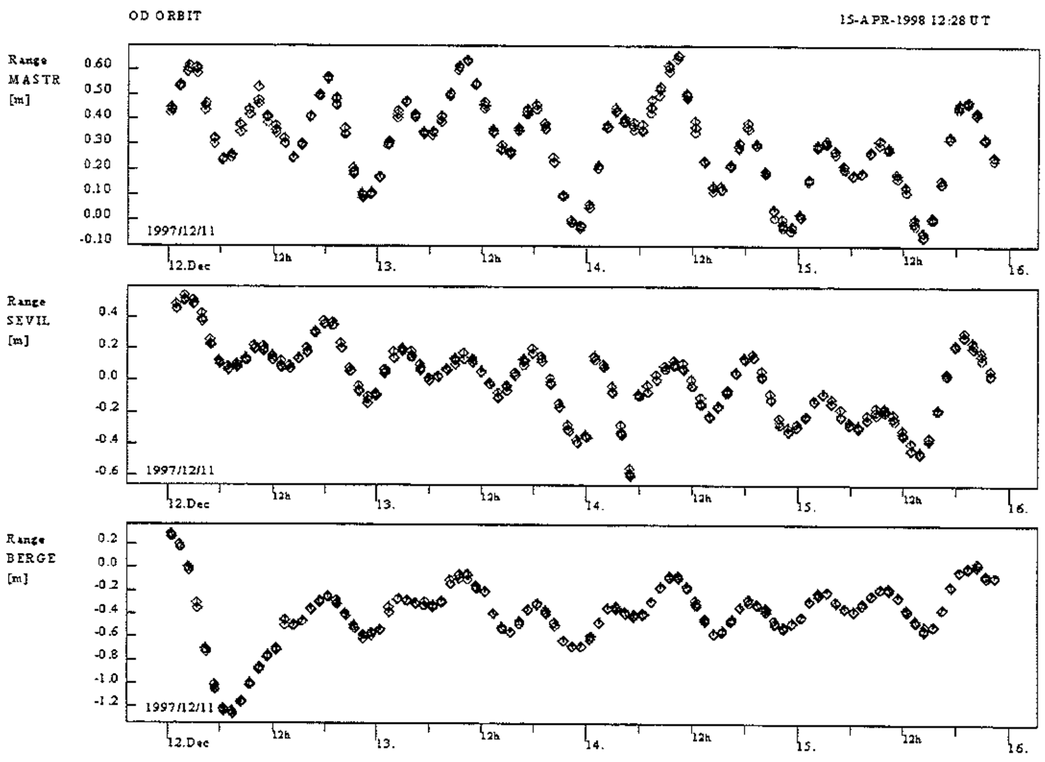


Figure 6.1

For comparison, Figure 6.2 shows the residual plot of a reference LS estimation from single station measurement, *i.e.* range, azimuth and elevation. On this plot, the two angular components also present periodic oscillations, but with a greater noise which is due to unpredictable atmospheric phenomena, while the range residual evolution has the same shape as for trilateration. The standard deviation of the range measurements are supposed to be 2 meters, and those of azimuth and elevation measurements are 20 arc-seconds.

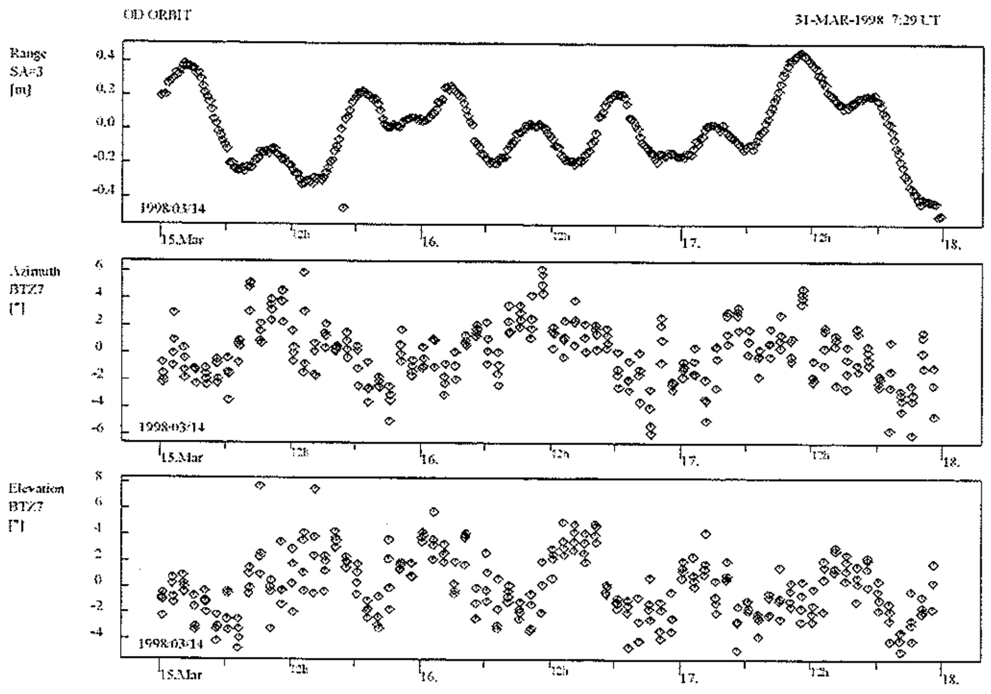


Figure 6.2

- *The evolution of the EKF covariance matrix with time*

Another time evolution can be drawn from the EKF covariance matrix. It is computed with EXCEL© from the EKF_SV.KF output of KALMAN, and represent the evolution of diagonal position elements. It thus gives a first visual representation of the EKF estimation error over time. The conclusion of such representation is that the EKF error decrease quickly, so the estimator converges.

Nevertheless, these figures do not provide any statistical information as they come from a single run, and cannot be drawn for LS as it would involve to propagate the estimated initial covariance matrix. So these figures are only used to be compared with statistical results in paragraph 6.3.

Figure 6.3 shows this evolution in the case of the simulation case 050052, for the three Cartesian positions.

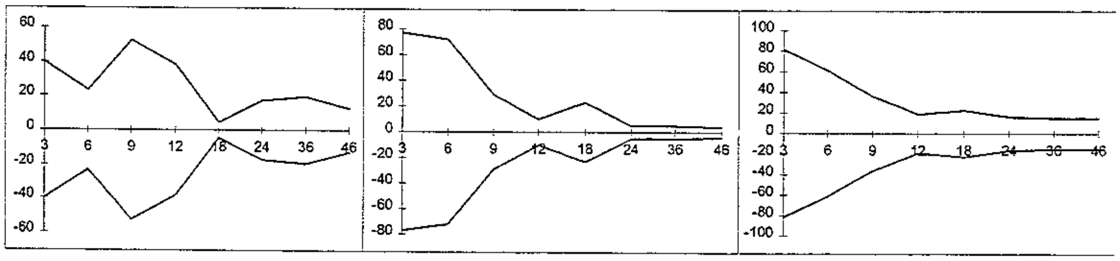


Figure 6.3

ii. **Data results**

To examine a simulation run it is also necessary to consider data files, where quantitative information is available. Among the different outputs of programs ORBIT [Mon 96] and KALMAN [Wel 97], two files are more interesting : the summary files of LS estimations for reference and simulated trajectory, which summarise the estimation and is produced by ORBIT, and the LSQ.OUT and EKF.OUT files, which contain all information on estimation and propagation.

Nevertheless these files do not yield any statistical result, and the summary file cannot be obtained for EKF. So an example of a LS summary is solely given in **Annexe 5**.

6.2.2. Error model effects

Before running main simulations, a better "feeling" of the effects of error model components is needed. Simulations thus have been realised (with 10 runs) to test the corresponding parameters effect and obtain a reasonable error model file. Results are drawn through mean and standard deviation of the differences in satellite axes at last epoch (48hours) in different tables.

The initial stdev was set to $(10000m, 10000m, 10000m, 0.1m/s, 0.1m/s, 0.1m/s)$, the process noise to $\left((10^{-9}m/s^2)^2, (10^{-9}m/s^2)^2, (10^{-9}m/s^2)^2\right)$, biases were estimated for Bergen and Rome and set to zero for Betzdorf and Seville, and additive white noise of 0.8 m for mean and 0.3m for stdev was used.

- *No error model*

With only white noise, the values of for the three local components are given in Table 6.11, and correspond to the graphical results given in paragraph 6.2.4.

Mean (m)	**Stdev (m)**	**LS radial / tangential / normal (m)**	**EKF radial / tangential / normal (m)**
0	0	0.2 ± 2.3 / 2.3 ± 6.8 / 1.6 ± 4.5	0.3 ± 2.1 / 2.5 ± 7.8 / 1.8 ± 5.1

Table 6.11

- *Station calibration constant term (m)*

With the constant term C_1, the differences are given in Table 6.12. For a standard deviation greater than 10m, both estimations diverge and means become generally larger than 100m. So this term must be handled with care and set to a reasonable value in the complete model.

Mean (m)	**Stdev (m)**	**LS radial / tangential / normal (m)**	**EKF radial / tangential / normal (m)**
0	1	0.4 ± 6.5 / 5 ± 12.3 / 2 ± 8.2	1.4 ± 8.7 / 6.5 ± 14.2 / 3.2 ± 6.9
0	10	0.5 ± 9.6 / 7 ± 50.8 / 3.5 ± 13.5	0.7 ± 11.3 / 10.2 ± 53.1 / 5.1 ± 17.3
0	100	85 ± 117 / 465 ± 290 / 260 ± 120	96 ± 32 / 520 ± 360 / 235 ± 170
1	10	1.8 ± 7.4 / 6.4 ± 51.5 / 2 ± 40.6	2.3 ± 8.3 / 7.9 ± 67.9 / 3.1 ± 23.6

Table 6.12

- *Spacecraft delays terms*

Linear term ($m/hour$)

A more important term is the linear part of the spacecraft delay effect. Table 6.13 shows that even for small values of the standard deviation, the tangential mean is relatively large and the three components would diverge for larger input values.

Mean (m)	Stdev (m)	LS radial / tangential / normal (m)	EKF radial / tangential / normal (m)
0	0.1	0.8 ± 2.2 / 9.3 ± 13.9 / 2.4 ± 6.8	1.5 ± 3.1 / 7.5 ± 12.7 / 4.1 ± 9.2
0	0.5	1.5 ± 6.1 / 52 ± 168 / 8.6 ± 26.2	3.2 ± 5.8 / 68 ± 213 / 12 ± 32.5

Table 6.13

Periodic term (m)

The periodic term of spacecraft delay effect is presented in Table 6.14. It is also a very important term, as estimations diverge for relatively small values of the input standard deviation (10 meters).

Mean (m)	Stdev (m)	LS radial / tangential / normal (m)	EKF radial / tangential / normal (m)
0	0.5	1.7 ± 8.3 / 5 ± 26.9 / 5 ± 15.1	1.9 ± 12.6 / 7 ± 19.7 / 4.2 ± 12.2
0	1	5 ± 14.3 / 30 ± 75.1 / 23 ± 39.8	7.8 ± 12.5 / 42 ± 94.4 / 35.9 ± 51.2
0	10	58 ± 125.6 / 1260 ± 369 / 950 ± 197	89 ± 155 / 1670 ± 263 / 780 ± 291

Table 6.14

Phase (rad)

The phase term of the spacecraft delay effect does not have a great importance as its corresponding errors stay under 5m for both methods as shown on Table 6.15, for an amplitude of 0m ± 0.1m and 0m ± 0.2m for both linear and periodic terms

Mean (m)	Stdev (m)	LS radial / tangential / normal (m)	EKF radial / tangential / normal (m)
1.57	1	0.5 ± 6.9 / 3 ± 17.8 / 0.7 ± 4.6	0.7 ± 9.0 / 3.8 ± 22.1 / 0.9 ± 3.5

Table 6.15

- *Ionospheric terms*

The ionospheric refraction is modelled here through a daily/night term and a phase term. The first two have similar roles and their influence is presented in Table 6.16, while the last term has a lower influence (similarly to the phase term of the spacecraft delay effect) and is presented in Table 6.17.

Ionospheric daily and night term

This term shows also a great sensitivity to its input parameters, as well for daily as for the night. The daily term induces divergence from a standard deviation greater than 5 meters, while the night term gives divergence for a mean input term greater than 0.1 meters.

Mean (m)	Stdev (m)	LS radial / tangential / normal (m)	EKF radial / tangential / normal (m)
0.8 0.08	0.6 0.06	1.1 ± 3.2 / 3 ± 11.9 / 1.8 ± 5	0.8 ± 1.9 / 4 ± 16.7 / 2.9 ± 9.1

1 0.08	0.6 0.06	2.0 ± 4.3 / 5.9 ± 25.6 / 4.1 ± 20	0.9 ± 6.3 / 4.7 ± 17.3 / 8.1 ± 10.2
1 0.08	10 0.06	10.2 ± 23.3 / 190 ± 53 / 80.9 ± 52	26 ± 33.2 / 485 ± 69 / 157.5 ± 36.1
1 0.1	0.6 0.06	8.9 ± 43.2 / 240 ± 96.7 / 88.9 ± 29.1	7.8 ± 31.9 / 184 ± 59.7 / 92.9 ± 31.1
0.8 0.08	0.6 0.5	1.9 ± 3.2 / 6.7 ± 13.3 / 10.1 ± 8	2.2 ± 5.3 / 8.7 ± 10.3 / 8.9 ± 17.2

Table 6.16

Ionospheric phase

Like the spacecraft delay phase, the ionospheric phase term has a relatively small impact on estimation errors as can be seen on Table 6.17, for 0.8m ± 0.06m and 0.08m ± 0.006m of daily and night terms.

Mean (m)	**Stdev (m)**	**LS radial / tangential / normal (m)**	**EKF radial / tangential / normal (m)**
1.57	0.26	0.3 ± 2.6 / 2.5 ± 5.1 / 0.5 ± 1.2	1.0 ± 2.5 / 3.3 ± 4.9 / 0.2 ± 1.7

Table 6.17

- *Complete error model*

The complete error model of Table 6.18[9], which is used for main simulations, has been chosen from the physical results of [Wau 95] and adapted from the previous considerations.

Error Model parameters	**Mean (m)**	**Standard deviation (m)**
$G(m\_stacal, s\_stacal)$	0.0 m	2.0 m
$G(m0\_delay, s0\_delay)$	0.0 m	0.1 m
$G(m1\_delay, s1\_delay)$	0.0 m	0.2 m
$G(m2\_delay, s2\_delay)$	1.57 rad	1.0 rad
$G(m1\_ion, s1\_ion)$	0.53 m	0.37 m
$G(m2\_ion, s2\_ion)$	0.053 m	0.037 m
$G(m3\_ion, s3\_ion)$	1.57 rad	0.26 rad

Table 6.18

Table 6.19 shows the corresponding values of differences for both methods. They present the same order of mean error ($\approx$ 1m for both radial and normal components and 10m for tangential component) when the complete error model is introduced.

LS radial / tangential / normal (m)	**EKF radial / tangential / normal (m)**
1.5 ± 2.1 / 8.3 ± 25.6 / 2.1 ± 6.4	1.1 ± 2.8 / 9.2 ± 26.6 / 3.8 ± 5.7

Table 6.19

[9] Values for the main station, Betzdorf.

6.2.3. Normality results

Before the results of the different simulation cases, the normality test results are presented. Indeed this test, which has been realised by the program ANALYSIS of the post process, was not able to conclude to the normality of the different populations at the level 0.05. It means that we don't know if populations are normal, so we only can say that normality may not be assumed for a lot of cases, and that in general normality may not be used for statistical analysis.

Table 6.20 and Table 6.21 show results for 7 different simulation cases whose index corresponds to the label xxxyyz resented in **Chapter 4** and detailed in **Annexe A7** for both position and velocity components. A cross means that the Kolmogorov-Smirnov test has concluded to normality for the corresponding simulations, for the other simulation no conclusion can be drawn.

EKF	**radial position**	**tangential position**	**normal position**	**radial velocity**	**tangential velocity**	**normal velocity**
050012	X		X	X		
050043				X		X
050074	X				X	
050105	X	X		X		
050196			X			
050216			X		X	
050257			X			X

Table 6.20

LS	**radial position**	**tangential position**	**normal position**	**radial velocity**	**tangential velocity**	**normal velocity**
050012		X				X
050043				X	X	
050074	X	X				X
050105	X				X	
050196			X			X
050216			X		X	
050257		X		X		

Table 6.21

From such tables (which may be drawn similarly for all cases), it is clear that normality may not be assumed in general. So statistical analysis is done without this assumption.

In order to investigate normality of Keplerian element differences, a Kolmogorov test has been done as well. The results are similar and may be seen on Figure 6.4 and Figure 6.5, where histograms of the semi-major axis **a** differences have been produced by EXCEL© for simulation cases 050082 and 050216. The would not allow to conclude, but similar histograms can be obtained for the cases, and tables similar to

Cartesian ones can also be drawn, they would show the same kind of results. So normality of Keplerian elements, as well as for Cartesian elements, cannot be assumed in the simulation results.

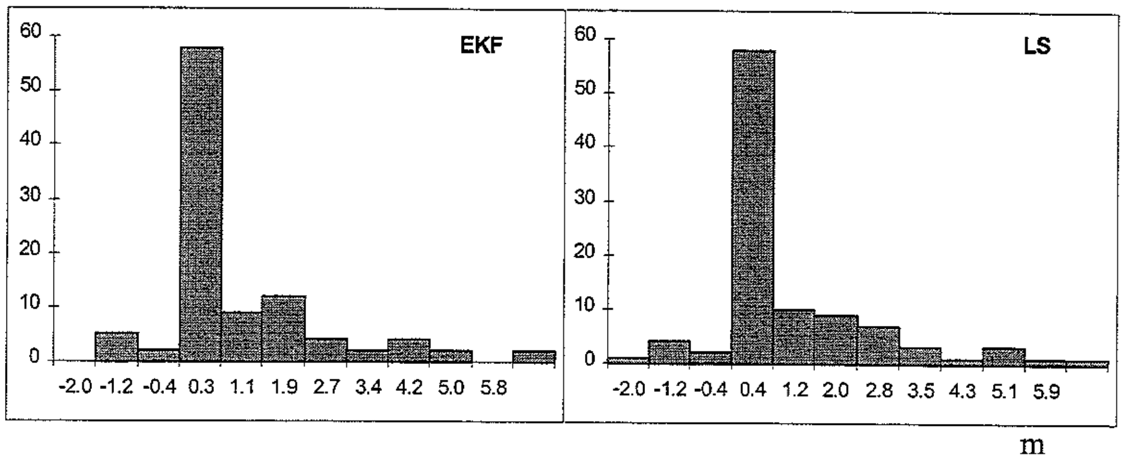


Figure 6.4

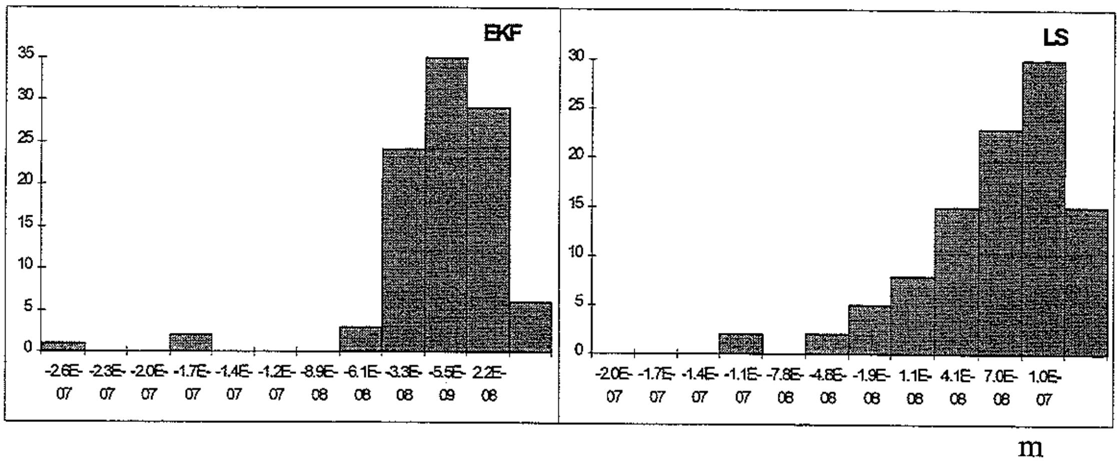


Figure 6.5

These histograms are not generally used in this work, because of their varying shape from one case to another. Furthermore, a single diagram could induce a conclusion which is probably not true in general.

6.2.4. Bias testing

To check that both estimators are unbiased, two simulations have been realised with no error model for both white and fitted Gaussian noise, with the same dynamic model for the reference trajectory and the LS and EKF estimations, with a process noise set to $(10^{-9} m / s^2)^2$ in each direction and a standard initial guess and covariance errors[10].

[10] See Annexe **A.6.5** where the different values of initial errors, schedule parameters, dynamic model parameters are presented.

The results, on Figure 6.6 to Figure 6.8, confirm that LS and EKF implementations are unbiased estimators, as their estimation means are very near to zero with a 3-sigma bands of some meters, when only white or fitted measurement noise is introduced in the reference tracking data.

i. Simulation with white noise (050000)

For this simulation, the tangential component evolution has been chosen and presented on Figure 6.6, where its mean (solid line) and confidence interval (dotted lines) evolution with the 3-sigma limits (dash-dotted lines) versus the eight tracking interval epochs. On this figure, the mean stays under 1 m for both methods with a 3-sigma lower than 3 m. This represents a very good estimation error.

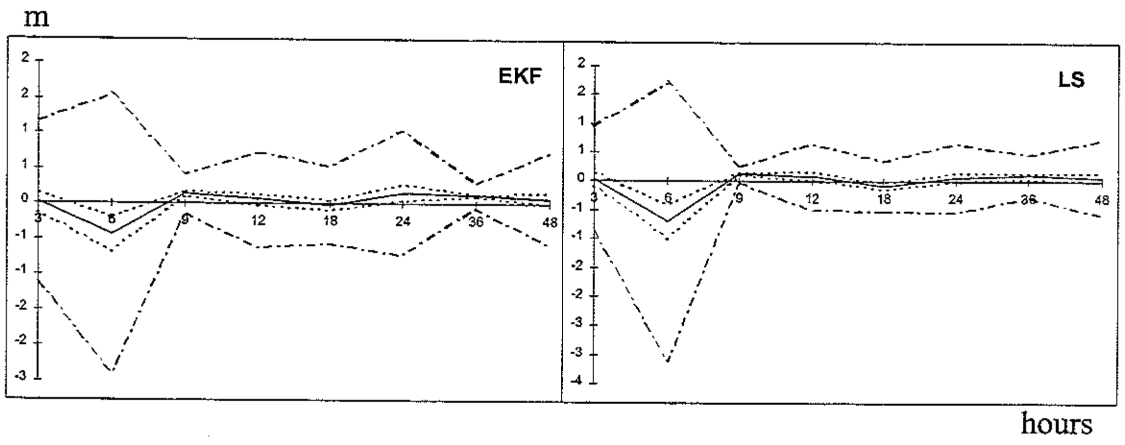


Figure 6.6

The worst case evolution versus the same eight epochs, on Figure 6.7, confirms this conclusion. Additionally, it shows a very close agreement of normal component evolution of both methods, while the two other components have a random behaviour which is due a great sensitivity to the additive noise, as estimation errors are very small.

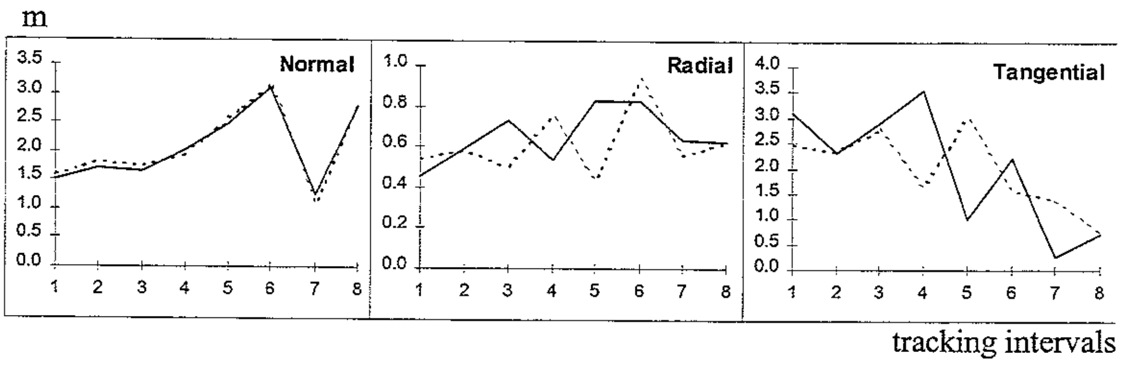


Figure 6.7

ii. Simulation with fitted noise (050001)

In this case, the mean error also stays under 1m with a 3-sigma limit lower than 2 m, so this estimation is of the same order as with white noise. The evolution of the worst case variable is presented on Figure 6.8 and gives the same conclusion. It also shows that the behaviour of the three components is less erratic, as the additive noise is fitted on residuals.

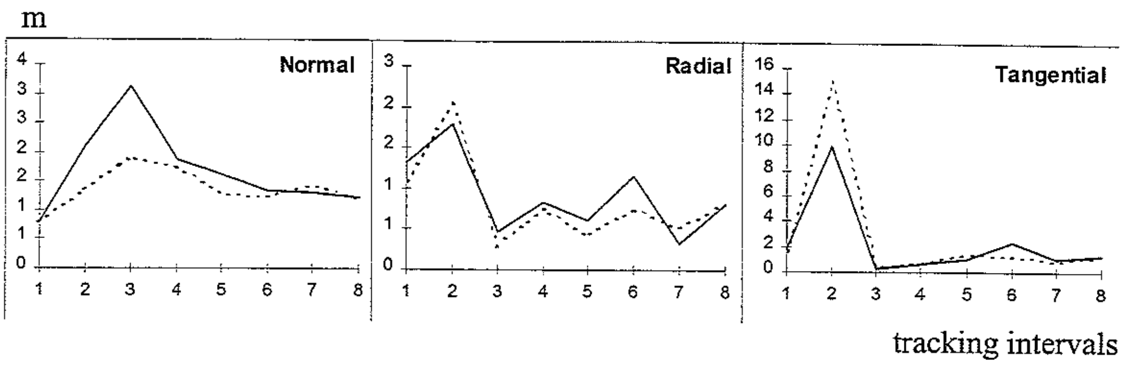


Figure 6.8

6.2.5. EKF and LS behaviour

To characterise the EKF and LS behaviours, 36 simulations have been realised. They are composed of simulations for two and four stations, with different input process noises, for different initial guess errors, and with schedules providing simulated measurement distributed in the centre, at the beginning, and at the end of the tracking interval of two days. In these simulations, the dynamic model is the same for both estimations and for the reference trajectory construction.

As it is not possible to present here all results, the most important cases are presented, and a first general conclusion can be drawn from this second group of simulations : both methods present similar behaviours in terms of time evolution, and are less sensitive to initial errors, process noise and schedule effect when trilateration is used than with single-station tracking. Nevertheless, simulations realised cannot allow to conclude on the superiority of one or the other method.

In this paragraph, the three groups of simulations testing schedule, initial error and process noise effects are presented through the usual worst case and global form versus the eight tracking interval epochs, for some simulations cases, the results of the others simulations yielding the same conclusions. These effects are presented separately to correctly analyse each effect and to avoid influences which could cancel each other.

i. Schedule effect

The schedule effect is presented throughout simulations 050012, 050013 and 050014 where the three types of schedule have been introduced, and where the initial error is standard [10] and the process noise is $(10^{-7} m/s^2)^2$ in each direction. From the different following figures, both methods seem to be less sensitive to a schedule effect with trilateration than with single-station tracking, the other simulations whose set-up come from 02 to 09 giving the same kind of results.

Furthermore, an important conclusion comes up from this first group of simulations and is valid in general for the different simulations : both estimators converge in a relatively short time, *i.e.* less than 24 hours to have a 3-sigma band lower than 50m for radial propagated component and lower than 200m for normal and tangential propagated components.

- *Middle schedule (050022)*

For a scheduling which gives measurements equally distributed along the tracking interval, the normal component evolution of both estimations errors (Figure 6.9) presents a very close agreement between the two estimation methods.

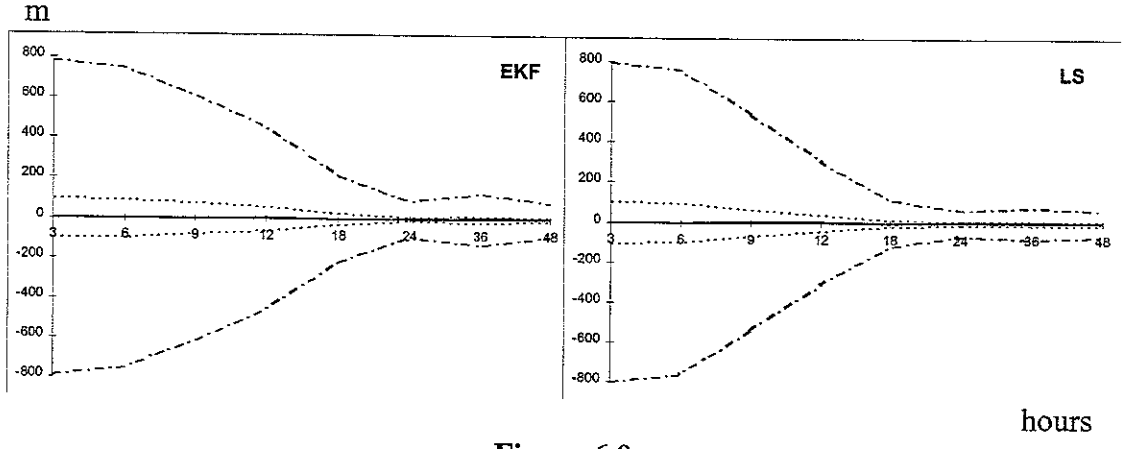


Figure 6.9

The corresponding worst case evolution of normal, radial and tangential components presented on Figure 6.10 indicates that this similarity may be extended to the three components.

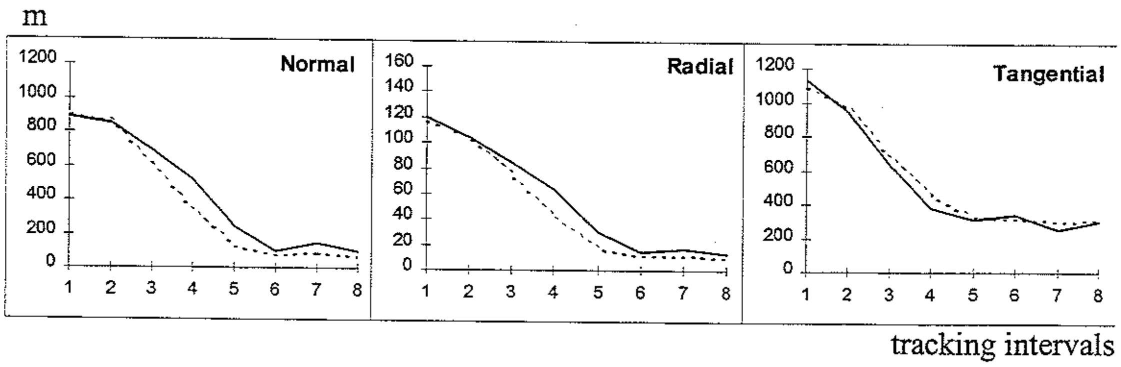


Figure 6.10

- *Beginning schedule (050023)*

For a schedule which concentrates the tracking data at the beginning of the interval, the resulting worst case error evolution is similar to the normal scheduling one, as shown on Figure 6.11.

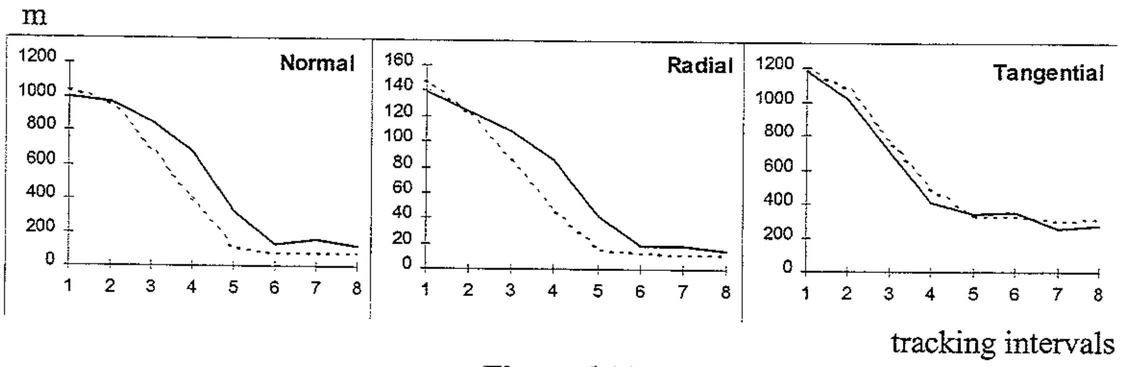


Figure 6.11

- *End schedule (050024)*

Similarly, for an end interval scheduling, the worst case error evolution, on Figure 6.12, for both estimations errors has the same shape and presents a similar convergence.

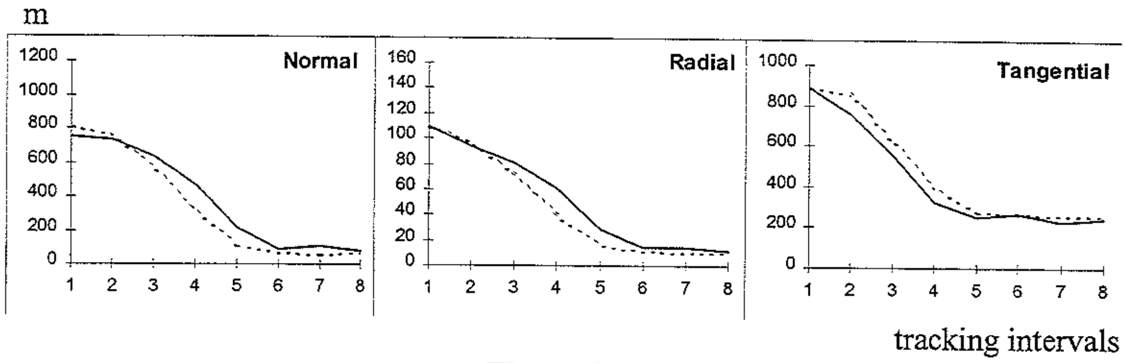


Figure 6.12

An important general conclusion can here be drawn from these three simulations : trilateration seems to render both methods less sensitive to schedule effects.

ii. Initial guess effect

The initial guess error and covariance error effects are presented throughout simulations 050012, 050042, and 050072 where their values are respectively standard, null and large[10], and where the schedule is standard and the process noise is $(10^{-9} m/s^2)^2$ in each direction.

- *Standard initial error (050012)*

For a standard initial error, the tangential evolution of both methods is shown in Figure 6.13, and the corresponding worst case in Figure 6.14. Both methods have a similar behaviour. Furthermore, Table 6.22 shows the last epoch mean ± standard deviation, confidence limit and worst case for both methods and for the radial component, in the case of the three simulations presented here.

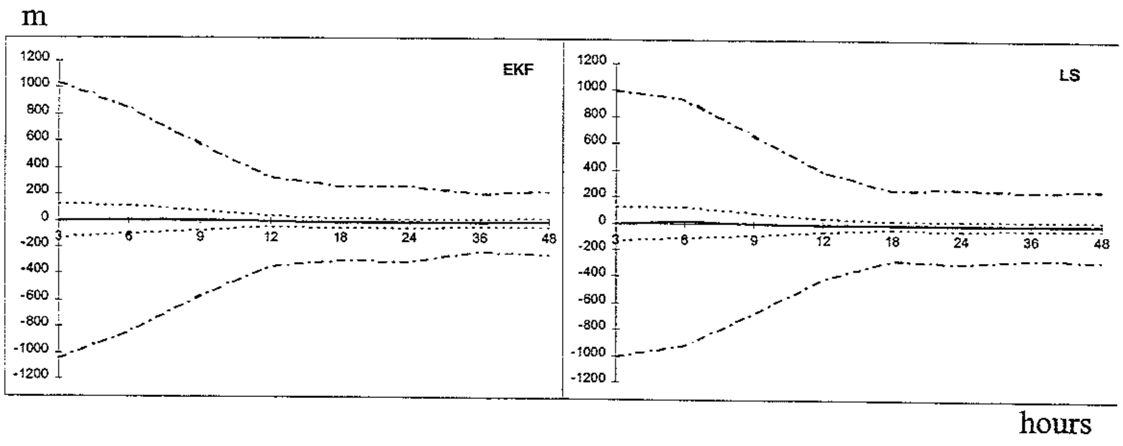
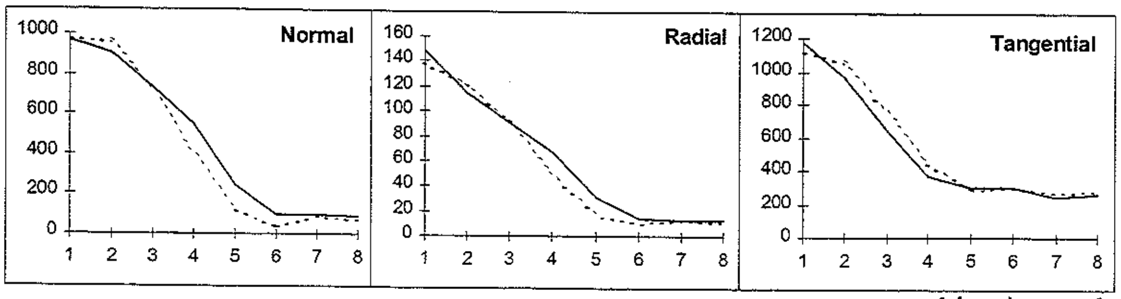


Figure 6.13

m



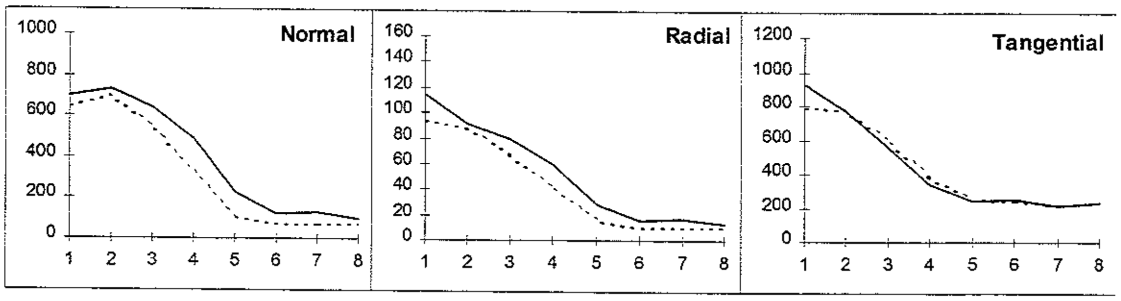
tracking intervals

Figure 6.14

- *Null initial error (050042)*

For a null initial error, the worst case becomes better at the beginning of the tracking interval than in the standard case, as shown in Figure 6.15, with a similar decrease for both methods and for the three components. However, the final estimations are of the same order of magnitude.

m



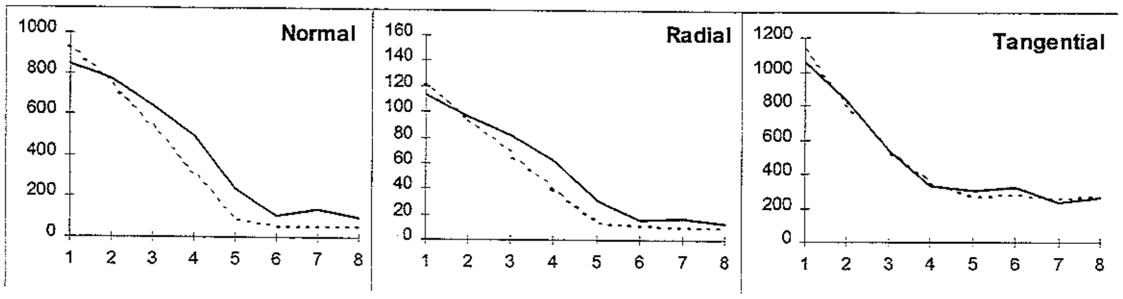
tracking intervals

Figure 6.15

- *Large initial error (050072)*

For a large initial guess error, all the rest being the same, the evolution is nearly the same as shown on Figure 6.16. Thus we conclude from these three simulations that trilateration seems to render both methods less sensitive to initial parameters, as well as to schedule effect.

m



tracking intervals

Figure 6.16

Table 6.22, where statistics of the last epoch differences for normal, radial and tangential components are drawn, yields to the conclusion that both methods are similar, with a little advantage for LS, and that the initial guess has little influence on the estimation after 48 hours of tracking data, as the trilateration system gives very precise measurements which allows both estimators to converge very quickly.

EKF (m)	Normal Mean ± Stdev	Worst case	Radial Mean ± Stdev	Worst case	Tangential Mean ± Stdev	Worst case
050012	0.06 ± 26.63	89.81	0.48 ± 3.64	12.76	8.38 ± 79.64	260.23
050042	0.03 ± 28.28	95.38	0.40 ± 3.61	12.56	5.78 ± 70.24	231.15
050072	0.10 ± 24.53	82.73	0.22 ± 3.56	12.23	4.95 ± 79.78	268.88
LS (m)	**Normal Mean ± Stdev**	**Worst case**	**Radial Mean ± Stdev**	**Worst case**	**Tangential Mean ± Stdev**	**Worst case**
050012	0.98 ± 22.64	76.49	0.56 ± 3.29	11.67	12.56 ± 88.32	285.86
050042	0.01 ± 21.11	68.32	0.34 ± 2.87	10.04	5.94 ± 70.68	232.85
050072	0.02 ± 20.78	70.43	0.29 ± 2.58	9.02	7.85 ± 85.72	265.57

Table 6.22

iii. Process noise effect

To show the process noise effect, simulations 050042, 050052 and 050062 are examined, with a null initial guess error, and a middle scheduling of the tracking data

- *Normal process noise (050042)*

This simulation has a process noise of $(10^{-9} m/s^2)^2$ in each direction, and is presented through its radial evolution in Figure 6.17, the worst case being in Figure 6.15.

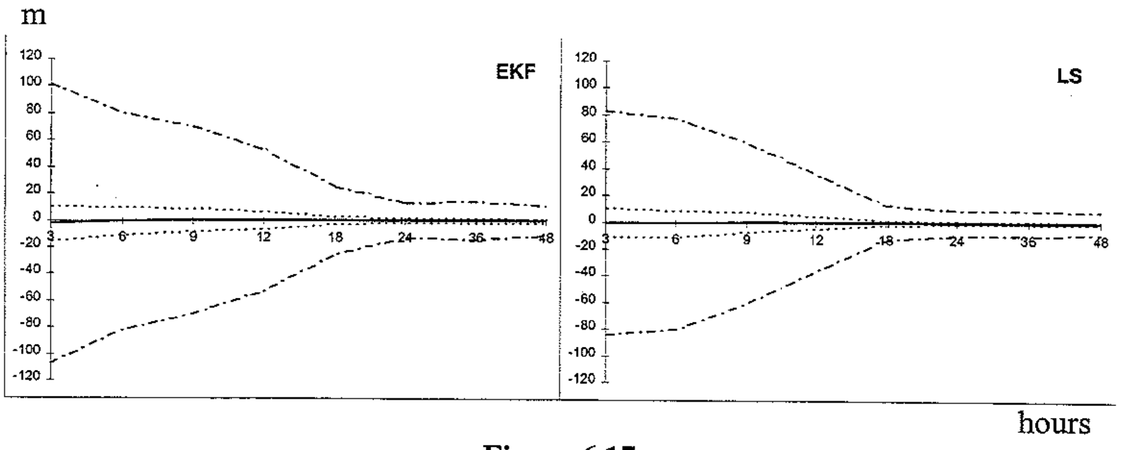


Figure 6.17

- *High process noise (050052)*

In this case, the process noise has been set to $(10^{-7} m/s^2)^2$, *i.e.* a high level. The resulting worst case evolution of Figure 6.18 presents the same characteristics as the preceding one in Figure 6.15.

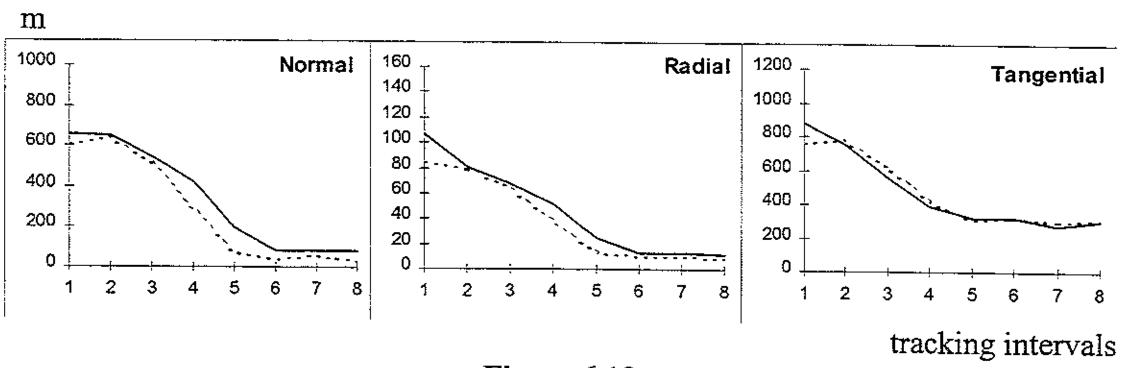


Figure 6.18

- *Low process noise (050062)*

For a low process noise of $(10^{-11} m/s^2)^2$, the worst case shown on Figure 6.19 again presents the same behaviour, the EKF being not better than the LS and even a little worse.

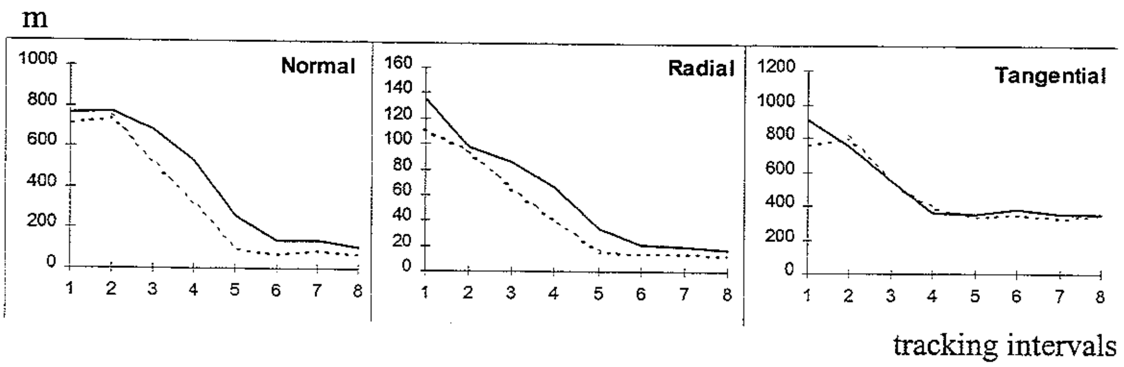


Figure 6.19

The differences between these three simulations are not very significant, so the conclusion is again that trilateration renders the estimation less sensitive to the process noise variations, as for initial error and scheduling effects, and thus places the two methods on the same level of accuracy.

In view of the results presented in the paragraphs 6.2.4 and 6.2.5, it seems evident that both methods have a similar behaviour and react similarly to the different effects introduced in the simulator, trilateration used with four stations probably reducing these effects. Furthermore, both estimators mean errors are very small after 3 hours of tracking data, this effect being again due to the trilateration measurements.

6.2.6. Dynamic model effect

To investigate further the behaviour of both methods, simulations with dynamic model errors and unmodeled accelerations have been realised. They are presented through three simulations. The first one tests the effect of the number of gravity potential development terms included, the second one the C_R and C_{R_P} values effect, and the last one the dynamic noise effect.

The first two simulations use the same dynamic model for both the reference orbit propagation and the estimators, while the last group uses a different dynamic model for reference trajectory and both estimators to simulate an unmodeled force term in the dynamic model of both LS and EKF. Furthermore, these simulations have been realised with real trilateration data, on the contrary of all preceding ones.

- *Simplified dynamic model (050196)*

This first simulation where the dynamic parameters of both reference and estimators model are the same but have been changed from the full model[10] is exposed through simulation 050196[11], where the initial guess error is standard and the process noise has been set to $(10^{-9} m/s^2)^2$.

This simulation has 3 terms in the geopotential part, on input of both reference and estimators set-ups, the rest of the set-ups being the same as before.

On Figure 6.20 the resulting normal error component is drawn, and yield to the conclusion that both estimators reacts similarly to a worse dynamic model, with a little preference for LS which converges more directly and with 3-sigma slightly smaller at the last epoch.

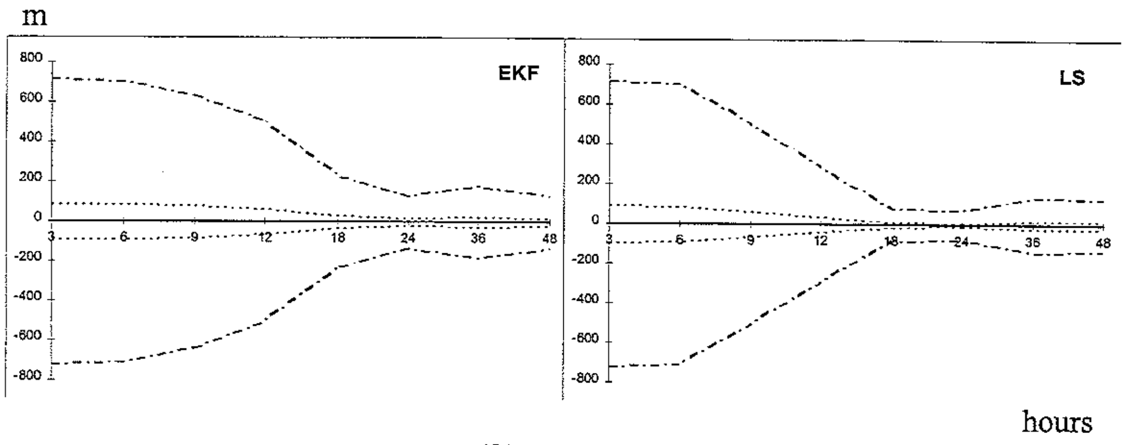


Figure 6.20

Nevertheless, Figure 6.21, where the three worst case errors have been drawn, shows that this conclusion cannot be generalised to the three components, as both tangential evolution present a very close agreement.

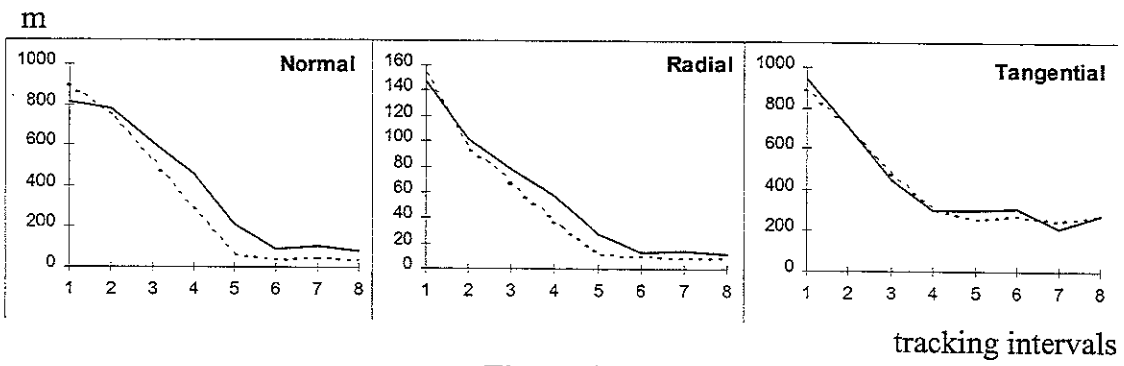


Figure 6.21

- *Wrong C_R and C_{R_p} values (050206)*

When C_R and C_{R_p} input parameters are not correct, *i.e.* arbitrarily set to 1.3 and 0.03 instead of the estimated values 1.2715 and 0.0364, the worst case evolution (Figure 6.22), gives a higher accuracy for the LS method for both normal and radial components, while the tangential one is in a very close agreement.

[11] This simulation results have been described in more details in the article [Hal 98].

Such simulations nevertheless may not yield the conclusion than one method is better, as differences are not significantly enough.

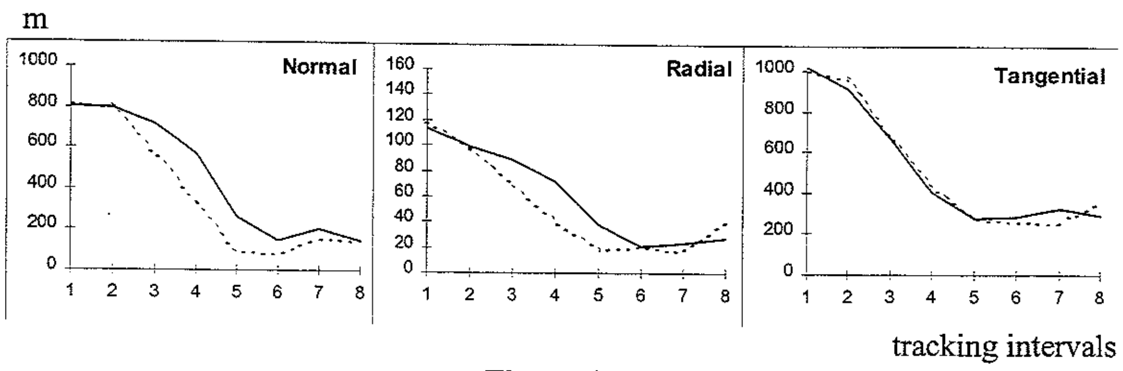
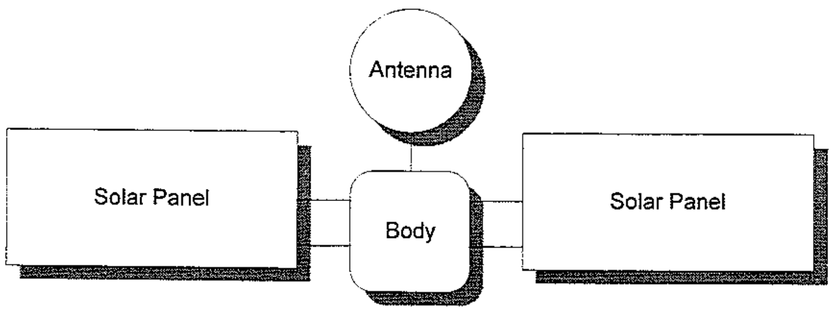


Figure 6.22

- *Wrong estimator model for C_R and C_{R_P} (050216)*

These simulations take into account an imperfect modelling of the trajectory in the estimation softwares.

To this end, the reference trajectory was generated with an improved version of the ORBIT program [Kri 98], where the propagator uses the supplementary modelling of the solar radiation force on the reflectors of the satellites. By contrast, this dynamic behaviour is not modelled by the usual LS and EKF estimators, which only have the capability to estimate inertial fixed components of the solar pressure. Picture 6.1 shows the four parts of the satellite which are modelled in this new version of the ORBIT propagator.



Picture 6.1

Even with an improved propagator for the reference trajectory, which add a dynamic noise for the tracking data estimation, both methods have a similar behaviour, as shown on Figure 6.23 for normal component evolution of this simulation, where a an initial error of $(10000m, 10000m, 10000m, 0.1m/s, 0.1m/s, 0.1m/s)$ has been set with a middle schedule, a process noise of $(10^{-9}m/s^2)^2$.

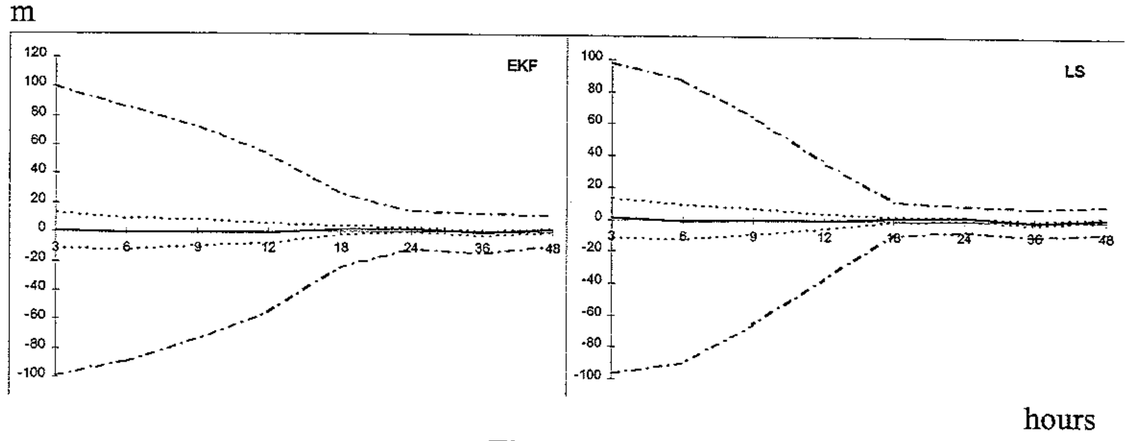


Figure 6.23

On Figure 6.24, the corresponding worst case evolution however shows that the LS presents a slight advantage over the EKF. Furthermore, on Table 6.23 the corresponding normal, radial and tangential mean ± standard deviations, together with the worst case are drawn for the three simulations and for both methods at the last epoch of estimation.

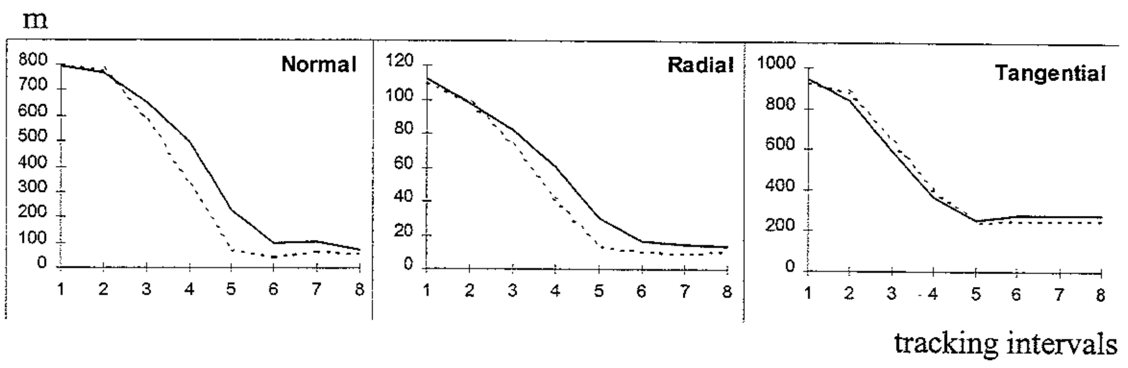


Figure 6.24

EKF	**Normal Mean ± Stdev**	**Worst case**	**Radial Mean ± Stdev**	**Worst case**	**Tangential Mean ± Stdev**	**Worst case**
050196	0.20 ± 26.91	110.60	1.23 ± 4.36	17.52	19.89 ± 72.15	273.20
050206	0.06 ± 44.27	149.57	2.18 ± 11.41	40.71	39.30 ± 117.89	389.03
050216	0.11 ± 35.33	131.26	1.90 ± 6.42	25.23	27.81 ± 79.39	298.61
LS	**Normal Mean ± Stdev**	**Worst case**	**Radial Mean ± Stdev**	**Worst case**	**Tangential Mean ± Stdev**	**Worst case**
050196	0.15 ± 14.78	102.98	1.98 ± 5.12	16.50	18.70 ± 75.61	275.56
050206	0.07 ± 43.69	147.36	2.50 ± 10.17	36.68	27.07 ± 93.68	339.69
050216	0.08 ± 32.74	130.13	1.57 ± 5.98	23.57	22.67 ± 85.32	282.70

Table 6.23

6.2.7. Overview of results

The eight last simulations of the Simulation Plan detailed in **Annexe A7** were expected to be the only ones realised with real trilateration data. However, as such measurements were available from December 1997 at the SES company, previous simulations (paragraph 6.2.6) have been realised with them. So these expected supplementary simulations with real data have already been presented here.

Furthermore, all the results presented in this part 6.2 are commented in **Chapter 7** throughout different conclusions and interpretations. Nevertheless, to improve our understanding of the main result, *i.e.* the similar behaviour of both estimators with few influence of the different parameters, further investigations have been realised and are detailed in the following paragraph 6.3.

6.3. Supplementary investigations

The goal of these supplementary investigations is to analyse some theoretical considerations to explain the similar behaviour of both methods.

The EKF is indeed theoretically better than the LS when the measurements errors are smaller, here with trilateration, because of the relative increase of the dynamic errors which cannot be treated by the LS. However, the precedent results show that both methods are similar, so it is necessary to understand the reason of this difference between theory and practice.

6.3.1. Variation of the process noise level

A first investigation has been done to observe the effect of different process noise levels, from $(10^{-5} m/s^2)^2$ to $(10^{-15} m/s^2)^2$, instead of the three levels $(10^{-7} m/s^2)^2$, $(10^{-9} m/s^2)^2$ and $(10^{-11} m/s^2)^2$ set in the simulations. On Figure 6.25, mean at the last epoch and for the three radial, normal and tangential components is presented versus the process noise level. It can be seen, that there is not really an optimum value. Nevertheless, the process noise level must be higher than $(10^{-7} m/s^2)^2$ and lower than $(10^{-13} m/s^2)^2$ to have a good behaviour of the EKF.

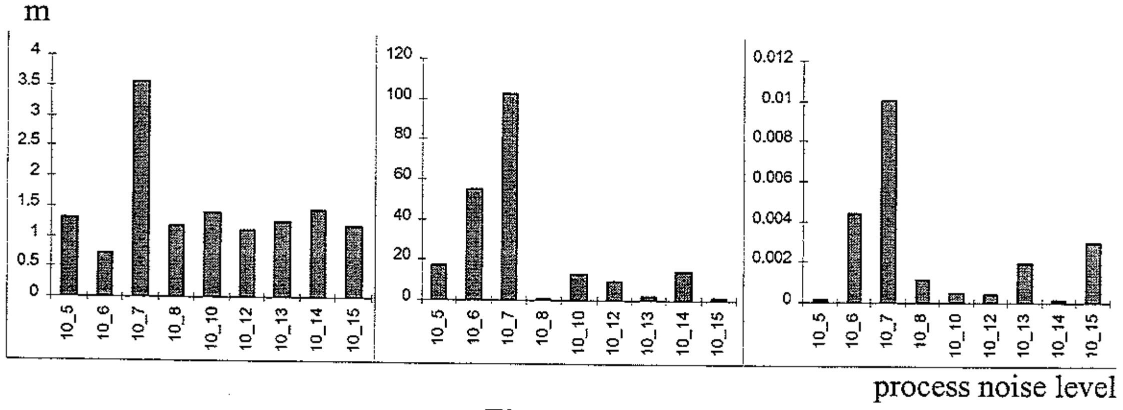


Figure 6.25

It thus shows that the EKF is less sensitive to process noise levels than in single-stations measurements [Wel 97].

Furthermore, there is a trade of between the speed of convergence and the accuracy of the estimation in steady state : the greater the input process noise, the less accurate are the estimations. So if one level has to be set, the values $(10^{-8} m/s^2)^2$ is probably the most appropriate [Hal 98].

6.3.2. Non linearity response

To investigate the reason of the similar behaviour when trilateration is used, non-linearity treatments of both measurement and dynamic models are here explored in two ways.

First, the comparison of the steady state EKF estimation for both one and seven iterations of LS estimations are shown on Figure 6.26, for radial errors (solid lines for LS and dashed for EKF). The *linear* evolution means that the LS only realises one linearisation around its reference trajectory, while the *non linear* evolution represents the estimation after 7 iterations. The LS treats non-linearity very well. Indeed both linear and non linear final errors (after 48 hours of tracking data) have an order of magnitude of 5 meters, the non-linear one being slightly better than the EKF.

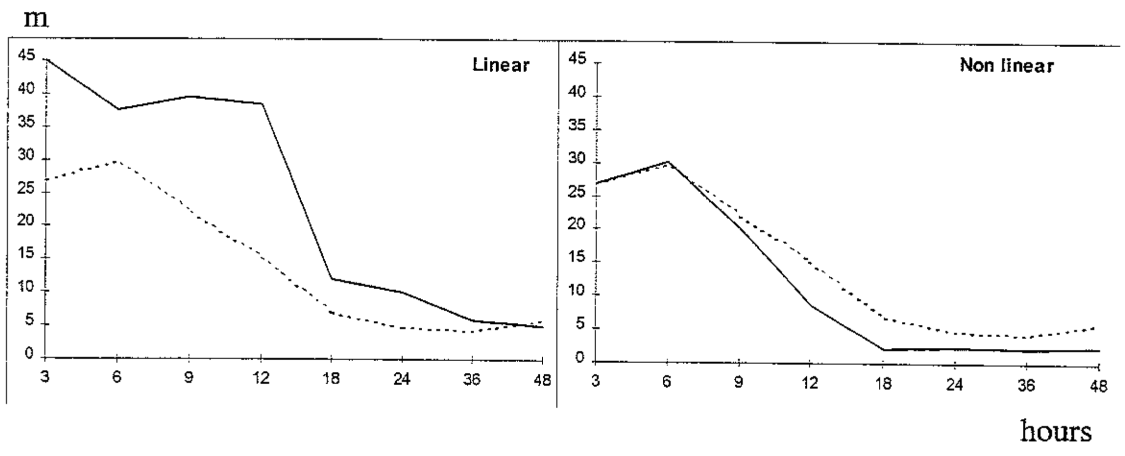


Figure 6.26

Secondly, the longitude error is presented in Figure 6.27, for different number of iterations of the LS algorithm. This figure shows that for a number higher than one, the LS error is of the same order of magnitude. For comparison, the EKF longitude error has an order of magnitude of 5.00E-05°. So the LS algorithm treats correctly non-linearities from the second iteration, which confirms the previous results.

Thus for one iteration, LS is worse than EKF, but for 2 and more iterations it is comparable in its estimation error, and even slightly better. So, this non-linearity cope could be a first explanation of the previous simulation results.

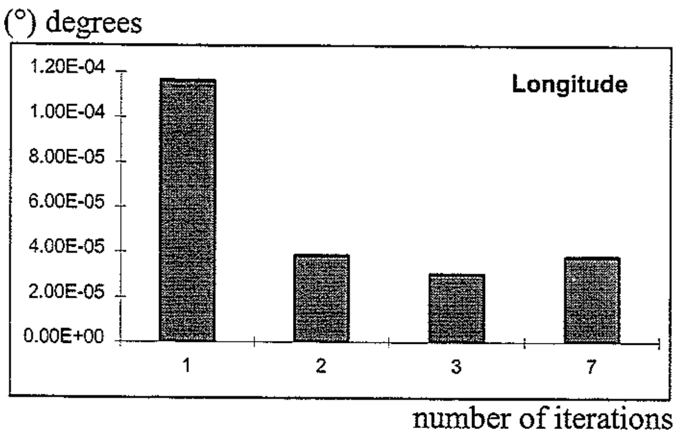


Figure 6.27

6.3.3. Adaptive behaviour of the EKF

To observe the EKF behaviour comparatively with the LS one, plots of residuals, coming from one typical simulation realised with three stations real data, are used in this paragraph. As shown on Figure 6.35 and Figure 6.29 for respectively LS and EKF, the residuals of EKF are smoother comparatively to the reference residual shown on Figure 6.37, these last residuals being the *ideal ones* computed with a very small initial covariance matrix when estimating reference trajectory from real trilateration tracking data.

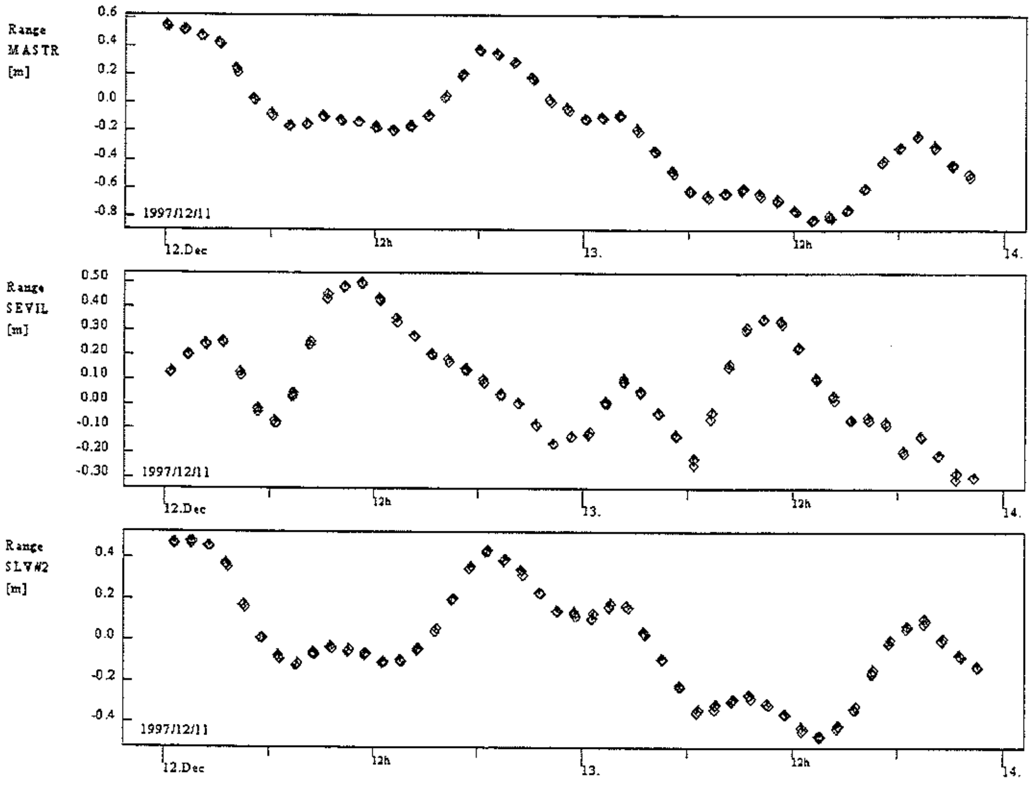


Figure 6.28

This is probably an other explanation of the similarity between both methods observed during the various simulations of paragraph 6.3.

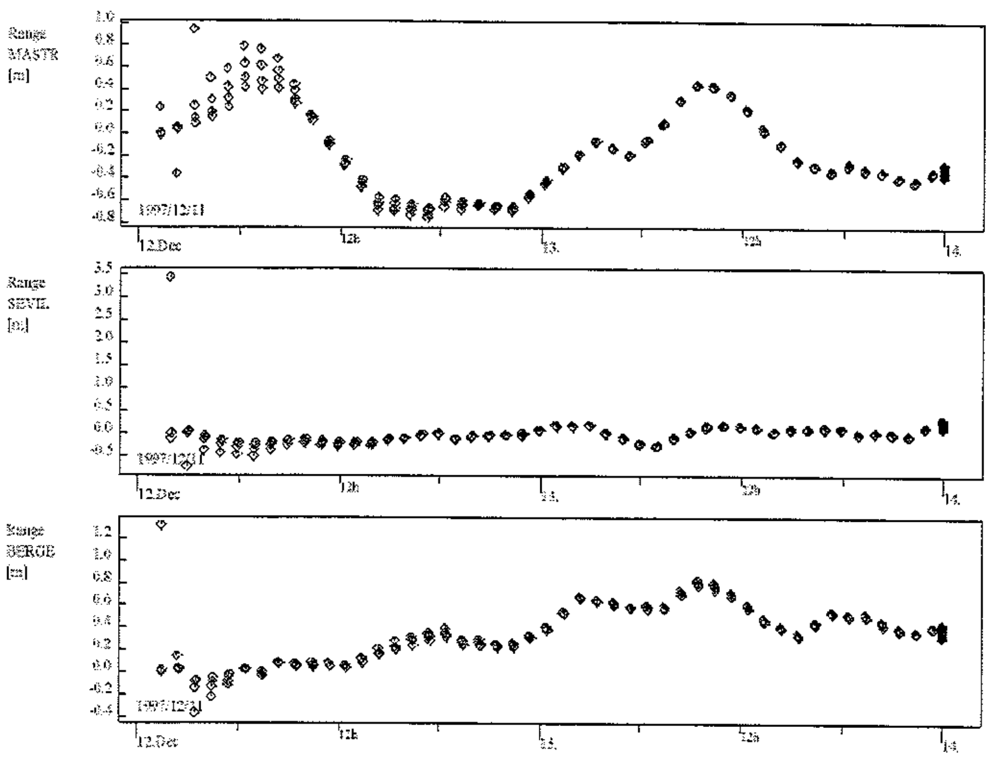


Figure 6.29

This flatness of the EKF residuals shows that the adaptive behaviour of the EKF forces it to follows the systematic measurements errors introduced in the simulation by absorbing them in the position vector.

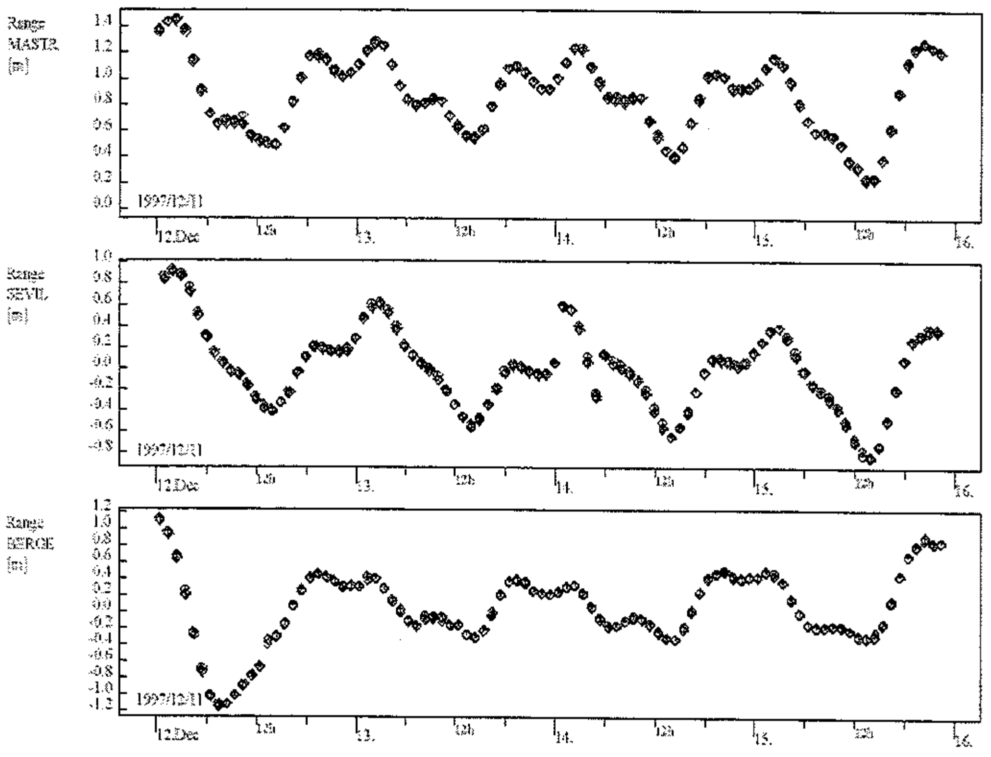


Figure 6.30

Chapter 7.

Conclusions

The result of this Diploma Thesis is the development of a Monte Carlo simulator and post processor software for the comparison between the continuous-discrete extended Kalman Filter and the Least Squares estimator, and its application to tracking data obtained via a trilateration system. The mathematical equivalence of both LS and EKF for the linearised formulation without propagation noise has been recalled in **Chapter** 3. The dedicated simulator and a statistical post processor treatment are exposed in **Chapter 4** and **Chapter 5**, while the simulation results are detailed in **Chapter 6**.

The comparison between both algorithms is based the comparison of their estimations relative to an artificial reference trajectory, and on the consideration of various observation and dynamic parameters described in **Chapter 2**, together with measurement and propagation noises. The following factors have been considered :

- Observation model
  - A white or residual fitted random noise
  - An error model containing ionospheric refraction, spacecraft delay and station calibrations effects
  - The number of stations included in the trilateration tracking system
  - The measurements scheduling
- Dynamic model
  - The number of terms in the geopotential development
  - The solar radiation pressure coefficients
  - The unmodeled forces
- Parameters of estimation methods
  - The input process noise of the Kalman Filter
  - The initial state variance and guess error

In addition, further theoretical investigations complete the comparison of both implemented methods :

- The impact of the process noise level on the EKF estimations
- The non-linearity treatment of the estimators
- The adaptive behaviour of the EKF algorithm

Simulations have shown that the estimation errors of both iterated Least Squares and extended Kalman Filter algorithms have a similar behaviour and order of magnitude. Stated more precisely, from the simulation performed, it is not possible to conclude from the statistics alone that one method surpasses the other.

Nevertheless, this conclusion is drawn from a qualitative interpretation of the results more than a quantitative one. Indeed, as populations could not be supposed normal from the samples considered, it was difficult to find a decisive criterion of comparison, such as the probability of the error at an epoch being lower than a constant value.

Moreover, trilateration measurements seem to render both estimators less sensitive to the observation, dynamic and algorithmic input parameters, since the trilateration is very accurate and allows both estimator to converge in a few hours.

Some explanations and interpretations can be put forward about this similarity of behaviour and decrease of sensitivity.

- Previous work [Wel 97] had concluded that both estimators were equivalent for single-station measurements, more perturbed than trilateration ones. Nevertheless, these simulations were performed with a reference trajectory including dynamic process noise, which is unrealistic for geostationnary orbits as all forces acting on the spacecraft can be modelled with a fairly high level of accuracy. In the present work, both methods have been compared to an artificial reference trajectory constructed with a deterministic propagator, so without process noise. Thus the present construction has probably given an advantage to the LS estimator, which can explain why EKF superiority to LS has not been shown here, therefore leading to similar conclusion as [Wel 97]. In short, the EKF seems to be more accurate with trilateration than with single-station tracking.

Furthermore similar simulations with a deterministic reference trajectory have been realised for single-stations [Hal 98] and have shown that the LS is better suited in that case, which confirms the previous consideration.

- The EKF has an adaptive behaviour by nature, as it relinearises the trajectory as it processes a new measurement, at each prediction step. So it should follow the dynamic evolution more accurately than the LS ; but on the other hand, systematic and correlated measurements errors are reflected in the estimations. So, the decrease of measurement error due to the trilateration system has made the EKF more sensitive to these systematic and correlated errors introduced in the simulator. The EKF thus includes them in its estimations, which decreases its sensitivity to dynamic variation and its precision. This effect can be observed on the EKF plots of residuals where errors patterns observed in the LS and the reference ones are washed out.

Further investigations to understand these two effects are possible.

- A modified version of the Kalman implementation which does not update the reference trajectory (i.e. the non-extended Kalman Filter) could be used with an initial trajectory close to the real one to avoid divergence. In principle, this would allow to observe the behaviour of non-extended KF and its processing of measurement noise to determine which fraction of the EKF estimation is due to absorption of the systematic and correlated errors.

- The propagation noise of these simulations is probably too small to observe the Kalman adaptive behaviour to dynamic variations. To increase this noise, both estimators could be applied to ionic propulsion manoeuvres of several hours with a random effect on its corresponding acceleration term in the dynamic model. In this case the EKF should be better suited than the LS to adapt its estimation of the trajectory, if the spectral density matrix is increased for the duration of the manoeuvre as it would then become more receptive to the latest measurements.

This work has thus principally shown that the EKF offers an alternative to the usual LS algorithm implementation to estimate the position of a geostationnary satellite, when a trilateration tracking system is used to collect measurement data. The EKF could thus be applied to systems for a truly real time orbit determination, *i.e.* if instantaneous spacecraft position needs to be broadcast to user terminals.

Annexes.

A.1. Hierarchical design of the simulator.

A.1.1. Data structures of the new F90 programs

- General parameters for the different programs

```
INTEGER, PARAMETER :: MAX_STATIONS=10    ! maximum number of tracking stations
INTEGER, PARAMETER :: MAX_EPHEM = 3000   ! maximum number of ephemeris
```

- GEOS-CX record structure, define in the GEOSLIB package.

```
TYPE     S_OBSERVATION
     INTEGER         ::    SATID              ! satellite ID
     INTEGER         ::    TYPE               ! measurement type (23,73)
     INTEGER         ::    TIME_RECEIVE       ! ex: GRT=0 column 10
     INTEGER         ::    TIME_SYSTEM        ! ex: UTC=3 column 11
     REAL*8          ::    MJD                ! Modif. Julian Date UTC
     INTEGER         ::    STATID             ! Station number
     LOGICAL         ::    APPLY_IONOS_REFR   ! Ionospheric refraction
     LOGICAL         ::    APPLY_TROP_REFR    ! Troposph. refraction flag
     LOGICAL         ::    MET_DATA_FLAG      ! Flag for metereol.data
     REAL*8          ::    PRESSURE           ! Surface pressure [hP]
     REAL*8          ::    TEMPERATURE        ! Temperature [K]
     REAL*8          ::    HUMIDITY           ! Rel. humidity [%]
     LOGICAL         ::    TRP_DEL_FLAG       ! TRUE if correct transponder delay
     INTEGER         ::    SPEED_LIGHT        ! speed of light specif.
     INTEGER         ::    AMBIGUITY          ! Range ambiguity indicator ex:3
     INTEGER         ::    NVAL               ! # measurement values
     REAL*8          ::    VAL(2)             ! measurement value(s)
     REAL*8          ::    STD(2)             ! standard deviation of value(s)
     CHARACTER*30    ::    COMMENT            ! extension 120->150 bytes
END TYPE  S_OBSERVATION
```

- Structure containing epochs, first and second derivatives of the spline cubic interpolation, used by GENEPH and DIFFERENCE, and define in the SKSPEC package.

```
TYPE     S_EPHEM
  INTEGER            ::    N                  ! Number of data
  REAL*8             ::    MJD(MAX_EPHEM)     ! MJD epochs
  REAL*8             ::    NS(MAX_EPHEM,6)    ! Non-singular elmts
  REAL*8             ::    NSD(MAX_EPHEM,6)   ! spline 2nd derivatives
END TYPE  S_EPHEM
```

- New structure containing state vector and corresponding epochs, used by SPLITS, GENEPH and EXTRACT in INTERPOL.

```
TYPE  S_EPHEM_POINT
        REAL*8          ::      MJD(MAX_EPHEM)
        REAL*8          ::      Y(MAX_EPHEM,6)      ! Interpolated state vector
END TYPE  S_EPHEM_POINT
```

- Structure define in TRACKLIB.

```
TYPE     S_STATION
        INTEGER                 ::      ID                  ! ID of station
        CHARACTER*7             ::      DATA_TYPE           ! Range or Azimuth-Elevation
        REAL*8                  ::      BIAS_MN(7)          ! Mean of bias
        REAL*8                  ::      BIAS_SG(7)          ! Stdev og bias
        INTEGER                 ::      N_ERR
        TYPE (S_ERR_TYPE)       ::      ERR(MAX_DATA)       ! interpol. errors
        INTEGER                 ::      N_RES
        TYPE (S_RES_TYPE)       ::      RES(MAX_DATA)       ! residuals
END TYPE  S_STATION
```

- Structure used in the TRACK_S0, and define in TRACKLIB.

```
TYPE     S_TRACKING
        REAL*8                  ::      TIMESTEP
        REAL*8                  ::      MIN_MJD
        REAL*8                  ::      MAX_MJD
        INTEGER                 ::      N_STATIONS
        TYPE (S_STATION)        ::      STAT(MAX_STATIONS)
END TYPE  S_TRACKING
```

- New structure of TRACK_S0.

```
TYPE S_SCHEDULE
        INTEGER                 ::      N
        REAL*8                  ::      MJD(MAX_EPHEM)
        INTEGER                 ::      STATID(MAX_EPHEM)
        INTEGER                 ::      TYPE(MAX_EPHEM)
END TYPE S_SCHEDULE
```

- New structure of TRACK_S0

```
TYPE S_SCHEDULE_INFO
        INTEGER                 ::      NSTATION
        REAL*8                  ::      PERIOD
        INTEGER                 ::      STATID(MAX_STATIONS)
        REAL*8                  ::      OFFSET(MAX_STATIONS)
        INTEGER                 ::      NSAMPLE(MAX_STATIONS)
        REAL*8                  ::      CYCLE_DURATION(MAX_STATIONS)
END TYPE  S_SCHEDULE_INFO
```

A.1.2. Procedure, programs, subroutines specifications

In this paragraph the global process, its three STAGE A, B and C and their different programs are presented.

```
!+*********************************************************************************
! MCSIM.COM
!
!       The global process is based on the three main stages A,B and C. Their logic is described here,
!       and corresponds to the three "running phases" described in the introduction.
!
!     Logic :  1. Stage_A When real trilateration tracking data are available, the process uses it to
!              create a reference ephemeris file containing a dense LS Fit of the real data. When no
!              real Data are available, it uses an initial reference state vector and propagates it to
!              form the Dense reference file. The programs involved in this part are READER and
!              ORBIT. The process uses the dense reference file to create an interpolation table
!              containing the second derivatives of the spline cubic interpolation functions of these
!              dense points. The program involved in this part is INTERPOL. This beginning part
!              also create a file containing error free simulated tracking data with the real tracking
!              data when they exist, or with a reference station and a schedule file in the other case.
!              The programs involved in this part are TRACK_S0 and TRACK_S1 (this last one is
!              used when there is real data).
!              2. Stage_B The process then uses the artificial error free tracking data and an error
!              model to create a simulated tracking file. The program involved in this part is
!              TRACK_S2.This file is treated by the LS and the EKF methods to create estimation
!              files versus time. The programs involved in this part are SPLITS, ORBIT (second
!              application) and KALMAN.
!              3. Stage_C The estimation times are used to construct two reference interpolated
!              files with the interpolation table. The process finally computes the differences
!              between the corresponding estimations and interpolated files. The two program
!              involved in this part are DIFFERENCE and SELECT.
!*********************************************************************************
```

```
!+*********************************************************************************
! STAGE_A
!
! Logic : This part of the process run five programs :
!        - READER first transform the reference set-up REF_SETUP into the appropriate one
!          ORBIT_SETUP for the first application of ORBIT, which produces the dense reference
!          ephemeris REF.OUT , the error free simulated tracking data file STAGE2_GEOSCX2 and
!          the reference ephemeris residual data file ORBIT_RESIDUAL_FILE [1]
!        - The program INTERPOL builds interpolation table TABLE with this dense reference file.
!        - When no real data are available, TRACK_S0 reads the STAGE2_GEOSCX_2, the file
!          SCHEDULE and a REFERENCE _STATION_FILE to produce the STAGE2_GEOSCX
!          and the missing STATIONS_FILE. Due to the lack of ORBIT_RESIDUAL_FILE,
!          TRACK_S1 is not used.
!        - When real data are available, the program TRACK_S0 also only reads the
!          STAGE2_GEOSCX_2 and produces the STAGE2_GEOSCX. The program TRACK_S1
!          reads the ORBIT_RESIDUAL_FILE to produce the STATIONS_FILE needed in
!          TRACK_S2
!*********************************************************************************
```

[1] When real data are available.

```
!+**************************************************************************
! PROGRAM READER.F90
!
! Purpose : To have a correct set-up in the first application of ORBIT, it reads the reference set-up and
!           makes a new one containing the same information except the OD part which is not
!           necessary when no real tracking data are available.
!
!       Input file : REF.SETUP    Output file : ORBIT.SETUP
!       J.Ph. Halain   31/08/97
!---------------------------------------------------------------------------
PROGRAM READER

CONTAINS
!+--------------------------------------------------------------------------
! GET_WORD
!
! Returns first word (contiguous non-blank characters) of input line, starting search at specified column.
! Stops on error.
!  P. Francken 16/12/93, Modified arguments list and logic (COL; N now output) on 24/02/95
!---------------------------------------------------------------------------
SUBROUTINE GET_WORD( LINE, COL, FNAME, N, WORD )

!------------------------------- ARGUMENTS ----------------------------

  CHARACTER*132         , INTENT(IN)       :: LINE      ! Data line
  INTEGER               , INTENT(INOUT)    :: COL       ! Start column
  CHARACTER*(*)         , INTENT(IN)       :: FNAME     ! File name
  INTEGER               , INTENT(OUT)      :: N         ! Word length
  CHARACTER*(*)         , INTENT(OUT)      :: WORD      ! Word
!+--------------------------------------------------------------------------
! GET_LINE
!
! Reads data file, skipping blank lines and comment lines marked by a percent (%) sign, and returns
!  either the next data line. Stops on error.
! P. Francken 16/12/93
!---------------------------------------------------------------------------
SUBROUTINE GET_LINE( LU, FNAME, LINE, EOF )

!------------------------------- ARGUMENTS ----------------------------

   INTEGER              , INTENT(IN)       ::  LU       ! Logical unit
   CHARACTER*(*)        , INTENT(IN)       ::  FNAME    ! File name
   CHARACTER*132        , INTENT(OUT)      ::  LINE     ! Data line
   LOGICAL              , INTENT(OUT)      ::  EOF      ! End of file flag
!+**************************************************************************
! PROGRAM ORBIT.ADA (first use)
!
! Purpose : Propagates the initial reference state vector given in the setup_file to form a dense
!           reference ephemeris when no real data are available, and produces the Stage2_Geoscx_2
!           REF.ATD. If real tracking data are available (in GCX format), it makes an OD before this
!           propagation, following the general set-up, and the Orbit_Residual_File REF.RES is then
!           created in addition to the Stage2_Geoscx_2. Time step of propagation is reduced to
!           produce the dense ephemeris
!
!       Main subroutines : ORBIT_OP
!       Main input file : ORBIT.SETUP    Main output files : REF.ATD, REF.OUT, REF.RES
!---------------------------------------------------------------------------
```

```
!+*****************************************************************************
!  PROGRAM INTERPOL.F90
!
!  Purpose : Reads the Dense Reference Ephemeris (extracts out of the REF.OUT file, and transforms
!            into equinoctial elements), compute and store the second derivatives of the cubic spline
!            interpolate function of these points into the SPLINE.TBL file.
!            N.B. Uses equinoctial elements (the cubic spline interpolation is then more accurate).
!
!        Subroutines : EXTRACT, GENEPH
!        Input file : REF.OUT       Output file : SPLINE.TBL
!        J.Ph. Halain   31/08/97
!-------------------------------------------------------------------------------
PROGRAM INTERPOL

  USE GENERAL,  PANLIB, SKSPEC
  CONTAINS
!+------------------------------------------------------------------------------
! EXTRACT
!
! Purpose : extracts out of the REF.OUT file the dense reference ephemeris,
!           calls it POINTS, and NPTS the number of these ephemeris points
! 97/11/04   J.Ph. Halain
!-------------------------------------------------------------------------------
SUBROUTINE EXTRACT(FILE,NPTS,POINT)

!--------------------------------- ARGUMENTS ------------------------------

   INTEGER                    , INTENT(IN)        :: FILE
   INTEGER                    , INTENT(OUT)       :: NPTS
   TYPE (S_EPHEM_POINT)       , INTENT(OUT)       :: POINT(MAX_EPHEM)

!+------------------------------------------------------------------------------
! GENEPH        97/11/04   J.Ph. Halain
!-------------------------------------------------------------------------------
SUBROUTINE GENEPH(NPTS,POINT,EPH)

!---------------------------ARGUMENTS-----------------------------------

  INTEGER                    , INTENT(IN)         :: NPTS
  TYPE(S_EPHEM_POINT)        , INTENT(IN)         :: POINT(MAX_EPHEM)
  TYPE(S_EPHEM)              , INTENT(OUT)        :: EPH

!+*****************************************************************************
!  PROGRAM TRACK_S0.F90
!
!  Purpose : This program run when real tracking data are or are not available. In the first case, it only
!            convert the error free tracking data Stage2_Geoscx_2 REF.ATD file coming from the OD
!            of ORBIT into the Stage2_Geoscx REF2.ATD containing error free tracking data at times
!            of the Real Tracking Data REAL.GCX file. In the other case, it uses the Schedule to create
!            a file containing corresponding measurement times and convert Stage2_Geoscx_2 into the
!            Stage2_Geoscx with them. It also creates the missing STATIONS.DAT.
!
!    Subroutines : GET_SCHEDULE, GEN_SCHEDULE, READ_STATION
!                  WRITE_STATIONS, SELECT_REC, DECODE_LINE,ADD_STATIONS
!    Input files : REF.ATD, REF_STATION.DAT, SCH.DAT, REAL.GCX
!    Output files : REF2.ATD, STATIONS.DAT
!    J.Ph. Halain  23/11/97
!-------------------------------------------------------------------------------
```

```
PROGRAM TRACK_S0

   USE PANLIB, GEOSLIB, TRACKLIB, GENERAL
   CONTAINS

!+-------------------------------------------------------------------------------------------
! GET_SCHEDULE
! 97/11/04   J.Ph. Halain
!-------------------------------------------------------------------------------------------
  SUBROUTINE GET_SCHEDULE(LU,TRACK_SCHEDULE)

!-------------------------- ARGUMENTS ----------------------------------

    INTEGER                          , INTENT(IN)          :: LU
    TYPE ( S_SCHEDULE )              , INTENT(OUT)         :: TRACK_SCHEDULE

!+-------------------------------------------------------------------------------------------
! GEN_SCHEDULE
! 97/11/04   J.Ph. Halain
!-------------------------------------------------------------------------------------------
SUBROUTINE GEN_SCHEDULE(SCHEDULE,MJD_I,MJD_F,TRACK_SCHEDULE)

!-------------------------- ARGUMENTS ----------------------------------

    TYPE ( S_SCHEDULE_INFO )         , INTENT(IN)          :: SCHEDULE
    REAL*8                           , INTENT(IN)          :: MJD_I, MJD_F
    TYPE ( S_SCHEDULE )              , INTENT(OUT)         :: TRACK_SCHEDULE

!+-------------------------------------------------------------------------------------------
! SELECT_REC
!
!  Purpose : Select closest GCX record that matches station id and measurement type
!  97/11/04   J.Ph. Halain
!-------------------------------------------------------------------------------------------
  SUBROUTINE SELECT_REC(GCX_REC,N,MJD,STATID,TYPE,REC_ID)

!--------------------------- ARGUMENTS --------------------------------

   TYPE (S_OBSERVATION)     , INTENT(IN)          :: GCX_REC(MAX_EPHEM)
   INTEGER                  , INTENT(IN)          :: N
   REAL*8                   , INTENT(IN)          :: MJD
   INTEGER                  , INTENT(IN)          :: STATID
   INTEGER                  , INTENT(IN)          :: TYPE
   INTEGER                  , INTENT(OUT)         :: REC_ID


!+-------------------------------------------------------------------------------------------
! ADD_STATIONS
!
! 97/11/04   J.Ph. Halain
!-------------------------------------------------------------------------------------------
SUBROUTINE ADD_STATIONS (SCHEDULE,MJDI,MJDF, TRACK )

!--------------------------- ARGUMENTS --------------------------------

   TYPE ( S_SCHEDULE_INFO ) , INTENT(IN)          :: SCHEDULE
   REAL*8                   , INTENT(IN)          :: MJDI, MJDF
   TYPE (S_TRACKING)        , INTENT(INOUT)       :: TRACK
```

```
!+*****************************************************************************
! PROGRAM TRACK_S1.F90
!
! Purpose : Second part of the TRACKING_DATA_SIMULATOR program, used when real data are
!           available. Reads the ORBIT_RESIDUAL_FILE containing residuals of the OD of ORBIT,
!           and makes the STATIONS_FILE which contain fitted residuals of this reference OD.
!
!        Input files : REF.RES
!        Output files : STATIONS.DAT
!-------------------------------------------------------------------------------------------


!+*****************************************************************************
! STAGE_B
!
! Logic : This part of the process run four programs.
!        - The TRACK_S2 is used to produce Simulated Tracking Data from the Stage2_Geoscx_2,
!          the Stations_File and an Error Model file which contains modelled measurement error.
!        - The second application of ORBIT and the application of KALMAN realise the two kind of
!          estimations versus time of the GCX simulated data. The ORBIT program is run eight times,
!          using the eight GCX Simulated_Tracking_Data realised by the program SPLITS from the
!          global ORBIT_GEOSCX file. This is done to have eight estimations for the Least Squares
!          method at different chosen epochs. We then have one file of successive estimations versus
!          time for the EKF method EKF_SV.KF and one containing the eight estimations for the Least
!          Squares method  LSQ_SV.LQ.
!
!*****************************************************************************

!+*****************************************************************************
! PROGRAM TRACK_S2.F90
!
! Purpose : Third part of the TRACKING_DATA_SIMULATOR program.
!           Reads the Stations_File STATIONS.DAT, the Stage2_Geoscx REF2.ATD file which
!           contain error-free reference tracking data for each station, and an error model file
!           SIM.MOD. It then makes the simulated tracking data file (used as real measurements by
!           the two procedures ORBIT_2 and KALMAN) SIM.GCX.
!
!        Input files : REF2.ATD, SIM.MOD, SIM.STAT
!        Output files : SIM.GCX
!-------------------------------------------------------------------------------------------

!+*****************************************************************************
! PROGRAM SPLITS.F90
!
! Purpose :  Split the Input GCX file into 8 GCX files containing data during 8 different epoch
intervals
!           beginning each one at the first epoch of the Input file.
!           N.B.  This program is run eight times in the command file.
!
!        Subroutine : DECODE_LINE
!        Input file : INPUT.GCX
!        Output file : SIMULATED_TRACK_DATA_I.GCX  (I=1,8)
!
!        J.Ph. Halain    31/08/97
!-------------------------------------------------------------------------------------------
PROGRAM SPLITS

  USE GENERAL, PANLIB, GEOSLIB
```

```
!+*******************************************************************************
! PROGRAM ORBIT.ADA (second application)
!
! Purpose : Makes an OD (Least Squares) of each of the eight GCX files produced by SPLITS and
!           propagates to the last epoch of the file. It then creates an estimation at the last epoch of
!           each of these GCX files and appends them in the LSQ_SV.LQ file, using the subroutine
!           WRITE_LAST_SV_LQ.
!
!         Subroutines : ORBIT_OD, ORBIT_OP
!         Input files : ORBIT.SETUP, SIM.ATD
!         Output file : LSQ_SV.LQ
!---------------------------------------------------------------------------------------------

---------------------------------------------------------------------------------------------
-- SUBROUTINE WRITE_LAST_SV_LQ.ADA
--
-- Purpose : Some changes has been done in the Dump-Ephemeris part of the ORBIT_OP, because we
--           do not realise the differences between reference orbit and EKF or Least Squares
--           estimations directly in this package. It was necessary to transform it into the
--           WRITE_LAST_SV_LQ program which only computes and stores in the LSQ_SV.LQ the
--           Least Squares estimation at the last epoch of simulated tracking data. This new subroutine
--           is therefore include in the ORBIT_OD package of ORBIT. The SETUP package has also
--           been changed in consequence to not include the control flag used to know which
--           difference was done.
--        97/08/31  J.Ph. Halain
---------------------------------------------------------------------------------------------

procedure WRITE_LAST_SV_LQ ( MJD_EPOCH : MJD_TYPE;         --  Epoch MJD UTC, TT
                             MJD_LAST_OBS : MJD_TYPE;--  Last observation TOD,GR
                             Y_EPOCH  : VECTOR_6D;         --  State vector EME2000
                             CR : in LONG_FLOAT;           --  Radiation pressure coeff.
                             CR_P : in LONG_FLOAT;         --  Radiation pressure coeff.
                             CD : in LONG_FLOAT;           --  Drag coefficient
                             SC_PARAMS : in SC_PARAM_RECORD ) – Spac. parameter

!+*******************************************************************************
! PROGRAM KALMAN.ADA
!
! Purpose : Uses the Tracking Data Simulator GCX file to make orbit determination (EKF) at
!           successive epochs, and to produce the EKF_SV.KF file.
!
!        Subroutines : ORBIT_OD_KALMAN, ORBIT_OP
!        Input files : ORBIT_SETUP, SIM.ATD
!        Output files: EKF_SV.KF
!---------------------------------------------------------------------------------------------

!+*******************************************************************************
! STAGE_C
!
! Logic : This last part of the process first run program DIFFERENCE which uses the subroutine
!         INTERPOLATE. This subroutine builds, from the Interpolation Table, two reference files
!         state vectors at times corresponding to those of LS and EKF estimations. The program
!         then makes the differences between these files and the two corresponding estimations files
!         LSQ_SV.LQ and EKF_SV.KF and store it in the two DIF.LQ and DIF.KF files with the
!         interpolated vectors. The program SELECT then select from the EKF file created by
!         DIFFERENCE the eight differences and interpolated vectors which correspond to the 8
!         epochs of the Least Squares difference file.
!*******************************************************************************
```

```
!+*****************************************************************************
!  PROGRAM DIFFERENCE.F90
!
!  Purpose : Read the SPLINE.TABLE file created by INTERPOL and uses the different time points
!            of Least Squares and EKF OD to make Cartesian reference ephemeris vectors at these
!            corresponding epochs. It then makes the differences between them and the corresponding
!            estimated state vector of the two LS and EKF procedures.
!
!         Subroutines : INTERPOLATE
!         Input files : SPLINE.TBL, LSQ_EQ.LQ, EKF_EQ.KF
!         Output files : DIF_'SIMUL'.LQ, DIFF_'SIMUL'.KF
!
!         J.Ph. Halain    31/08/97
!-------------------------------------------------------------------------------
PROGRAM DIFFERENCE

  USE GENERAL, SKSPEC, PANLIB

!+*****************************************************************************
!  PROGRAM SELECT.F90
!
!  Purpose :  Selects the eight EKF differences and interpolated vectors at its eight epochs which
!             correspond to LS epochs.
!
!         Input files : DIFF_'SIMUL'.KF              Output files : DIF_'SIMUL'.KF
!         J.Ph. Halain    31/08/97
!-------------------------------------------------------------------------------
```

A.2. The MCSIM Command File

```
$ !+*************************************************************************
$ ! MCSIM.COM  (Monte Carlo Simulator)
$ !**************************************************************************
$ !
$ FLAG = 2
$ NUM = 10
$ CHOICE = 6
$ XXX[0,3] := 010
$ YY[0,2] := 00
$ Z[0,1] := 1
$ R[0,1] := 1
$ !
$  NUMBER1[0,1]:=3
$  NUMBER2[0,1]:=6
$  NUMBER3[0,1]:=9
$  NUMBER4[0,2]:=12
$  NUMBER5[0,2]:=18
$  NUMBER6[0,2]:=24
$  NUMBER7[0,2]:=36
$  NUMBER8[0,2]:=48
$ !
$ MODE = F$MODE()
$ SAY           = "WRITE SYS$OUTPUT"
$ CLEAR_SCREEN   = "ESC" + "[2J"   + "ESC" + "[?61"
$ POSITIVE       = "ESC" + "[27m" + "ESC" + "[1A"
$ NEGATIVE       = "ESC" + "[7m"  + "ESC" + "[1A"
$    IF (MODE.EQS."INTERACTIVE")
$    THEN
$        CALL HEADER
$        INQUIRE/NOPUNCT FLAG "     Real tracking data (1) or not (2) ?"
$        IF FLAG.NES."1" .AND. FLAG.NES."2"
$            THEN
$                SAY "     Choice not correct : end of the Simulation process"
$                EXIT
$        ENDIF
$        START :
$        CALL PRESENTATION
$        INQUIRE/NOPUNCT CHOICE "     Your choice ?"
$        IF CHOICE.EQS."0"
$            THEN
$               INQUIRE/NOPUNCT YY "     Setup and Schedule number yy ?"
$               INQUIRE/NOPUNCT Z   "     Model number z ?"
$               IF Z.EQS."0"
$                THEN
$                    R[0,1] := 0
$               ENDIF
$               IF Z.NES."0"
$                THEN
$                    R[0,1] := 1
$               ENDIF
$               GOTO START
$        ENDIF
$        IF CHOICE.EQS."1"
$            THEN
$               CALL EDIT_CONFIG
$               GOTO START
$        ENDIF
$        IF CHOICE.EQS."2"
$            THEN
$               CALL STAGE_A
$               CALL STAGE_B
$               CALL STAGE_C
$               GOTO START
$        ENDIF
$        IF CHOICE.EQS."3"
$            THEN
$               CALL STAGE_A
$               GOTO START
$        ENDIF
$        IF CHOICE.EQS."4"
$            THEN
$               CALL STAGE_B
$               GOTO START
$        ENDIF
$        IF CHOICE.EQS."5"
$            THEN
$               CALL STAGE_C
$               GOTO START
$        ENDIF
$        IF CHOICE.EQS."6"
$            THEN
$               XXX[0,3] := 'NUM'
$               CALL MONTECARLO
$               GOTO START
$       ENDIF
$       IF CHOICE.EQS."7"
$          THEN
$             SAY "    End of the Simulation process"
$             EXIT
$       ENDIF
$       IF CHOICE.NES."0".AND.CHOICE.NES."1".AND.CHOICE.NES."2"
          .AND.CHOICE.NES."3".AND.CHOICE.NES."4".AND.CHOICE.NES."5"
          .AND.CHOICE.NES."6".AND.CHOICE.NES."7"
$          THEN
$             SAY "    Choice not correct : end of the Simulation process"
$             EXIT
$       ENDIF
$       SAY ""
$       CALL MESSAGE "    Invalid option..."
$       WAIT 00:00:01
$       EXIT
$    ELSE
$       CALL MONTECARLO
$    ENDIF
$ !
$ !+***********************************************************************
$ ! SUBROUTINE HEADER
$ !
$ ! Purpose : clean screen and print system baneer
$ !************************************************************************
$ HEADER : SUBROUTINE
$ !
$ SAY CLEAR_SCREEN
$ SAY NEGATIVE
$ SAY ""
$ SAY "        Comparison of Least Squares Fit and Kalman Filtering         "
$ SAY "            when trilateration tracking is used                     "
$ SAY ""
$ SAY POSITIVE
$ !
$ ENDSUBROUTINE
$ !
$ !+***********************************************************************
$ ! SUBROUTINE PRESENTATION
$ !************************************************************************
$ PRESENTATION: SUBROUTINE
$ !
$    CALL HEADER
$    say ""
$    SAY "    Main Menu"
$    SAY ""
$    SAY "    0. Choice of configuration file indexes"
$    SAY "    1. Edit configuration files"
$    SAY "    2. Run one time the process"
$    SAY "    3. Run only the STAGE_A of the process"
$    SAY "    4. Run only the STAGE_B of the process"
$    SAY "    5. Run only the STAGE_C of the process"
$    SAY "    6. Monte Carlo running of the process"
$    SAY "    7. End of the procedure SIMULATOR"
$    SAY ""
$ !
$ ENDSUBROUTINE
$ !+**************************************************************************
$ ! SUBROUTINE EDIT_CONFIG
$ !**************************************************************************
$ !
$ EDIT_CONFIG : SUBROUTINE
$ !
$     DEFINE/NOLOG SYS$INPUT SYS$COMMAND
$     START1 :
$         CALL HEADER
$         SAY ""
$         SAY "     Database file edit menu"
$         SAY ""
$         SAY "     1. Edit SETUP file"
$         SAY "     2. Edit MODEL file"
$         SAY "     3. Edit SCHEDULE file"
$         SAY "     4. Edit REF_STATION file"
$         SAY "     5. Back to main menu"
$         SAY ""
$         INQUIRE/NOPUNCT CHOICE1 " Your choice ?"
$         IF CHOICE1.EQS."1"
$             THEN
$                 EVE [E.HALAIN.SETUP]STD_'YY'.SETUP
$                 GOTO START1
$         ENDIF
$         IF CHOICE1.EQS."2"
$             THEN
$                 EVE [E.HALAIN.SETUP]SIM_'Z'.MOD
$                 GOTO START1
$         ENDIF
$         IF CHOICE1.EQS."3"
$             THEN
$                 EVE [E.HALAIN.SETUP]SCH_'YY'.DAT
$                 GOTO START1
$         ENDIF
$         IF CHOICE1.EQS."4"
$             THEN
$                IF Z.EQS."0"
$                 THEN
$                     EVE [E.HALAIN.TAB]REF_STATIONS_0.DAT
$                     GOTO START1
$                ENDIF
$                IF Z.NES."0"
$                 THEN
$                     EVE [E.HALAIN.TAB]REF_STATIONS_1.DAT
$                     GOTO START1
$                ENDIF
$         ENDIF
$         IF CHOICE1.EQS."5"
$             THEN
$                EXIT
$         ENDIF
$         IF CHOICE1.NES."1".AND.CHOICE1.NES."2".AND.CHOICE1.NES."3"
            .AND.CHOICE1.NES."4".AND.CHOICE1.NES."5"
$             THEN
$                 CALL MESSAGE "Invalide option..."
$                 EXIT
$         ENDIF
$         SAY ""
$        CALL MESSAGE "Invalide option..."
$        GOTO START1
$ !
$ ENDSUBROUTINE
$ !
$ !+*************************************************************************
$ ! SUBROUTINE STAGE_A
$ !*************************************************************************
$ STAGE_A : SUBROUTINE
$ !-----------------------------------------------------------------------
$ !     Run program ORBIT
$ !-----------------------------------------------------------------------
$ !
$ SET DEF $DISK18:[E.HALAIN.COMPARE.ESSAI]
$ DEFINE IERS_TABLES              [E.HALAIN.TAB]NEWMAS.DAT            ! in
$ DEFINE STATION_FILE             [E.HALAIN.TAB]STATION.DAT           ! in
$ !
$ IF FLAG.EQS."1"
$     THEN
$     DEF/USER ORBIT_GEOSCX       [E.HALAIN.MEASUREMENT]TRILAT.GCX         ! in
$     DEF/USER ORBIT_SETUP        [E.HALAIN.SETUP]STD_'YY'.SETUP      ! in
$     DEF/USER ORBIT_RESIDUAL_FILE   REF.RES                          ! out
$     WRITE SYS$OUTPUT "Running reference OD and OP..."
$ ENDIF
$ !
$ IF FLAG.EQS."2"
$     THEN
$     DEF/USER STD_SETUP          [E.HALAIN.SETUP]STD_'YY'.SETUP      ! in
$     DEFINE   ORBIT_SETUP        ORBIT.SETUP                         ! out/in
$     RUN [E.HALAIN.F90.JPH]READER
$     WRITE SYS$OUTPUT "Running reference OP..."
$ ENDIF
$ !
$ DEF/USER ORBIT_DATABASE         REF.ELM                     ! out
$ DEF/USER ORBIT_ATD              REF.ATD                     ! out
$ DEF/USER ORBIT_SUMMARY                  REF.SUM                     ! out
$ DEF/USER ORBIT_SUMMARY2         REF.DBR                     ! out
$ DEF/USER ADA$OUTPUT             REF.OUT                     ! out
$ DEF/USER GRAPHIC_FILE           REF.PS                      ! out
$ !
$ DEF/USER ORBIT_STATE            REF"_SV"                    ! out
$ DEF/USER ORBIT_EQUINOCTIAL      REF"_EQ"                    ! out
$ !
$ RUN [E.HALAIN.ORBIT]ORBIT
$ !
$ !-----------------------------------------------------------------------
$ !    Run program INTERPOL
$ !-----------------------------------------------------------------------
$ DEF/USER POINT_IN               REF.OUT                     ! in
$ DEF/USER TABLE                  SPLINE.TBL                  ! out
$ !
$ RUN [E.HALAIN.F90.JPH]INTERPOL
$ !
$ !-----------------------------------------------------------------------
$ !    Run program TRACK_S0
$ !-----------------------------------------------------------------------
$ DEF/USER STAGE2_GEOSCX_2        REF.ATD                                  ! in
$ DEF/USER ORBIT_GEOSCX           [E.HALAIN.MEASUREMENT]TRILAT.GCX         ! in
$ DEF/USER SCHEDULE               [E.HALAIN.SETUP]SCH_'YY'.DAT              ! in
$ DEF/USER STATIONS_IN_FILE       [E.HALAIN.TAB]REF_STATIONS_'R'.DAT        ! in
$ DEF/USER STAGE2_GEOSCX          REF2.ATD                                  ! out
$ DEF/USER STATIONS_OUT_FILE      STATIONS.DAT                 ! out (if no real data)
$ !
$ RUN [E.HALAIN.F90.JPH]TRACK_S0
$ !
$ !-----------------------------------------------------------------
$ !    Run program TRACK_S1 (only if real data are available)
$ !-----------------------------------------------------------------
$ !DEASSIGN SYS$INPUT
$ IF FLAG.EQS."1"
$    THEN
$     DEF/USER ORBIT_FILE          REF.OUT                       ! in
$     DEF/USER STATIONS_OUT_FILE   STATIONS.DAT                  ! out
$     DEFINE SYS$INPUT             SYS$COMMAND                   ! in
$ !
$     RUN [E.HALAIN.F90]TRACK_S1
$ !    2
$ ENDIF
$ !
$ ENDSUBROUTINE
$ !
$ !+***********************************************************************
$ ! SUBROUTINE STAGE_B
$ !************************************************************************
$ STAGE_B : SUBROUTINE
$ !*----------------------------------------------------------------
$ !    Run program TRACK_S2
$ !-----------------------------------------------------------------
$ DEF/USER STAGE2_GEOSCX          REF2.ATD                         ! in
$ DEF/USER STATIONS_IN_FILE       STATIONS.DAT                     ! in
$ DEF/USER MODEL_FILE             [E.HALAIN.SETUP]SIM_'Z'.MOD      ! in
$ DEF/USER ORBIT_GEOSCX           SIM.GCX                          ! out
$ !
$ RUN [E.HALAIN.F90]TRACK_S2
$ !
$ !*----------------------------------------------------------------
$ !    Run the two estimation programs
$ !-----------------------------------------------------------------
$ SET DEF [E.HALAIN.COMPARE.ESSAI]
$ DEFINE IERS_TABLES              [E.HALAIN.TAB]NEWMAS.DAT         ! in
$ DEFINE STATION_FILE             [E.HALAIN.TAB]STATION.DAT        ! in
$ !
$ !*-------------------------------
$ !    Program KALMAN
$ !--------------------------------
$ DEFINE ORBIT_SETUP              [E.HALAIN.SETUP]STD_'YY'.SETUP   ! in
$ DEF/USER ORBIT_GEOSCX           SIM.GCX                          ! in
$ !
$ DEF/USER ORBIT_RESIDUAL_FILE    EKF.RES                          ! out
$ DEF/USER ORBIT_DATABASE         EKF.ELM                          ! out
$ DEF/USER ORBIT_ATD              EKF.ATD                          ! out
$ DEF/USER ORBIT_SUMMARY                  EKF.SUM                          ! out
$ DEF/USER ORBIT_SUMMARY2         EKF.DBR                          ! out
$ DEF/USER ADA$OUTPUT             EKF.OUT                          ! out
$ DEF/USER GRAPHIC_FILE           EKF.PS                           ! out
$ !
$ DEF/USER ORBIT_STATES          EKF"_SV"                              ! out
$ DEF/USER ORBIT_EQUINOCTIALS    EKF"_EQ"                              ! out
$ !
$ WRITE SYS$OUTPUT "Running Kalman OD and OP..."
$ RUN [E.HALAIN.ORBIT]KALMAN
$ !
$ !*-----------------------------------
$ !    Program ORBIT (and SPLITS)
$ !-----------------------------------
$ COPY SIM.GCX               INPUT.GCX
$ DEFINE  INPUT              INPUT.GCX                                 ! in
$ !
$   NRUN = 8
$   N=1
$ !
$   SC1:
$     NUMBER = NUMBER'N'
$     DEFINE ORBIT_GEOSCX   SIMULATED_TRACK_DATA_'NUMBER'.GCX   ! out/in
$     RUN [E.HALAIN.F90.JPH]SPLITS
$ !
$        DEF/USER ORBIT_RESIDUAL_FILE        LSQ.RES                   ! out
$        DEF/USER ORBIT_DATABASE             LSQ.ELM                   ! out
$        DEF/USER ORBIT_ATD                  LSQ.ATD                   ! out
$        DEF/USER ORBIT_SUMMARY              LSQ.SUM                   ! out
$        DEF/USER ORBIT_SUMMARY2             LSQ.DBR                   ! out
$        DEF/USER ADA$OUTPUT                 LSQ.OUT                   ! out
$        DEF/USER GRAPHIC_FILE               LSQ.PS                    ! out
$ !
$        DEF/USER ORBIT_STATE                LSQ"_SV"                  ! out
$        DEF/USER ORBIT_EQUINOCTIAL          LSQ"_EQ"                  ! out
$ !
$     WRITE SYS$OUTPUT "Running Least Squares OD and OP..."
$        RUN [E.HALAIN.ORBIT]ORBIT
$ !
$        IF N.EQS."1"
$           THEN
$                COPY LSQ_SV.LQ       LSQ_SV_T.LQ
$                COPY LSQ_EQ.LQ       LSQ_EQ_T.LQ
$              ELSE
$                APPEND LSQ_SV        LQ LSQ_SV_T.LQ
$                APPEND LSQ_EQ.LQ     LSQ_EQ_T.LQ
$        ENDIF
$        N=N+1
$ !
$        IF N.LE.NRUN
$           THEN
$                GOTO SC1
$        ENDIF
$ !
$   RENAME LSQ_SV_T.LQ      LSQ_SV.LQ
$   RENAME LSQ_EQ_T.LQ      LSQ_EQ.LQ
$ !
$ ENDSUBROUTINE
$ !
$ !+***********************************************************************
$ ! SUBROUTINE STAGE_C
$ !***********************************************************************
$ STAGE_C : SUBROUTINE
$ !*-----------------------------------------------------------------------
$ !     Run program DIFFERENCE
$ !-----------------------------------------------------------------------
$ SIMUL[0,1] := 'FLAG'
$ SIMUL[1,3] := 'XXX'
$ SIMUL[4,2] := 'YY'
$ SIMUL[6,2] := 'Z'
$ !
$ DEF/USER TABLE              SPLINE.TBL                    ! in
$ DEF/USER LQ_ESTIM           LSQ_SV.LQ                     ! in
$ DEF/USER KF_ESTIM           EKF_SV.KF                     ! in
$ DEF/USER LQ_DIFF            DIFF_'SIMUL'.LQ               ! out
$ DEF/USER KF_DIFF            DIFF_'SIMUL'.KF               ! out
$!
$ RUN [E.HALAIN.F90.JPH]DIFFERENCE
$ !
$ ENDSUBROUTINE
$ !
$ !+**********************************************************************
$ ! SUBROUTINE MONTECARLO
$ !
$ !***********************************************************************
$ MONTECARLO : SUBROUTINE
$ !
$ SIMUL[0,1] := 'FLAG'
$ SIMUL[1,3] := 'XXX'
$ SIMUL[4,2] := 'YY'
$ SIMUL[6,2] := 'Z'
$ !
$ WRITE SYS$OUTPUT "Running Monte Carlo simulations..."
$ !
$       CALL STAGE_A
$ !
$ NRUNS = 1
$ SC2 :
$ !
$       CALL STAGE_B
$       CALL STAGE_C
$ !
$       DEF/USER DIFF        DIFF_'SIMUL'.LQ
$       DEF/USER DIF_MC      DIF_'SIMUL'.LQ
$       RUN [E.HALAIN.F90.JPH]SELECT
$ !
$       DEF/USER DIFF        DIFF_'SIMUL'.KF
$       DEF/USER DIF_MC      DIF_'SIMUL'.KF
$       RUN [E.HALAIN.F90.JPH]SELECT
$ !
$       IF NRUNS.EQS."1"
$          THEN
$               COPY DIFF_'SIMUL'.LQ          DIFF_MC_'SIMUL'.LQ
$               COPY DIFF_'SIMUL'.KF          DIFF_MC_'SIMUL'.KF
$               COPY DIF_'SIMUL'.LQ           DIF_MC_'SIMUL'.LQ
$               COPY DIF_'SIMUL'.KF           DIF_MC_'SIMUL'.KF
$       ELSE
$               APPEND DIFF_'SIMUL'.LQ        DIFF_MC_'SIMUL'.LQ
$               APPEND DIFF_'SIMUL'.KF        DIFF_MC_'SIMUL'.KF
$               APPEND DIF_'SIMUL'.LQ         DIF_MC_'SIMUL'.LQ
$               APPEND DIF_'SIMUL'.KF         DIF_MC_'SIMUL'.KF
$       ENDIF
$ !
$       NRUNS = NRUNS + 1
$ !
$       IF NRUNS.LE.NUM
$              THEN
$                     GOTO SC2
$       ENDIF
$ !
$ ENDSUBROUTINE
```

A.3. User manual

The MCSIM procedure is written in the MCSIM.COM file and is run with the @MCSIM command. To store the results in a readable way, the process writes them in the following form : *DIF_MC_'xxxyyzz'.LQ* and *DIFF_MC_'xxxyyzz'.KF*, where xxx indicates the number of runs of the process (< 100), yy indicates which set-up file is used and zz the model file. All these results are stored in the directory ANALYSIS and are treated by the post process presented in the next chapter.

This procedure may be run interactively or in 'batch queue'. When it is used interactively, the user may act in changing the different following inputs :

- The existence or not of real tracking data.
- The eight hours intervals used in the SPLITS program.
- The SETUP files, which are modified to realise different simulations. The user may chose which set-up is used in introducing the number yy. The same index is used for reference and standard set-ups, because in simulations they are used together.
- The MODEL file, which is modified to introduce different kind of error model. The user may chose which model is used in introducing the number zz.
- The SCHEDULE file, which is modified to take into account different kind of stations and their time intervals parameters. The user must of course chose the SCHEDULE corresponding to the set-up yy.
- The REFERENCE_STATIONS file which is chosen with null or no null values depending on the tested case, i.e. fitted or white additive Gaussian noise, this choice being made with the number zz as described in the Simulation Plan of the next chapter.

All these inputs have to be consistent to avoid problem in the process. It is then necessary to respect the following rules :

1. The whole process normally starts with real tracking data (in a GCX format), OD and OP are then made in the STAGE A and the TRACK_S1 forms the Stations_File. The number of stations and their ID must be same in the three input files and in the Real_Gcx data.

2. The process can run without such real trilateration data. It then begins with an initial state vector, and in this case OP is only needed in the STAGE A. The Stations_File is then created by the program TRACK_S0.
   - Here the number of stations and their ID must be the same in the three input files and in the Ref_Stations_File.
   - The Ref_Setup and Std_Setup have to be adapted to the number of data of the Ref_Stations_File : the "Propagate" part of the set-up must be done between a number of days inferior to the number of days of the reference stations file plus one.
   - The kind of data of the Ref_Stations_File must be the same as the one of the Stage2_Geoscx_2, that is range or azimuth (elevation).
   - The cycle duration of the Schedule must be inferior to the time step of the Ref_Setup and Std_Setup.
   - This time step must be reduced when one include more stations in the different input files to avoid having more than MAX_EPHEM data.
   - The initial standard deviation in set-ups must be different of zero. The ORBIT application indeed need a non zero initial matrix to run correctly.

Some other important points must also be tacking into account to avoid problems which may be not visible when the process is run, but which have great consequences on results obtained :

i) The KALMAN program cannot estimate parameters like CD, CP and station bias, in the used version of this application. It is then necessary to introduce in the Std_Setup fixed such parameters. It is then necessary to fist run the ORBIT application alone to estimate them.

ii) The initial date and state vector of the Ref and Std_Setup need not to be related to the first epoch of the real data or the reference stations file. But it is logical for this date not to be too different.

iii) The number of hours introduced for the program SPLITS must not exceed the number of days of the propagation part of the set-up.

iv) The number MAX_EPHEM, defined in the GENERAL.F90 program, must be sufficiently big to allow the use of large amount of dense points. It is fixed at 50.000. If t is the time step (in seconds) of the dense ephemeris and n the number of days of data, this number must be greater than $d\frac{24*3600}{t}$. For example, with two days of data and a time step of 60 seconds, $MAX\_EPHEM > 28800$.

v) The total number of estimated parameters (that is the six elements of the state vector plus others parameters) must be lower than the number of data of the Simulated_Track_Data files.

With two days of data, as it is the case in following simulations, it is not correct to estimate station bias. Indeed, even with eight days of real data, the bias of the four used stations cannot be correctly estimated. The simulations are then run with either no estimated bias or with only two estimated bias (Bergen and Roma, because the estimation of Seville Bias causes some divergence in results).

A.4. Test plan of the simulator

To verify the correctness and accuracy of the whole process and its different parts, tests were realised They were run in a systematic way to validate the process, following a *Test Plan*. It is presented in the Annexes.

The first part of this test plan consists in checking important new programs and subroutines through all possible cases of input data

New programs and subroutines	Test
READER	Compare input and output for different kind of set-up (containing too much or not enough information).
EXTRACT subroutine	Compare input and output in different situations of reference orbit propagation (density i.e.).
GET(GEN) SCHEDULE subroutines	Compare the two outputs run with real or no real data and see if they have the same structure, see if the epochs are the corresponding ones with respectively the GCX file and the Schedule. Try different kind of schedules or real GCX files (number of stations i.e.).
SELECT_REC subroutine	Compare the file STAGE2_GEOSCX with STAGE2_GEOSCX_2 and the time file created by GET(GEN)_SCHEDULE. Test the different cases of schedule or GCX data. The data in the STAGE2_GEOSCX must be correctly sorted to have increasing epochs (because the Kalman program cannot do backward integration).
READ_STATION / ADD_STATIONS / WRITE_STATIONS	See if the Stations_File is correctly created from the Ref_Station_File. Try different cases of epochs of the stage2_geoscx and ref_stations_file.
SPLITS	To test this subroutine, one has to compare its output and the initial GCX file, and have 8 outputs files containing the same data during increasing period. Test for different number of output file and various input GCX files.
WRITE_LAST_SV_LQ subroutine	See if the files are correctly created and contains a state vector and its corresponding epoch. Try it with the different outputs of splits program.
SELECT	Compare input and output.

The second phase of the test plan consists in checking the combinations of the different programs and the expected results. For these tests, the three stages of the MCSIM process are run separately[2] to check their own results. Thereafter, the whole process MCSIM is run to verify its correctness. The tests on these stages are presented in the following points.

1. To verify the construction of the interpolation table (program INTERPOL) in Stage A, and its use in the creation of the two reference ephemerides corresponding to the EKF and LS estimation (program DIFFERENCE) in Stage

[2] Stage A may be run alone. Nevertheless Stage B and Stage C must be run respectively with simulated tracking data (outputs of Stage A), interpolation table and estimation files (outputs of Stage A and Stage B) . It is then necessary to test then in the logical order A, B and C.

C, and compute the differences between the dense reference ephemeris and interpolation obtained with the interpolation table at times of the dense reference ephemeris. The resulting differences should be null.

2. To verify the accuracy of the construction of the Stations_File in Stage A, one has to compare the two programs TRACK_S0 and TRACK_S1 used without and with real tracking data. The data must be correctly sorted in order to avoid problem in KALMAN, because it cannot do backward integration.
3. To verify the accuracy of the construction of the simulated tracking data by the TRACK_S2 of the T.D.S. in the Stage B, one has to enter an error free model in TRACK_S2 and deactivate the random generator of noise. The simulated tracking data should then be the same as the input file STAGE2_GEOSCX.
4. Finally, to check the whole process, one runs different "free" simulations (error free model, Ref_Stations file reduced to only initial and final epoch, and set-up without noise and standard deviation) and verify that perfect null residuals are obtained following the theoretical propagation model.

Note :

- The two stages of the T.D.S. used here are supposed correct. The program KALMAN and the program ORBIT (except the ORBIT_OD part which has been changed with the Write_Last_Sv_Lq) are also supposed without any problems.
- It is necessary to use the ORBIT_OP part which do not contains additional errors, to avoid counting them two times (in ORBIT_OP and in the STAGE_S2).

A.5. Input files

A.5.1. Setup (Std_01)

```
TITLE
 OD_ORBIT

INIT
  EPOCH
        DATE 1997 12 12  HOUR  0  0  0.000
  KEPLER_ELEMENTS
        A  42166062.90   E  0.000298384   I     0.057977
        RAN  355.831363   AOP  320.011165   M    144.071342
 STATE_STDDEV
        10000.0 1000.0 10000.0 0.1 0.1 0.1

SATELLITE
  SAT_ID                         9103002
  MASS                           1405.90
  AREA_SOLRAD SUN_POINTING       44.20
  CR                             1.27150
  CR_P                           0.03640
  AREA_DRAG                      0.00
  CD                             2.30000
  TRP_DELAY                      0.00000
```

```
MODEL  % the following values define the force model used
 POTENTIAL_ORDER  2
 F_107             160.0
 F_BAR             160.0
 K_P               3.0
MANEUVER
END_MANEUVER

PROCESS_NOISE
 RADIAL      1.0e-9
 EAST        1.0e-9
 NORTH       1.0e-9

STATION
 NUMBER   8000                          % TRILAT MASTER BTZ
        TYPE
              S_RANGE   SIGMA   1.80  BIAS   2.5   ESTIMATE  FALSE  % [m]
        END_TYPE
 NUMBER   8001                          % TRILAT SEVILLA
        TYPE
              S_RANGE   SIGMA   1.80  BIAS   0.0   ESTIMATE  false  % [m]
        END_TYPE
 NUMBER   8002                          % TRILAT ROMA
        TYPE
              S_RANGE   SIGMA   1.80  BIAS   0.0   ESTIMATE  false  % [m]
        END_TYPE
 NUMBER   8003                          % TRILAT BERGEN
        TYPE
              S_RANGE   SIGMA   1.80  BIAS   2.5   ESTIMATE  FALSE  % [m]
        END_TYPE
END_STATION

ORBIT_DETERMINATION
 ITERATIONS    7
 ESTIMATE
        CR            FALSE
        CR_P          FALSE
        CD            FALSE
 EDIT
        99.0                            % Initial editing level to be applied for n iteration(s)
        2                               % n
        2.0                             % Editing level for further iterations

ORBIT_PREDICTION
 EPHEMERIS
        FROM     DATE  1997 12 12  HOUR  0  0 00.0     % UTC
        TO     DATE  1997 12 14  HOUR  0  0 00.0       % UTC
        TIME_STEP  60.0                                % seconds
 OPTIONS
        KEPLERIAN
        CARTESIAN
        TRACKING_DATA
END_OPTIONS

PROPAGATE
 TO  DATE  1997  12 12  HOUR  00  00 00.0              % UTC

END
```

A.5.2. Error model (full trilateration model)

```
STATION 8000 RANGE_2
          0.0000   2.0000            constant term
          0.0000   0.1000            linear term
          0.0000   0.2000            periodic term
          1.5700   1.0000            phase
          0.5300   0.3700            ionosph.daily term
          0.0530   0.0370            ionosph.night term
          1.5700   0.2600            ionosph.phase
STATION 8001 RANGE_2
          0.0000   2.0000            constant term
          0.0000   0.1000            linear term
          0.0000   0.2000            periodic term
          1.5700   1.0000            phase
          0.4300   0.3000            ionosph.daily term
          0.0430   0.0300            ionosph.night term
          1.5700   0.2600            ionosph.phase
STATION 8002 RANGE_2
          0.0000   2.0000            constant term
          0.0000   0.1000            linear term
          0.0000   0.2000            periodic term
          1.5700   1.0000            phase
          0.4400   0.3100            ionosph.daily term
          0.0440   0.0310            ionosph.night term
          1.5700   0.2600            ionosph.phase
STATION 8003 RANGE_2
          0.0000   2.0000            constant term
          0.0000   0.1000            linear term
          0.0000   0.2000            periodic term
          1.5700   1.0000            phase
          0.7800   0.5600            ionosph.daily term
          0.0780   0.0560            ionosph.night term
          1.5700   0.2600            ionosph.phase
```

A.5.3. Schedule File (middle, and for four station)

```
4
3600.0000
8000    0000.0000       3        0150.0000
8001    0900.0000       3        0150.0000
8003    2700.0000       3        0150.0000
8002    1800.0000       3        0150.0000
```

A.5.4. Ref Stations File (with white Noise, and for 1 station)

```
3.D0
  50794.0088282060     50797.9350782754                  ! limit epochs
       1
     8000
RANGE_2
      2
  50794.0296615393     0.0000000       0.0000            ! mean - stdev
  50797.9059925579     0.0000000       0.0000            ! mean - stdev
```

A.6. Output files

A.6.1. Summary

```
ORBIT DETERMINATION SUMMARY                          31-MAR-1998  7:30

OD ORBIT

Tracking data summary
        From  1998/03/15  00:00:58
        To    1998/03/17  23:59:37

Station  Type  Unit  Sigma   N       Residuals          Bias
SA#3     Range  [m]    1.8   356    0.00 +/- 0.21       0.00
BTZ7     Azim   ["]   20.0   288    0.00 +/- 2.21    -144.05 +/- 1.39
BTZ7     Elev   ["]   20.0   288    0.00 +/- 2.36    -113.90 +/- 1.18

Satellite
        Sat_ID   9103002
        Mass     1400.20 kg
        Area(CR)   44.20 m**2    CR1  1.3490  (panels normal to equator)
                                 CRP  0.0180
        Area(CD)    0.00 m**2    CD   2.3000

Epoch
        Date  1998/03/14
        UTC   00:00:00.000000

State vector                       estimate                  correction
x [m]                   -41466271.39 +/- 38.79               -128.07
y [m]                   -7752980.81 +/- 73.81                 985.84
z [m]                   -20844.06 +/- 289.18                 -467.96
x-dot [m/s]             565.244000 +/- 0.002793              -0.064521
y-dot [m/s]             -3020.858503 +/- 0.002675            -0.016174
z-dot [m/s]             -0.259444 +/- 0.021236               -0.044908

Keplerian elements                 estimated                 correction
Semimajor axis [m]      42167814.63 +/-  0.50                0.53
Eccentricity            0.00042672 +/-  0.00000087           -0.00000063
Inclination [deg]       0.0287214 +/-  0.0003926             0.0007584
RA. asc. node [deg]     290.2934094 +/-  0.7901740           1.4704994
Arg. perigee [deg]      61.4293919 +/-  0.8927179            -1.7722141
mean anomaly [deg]      198.8833775 +/-  0.1172267           0.3005865

East longitude [deg]    19.1797207 +/-  0.0001016            -0.0013481

Convergence (4 Iterations)
        position 0.02 m
        velocity 0.00000 m/s
```

A.6.2. Residual and Output

Other main outputs are the residual and outputs files, additionally to the residual plots. Nevertheless these files are very large and their presentation is not necessary here.

A.7. Simulation plan tables

A.7.1. First group of simulations

The two simulations realised to check that the estimators are unbiased may be summarised in Table 1.

	Stations	Process Noise (a)	Initial Error (b)	Dynamic Model (c)
Std_00	4	Standard (2)	Standard (2)	Full Model (1)

		Error Model (d)	Gaussian Noise (e)	Station Combination (f)	Schedule (g)
Sim_0	Sch_1	None (2)	White (1)	4 stations (1)	Standard (1)
Sim_1	Sch_1	None (2)	Fitted (2)	4 stations (1)	Standard (1)

Table 1

A.7.2. Second group of simulations

The 36 simulations realised to characterise the EKF may be summarised in Table 2 and Table 3 for model and set-up files, with four and two stations. The 27 first one are realised with four stations and the 9 next ones with two stations.

	Stations	Process Noise (a)	Initial Error (b)	Dynamic Model (c)
Std_01	4	Standard (2)	Standard (2)	Full Model (1)
Std_02	4	Low (1)	Standard (2)	Full Model (1)
Std_03	4	High (3)	Standard (2)	Full Model (1)
Std_04	4	Standard (2)	Null (1)	Full Model (1)
Std_05	4	Low (1)	Null (1)	Full Model (1)
Std_06	4	High (3)	Null (1)	Full Model (1)
Std_07	4	Standard (2)	Large (3)	Full Model (1)
Std_08	4	Low (1)	Large (3)	Full Model (1)
Std_09	4	High (3)	Large (3)	Full Model (1)

		Error Model (d)	Gaussian Noise (e)	Station Combination (f)	Schedule (g)
Sim_2	Sch_1	Full (1)	Fitted (2)	4 stations (1)	Standard (1)
Sim_3	Sch_2	Full (1)	Fitted (2)	4 stations (1)	Beginning (2)
Sim_4	Sch_3	Full (1)	Fitted (2)	4 stations (1)	End (3)

Table 2

	Stations	Process Noise (a)	Initial Error (b)	Dynamic Model (c)
Std_10	2	Standard (2)	Standard (2)	Full Model (1)
Std_11	2	Low (1)	Standard (2)	Full Model (1)
Std_12	2	High (3)	Standard (2)	Full Model (1)
Std_13	2	Standard (2)	Null (1)	Full Model (1)
Std_14	2	Low (1)	Null (1)	Full Model (1)
Std_15	2	High (3)	Null (1)	Full Model (1)
Std_16	2	Standard (2)	Large (3)	Full Model (1)
Std_17	2	Low (1)	Large (3)	Full Model (1)
Std_18	2	High (3)	Large (3)	Full Model (1)

		Error Model (d)	Gaussian Noise (e)	Station Combination (f)	Schedule (g)
Sim_5	Sch_4	Full (1)	Fitted (2)	2 stations (2)	Standard (1)

Table 3

A.7.3. Third group of simulations

The six simulations realised to analyse the effect of the dynamic model on the EKF are summarised in Table 4.

	Stations	Process Noise (a)	Initial Error (b)	Dynamic Model (c)
Std_19	4	Standard (2)	Standard (2)	Simplified (2)
Std_20	4	Standard (2)	Standard (2)	Wrong Cr and Crp (3)
Std_21	4	Standard (2)	Standard (2)	Wrong SRFmodel (4)
Std_22	4	High (3)	Standard (2)	Simplified (2)
Std_23	4	High (3)	Standard (2)	Wrong Cr and Crp (3)
Std_24	4	High (3)	Standard (2)	Wrong SRFmodel (4)

		Error Model (d)	Gaussian Noise (e)	Station Combination (f)	Schedule (g)
Sim_6	Sch_1	Full (1)	Fitted (2)	4 stations (1)	Standard (1)

Table 4

A.7.4. Simulations with real data

The last eight simulations realised with real data are summarised in Table 5.

	Stations	Process Noise (a)	Initial Error (b)	Dynamic Model (c)
Std_25	3	Standard (2)	Standard (2)	Full Model (1)
Std_26	3	Standard (2)	Standard (2)	Wrong SRFmodel (2)
Std_27	3	High (3)	Standard (2)	Full Model (1)
Std_28	3	High (3)	Standard (2)	Wrong SRFmodel (2)

		Error Model (d)	Gaussian Noise (e)	Station Combination (f)	Schedule (g)
Sim_7	-	Full (1)	White (1)	4 stations (3)	-
Sim_8	-	Full (1)	Fitted (2)	4 stations (3)	-

Table 5

A.7.5. Set-up and model files characteristics

Table 6 contains the parameters values corresponding to the characteristics level of the set-up and model files used in the simulations as presented above in the different corresponding tables.

Process Noise (a)[3]	**Low (1)**	**Standard (2)**	**High (3)**
Q(t) $(m/s^2)^2$	$(10E\text{-}11)^2$	$(10E\text{-}9)^2$	$(10E\text{-}7)^2$
Initial Guess Error (b)	**Null (1)**	**Standard (2)**	**Large (3)**
Initial guess error (m m m m/s m/s m/s)	0 - 0 - 0 0 - 0 - 0	100 - 100 - 10 0.01 - 0.01 - 0.001	1000 - 1000 - 100 0.1 - 0.1 - 0.01
Initial state variance (m m m m/s m/s m/s)	100 - 100 - 100 0.01 - 0.01 - 0.01	10000 - 10000 - 10000 0.1 - 0.1 - 0.1	10E5 - 10E5 - 10E5 1 - 1 - 1

Dynamic Model (c)	**Full (1)**	**Simplified (2)**	**Wrong CR, CRP (3)**	**Wrong SRF Model (4)**
CR	1.2715	0	1.3	Estimated in reference set-up / 1.2715 in standard set-up
CRP	0.0364	0	0.03	Estimated in reference set-up / 0.0364 in standard set-up
Potential_order	10	3	10	10 in both set-ups

Error Model parameters (d)[4]	**Full (1) : Mean**	**Standard deviation**	**None (2) : Mean**	**Standard deviation**
C_1 (constant term)	0.0 m	2.0 m	0.0 m	0.0 m
C_2 (linear term)	0.0 m	0.1 m	0.0 m	0.0 m
C_3 (periodic term)	0.0 m	0.2 m	0.0 m	0.0 m
C_4 (phase)	1.57 rad	1.0 rad	0.0 rad	0.0 rad
C_5 (ion. Day term)	0.53 m	0.37 m	0.0 m	0.0 m
C_6 (ion. Night term)	0.053 m	0.037 m	0.0 m	0.0 m
C_7 (ion. Phase)	1.57 rad	0.26 rad	0.0 rad	0.0 rad

Gaussian Noise (e)	**White (1)**	**Fitted (2)**
Mean	0.8 m	Stations File when real data Ref_Stations File when no data
Standard deviation	3E-001 m	Stations File when real data Ref_Stations File when no data
Ref_Stations File	Ref_Stations_1	Ref_Stations_2
Time Step	Non applicable	2 hours

Station Combination (f)	**Four Stations (1)**	**Two Stations (2)**	**Four Stations (3)**
Index of stations	8000-8001-8002-8003	8000-8001	8000-8001-8002-8003
Status	• Betzdorf master • 3 other slaves	• Betzorf master • Seville slave	• Betzorf master • 3 others slaves
Time Step (in set-up)	120 s	60 s	90 s
Sigma (m)	1.8 - 1.8 - 1.8 - 1.8	1.8 - 1.8	1.8 - 1.8 - 1.8
Bias[5] (m)	0.0 - 0.0 for Betzdorf and Seville	0.0 - 0.0 for Betzdorf and Seville	0.0 - 0.0 for Betzdorf and Seville

[3] These values may be very small here because trilateration induce less EKF divergence. Furthermore, these small values do not disadvantage the EKF against LS.

[4] Theses values are given for Betzdorf. The other station have different ionospheric values due to the sin(el_i) term included in C_5 and C_6. The values are 0.53 - 0.43, 0.44 - 0.78, 0.37- 0.30and 0.31 - 0.56.

[5] The stations bias are fixed to zero for Betzdorf and Seville and estimated for the two other stations. Indeed with two days of data, 4 bias estimation is unrealistic and gives large erroneous values and differences results diverge.

Schedule (g)	**Middle (1)**	**Beginning (2)**	**End (3)**
Period (s)	3600	3600	3600
Number of stations	• 4 stations : Sch_1 • 2 stations : Sch_4	4 stations : Sch_2	4 stations : Sch_3
Offset (s)	• 0-900-1800-2700 • 0 - 1800	0 - 300 - 600 - 900	2400 - 2700 - 3000 - 3300
Number of samples	3	3	3
Cycle duration (s)	150	150	150

Table 6

A.8. Hierarchical design of the post process

A.8.1. Data structures of the F90 programs

```
INTEGER,PARAMETER         :: NDAYS = 2
INTEGER,PARAMETER         :: MAX_DIFF = 100       ! Maximum number of differences
REAL*8,PARAMETER          :: ALPHA_MAX = 0.5      ! Maximum probability for K-S test
REAL*8,PARAMETER          :: ALPHA = 0.05 ! Confidence level
INTEGER,PARAMETER         :: STEPSIZE = 100
REAL*8,PARAMETER          :: PSIREF= PSI/86400.D0
INTEGER,PARAMETER         :: KCLASS = 8           ! Number of classes

TYPE    S_METHOD
        REAL*8      ::    LQ
        REAL*8      ::    KF
END TYPE  S_METHOD

TYPE    S_COORD
        TYPE (S_METHOD)    ::    X
        TYPE (S_METHOD)    ::    Y
        TYPE (S_METHOD)    ::    Z
END TYPE  S_COORD
```

A.8.2. Subroutines and Programs Specifications

```
!+**********************************************************************
! MAXP.F90
!
! Purpose: This program treat the difference files LQ and KF coming from the simulator.
!          It reads the differences and the interpolated vectors at the eight epochs for the N
!          realisations and build the corresponding differences in the satellite axes.
!          It then propagates them over two days with the EULER_HILL subroutine.
!          It also transform the input differences into Keplerian elements.
!
! Input files  : DIF_MC_LQ, DIF_MC_KF
! Output files : DIF_MAXP_LQ, DIF_MAXP_KF, DIF_EQ_LQ, DIF_EQ_KF
!
! 98/02/16 J.Ph. Halain
!***********************************************************************
```

```
PROGRAM MAXP

   USE PANLIB
   IMPLICIT NONE

!+*********************************************************************
! STATISTICS.F90
!
! Purpose: This program treat the propagated difference LQ and KF as samples
!          of N realisations. It runs the subroutines G01ABF,G01AEF and G01ASF to
!          produce mean, variance, max and min of the two samples for the eight
!          epochs, and box and whisker point plots matrix for the eight epochs.
!
!  Input files  : DIF_MAXP_LQ, DIF_MAXP_KF
!  Output files : STAT_MAXP, FREQ_MAXP,PLOT_MAXP
!
!  98/02/16  J.Ph. Halain
!
!*********************************************************************

PROGRAM STATISTICS

   USE PANLIB
   IMPLICIT NONE

    REAL*8                  :: MJD,SV(6)
    INTEGER                 :: SCALE
    INTEGER                 :: i,j,k,l,N
    TYPE (S_COORD)          :: MAXP(MAX_DIFF,8)
    TYPE (S_COORD_REAL)     :: RES_MAXP(13,8)
    TYPE (S_COORD)          :: MAX_MAXP(8),MIN_MAXP(8),CINT_MAXP(KCLASS,8)
    TYPE (S_COORD_INT)      :: IFREQ_MAXP(KCLASS,8)

!----G01ABF subroutine-----------------

    TYPE (S_COORD)          :: ZZ(MAX_DIFF)           ! in
    REAL*8                  :: RES(13)                ! out
    INTEGER                 :: IWT, IFAIL1            ! in/out
    REAL*8                  :: WT(MAX_DIFF)           ! in/out

!----G01AEF subroutine-----------------

    INTEGER                 :: ICLASS                 ! in
    REAL*8                  :: XMIN, XMAX             ! out
    INTEGER                 :: IFREQ(KCLASS)          ! out
    INTEGER                 :: IFAIL2                 ! in/out
    REAL*8                  :: CINT(KCLASS)           ! in/out

!----G01ASF subroutine-----------------

    CHARACTER               :: PRT                    ! in
    INTEGER                 :: NM(M),LDX              ! in
    TYPE (S_COORD)          :: MAXPG(M,MAX_DIFF)      ! in
    TYPE (S_COORD)          :: WORK_MAXP(5*M)         ! out
    TYPE (S_COORD_CHAR)     :: PLOT_MAXP(LDP,NSTEPX)  ! out
    INTEGER                 :: IFAIL3                 ! in/out
    INTEGER                 :: LWORK(MAX_RUNS)
```

```
!+************************************************************************
! ANALYSIS.F90
!
! Purpose: Test normality of the population and compute its parameters with confidence
!           interval of mean, and determine if there are enough run. If population is normal, it
!          computes the probability value that the variable is lower than a fixed value.
!
! Input files  : DIF_MAXP_LQ, DIF_MAXP_KF
! Output files : POP_MAXP,PROBA_MAXP
!
! 98/01/15 J.Ph. Halain
!*************************************************************************
PROGRAM ANALYSIS

   USE PANLIB

   IMPLICIT NONE

   INTEGER                  :: i , j , k , l, N
   REAL*8                   :: MJD, SV(6),SV2(6)
   INTEGER                  :: SCALE
   TYPE (S_COORD)           :: MAXP(MAX_DIFF,8)                   ! in
   TYPE (S_COORD)           :: SX_MAXP(MAX_DIFF,8)
   TYPE (S_COORD)           :: PROBA_MAXP(8)                      ! out G08CGF
   TYPE (S_COORD_LOG)       :: NORM_MAXP,ENOUGH
   TYPE (S_COORD)           :: CONF_INT_MAXP(8),TAIL_MAXP(8)      ! out
   INTEGER                  :: IFAIL,MORE

!----G08CBF subroutine-----------------

   TYPE (S_COORD)           :: ZZ(MAX_DIFF)                       ! in
   CHARACTER*1              :: DIST,ESTIMA                        ! in
   INTEGER                  :: NTYPE                              ! in
   REAL*8                   :: SX(MAX_DIFF)                       ! out
   TYPE (S_COORD)           :: D_MAXP(8),Z_MAXP(8),p_MAXP(8)      ! out
   TYPE (S_COORD)           :: PAR(2)                             ! in/out
   INTEGER                  :: IFAIL1                             ! in/out

!----G08CGF subroutine-----------------

   INTEGER                  :: NPEST                                    ! in
   TYPE (S_METHOD_INT)      :: IFREQ_P(KCLASS), IFREQ_V(KCLASS),        &
                                  IFREQ_MAXP(KCLASS)                    ! in

   TYPE (S_METHOD)          :: CINT_P(KCLASS-1), CINT_V(KCLASS-1),
                                  CINT_MAXP(KCLASS-1)                   !in

   REAL*8                   :: PAR2(2),PROB(KCLASS)                     ! in
   CHARACTER*1              :: DIST                                     ! in
   REAL*8                   :: CHISQ,PROBA,CHISQI(KCLASS),EVAL(KCLASS)  ! out
   INTEGER                  :: NDF                                      ! out
   INTEGER                  :: IFAIL2                                   ! in/out!

 CONTAINS
```

```
+------------------------------------------------------------------------------------------------
! Subroutine POPULATION
!
! Purpose: Computes confidence intervals of population parameters with unknown
!           population variance (robust if not normal), and determine the number of runs
!           needed to reach a confidence interval corresponding to the fixed confidence level.
!------------------------------------------------------------------------------------------------
  SUBROUTINE POPULATION (NN,ALPHA,PAR,DIFF,CONF_INT,ENOUGH,MORE)

!------------------------- ARGUMENTS --------------------------------

    INTEGER, INTENT(IN)      :: NN,DIFF
    REAL*8,  INTENT(IN)      :: PAR(2),ALPHA
    REAL*8,  INTENT(OUT)     :: CONF_INT
    LOGICAL, INTENT(OUT)     :: ENOUGH
    INTEGER, INTENT(OUT)     :: MORE

!------------------------------------------------------------------------------------------------
! FUNCTION REAL*8 : T_CENTILE
!
! Purpose : Search the student centile corresponding to confidence level 1-alpha/2 by
!           inverting numerically the lower tail probability of the Student's t-distribution
!           versus centile t, for N-1 degree of freedom.
!------------------------------------------------------------------------------------------------
    REAL*8 FUNCTION T_CENTILE(N,Y0)

!------------------------- ARGUMENTS --------------------------------

    REAL*8 ,INTENT(IN)      :: Y0
    REAL*8 ,INTENT(IN)      :: N
```

Summary.

Performance of Kalman Filter versus Least Squares estimator for orbit determination using trilateration.

The purpose of this work is to compare the Least Squares and Kalman Filtering methods for orbit determination of geostationnary satellites, when tracking measurements are realised via a trilateration system.

For this purpose a simulator process has been developed. It compares the differences between a reference trajectory and estimations obtained with the two methods, for different observation and propagation models. Statistical analysis based on Monte Carlo simulations has been applied to compare the results of both methods, and determine if one method is better suited to the process of trilateration data.

This work has been realised for the *Société Européenne des Satellites* who has in charge 7 geostationnary satellites governed in real-time from Betzdorf (Luxembourg), whose 6 are co-located. Additionally, an article for the 13th International Symposium on Space Flight Dynamics, Goddard Space Flight Center, NASA, has been published. It extends the comparison of the present work to both single- and multi-stations tracking systems.

The Monte Carlo simulations performed in the framework of this Diploma Thesis have shown that there is a close agreement between estimation errors of both iterated Least Squares and extended Kalman Filter algorithms. Moreover, trilateration measurements, as compared to single-stations, seem to make both estimators less sensitive to the observation, dynamic and algorithmic input parameters. So, the Kalman Filter algorithm might offer an alternative to the usual Least Squares implementation, as for long correction manoeuvres where an important process noise is introduced.

Jean-Philippe Halain
3éme Technique Ingénieur Civil Physicien, 1997-1998
Faculté des Sciences Appliquées, Université de Liège

Société Européenne des Satellites
Space System Division
L-6815 Château de Betzdorf, Luxembourg

Bibliography.

[Ban 95] T. A. Banh : *Calcul des probabilités*; Notes de cours, Faculté des Sciences Appliquées, Université de Liège, 1995.

[Bra 93] Y. Bar-Shalom, X. R. Li : *Estimation and Tracking : Principles, Techniques and Software*; Artech House, Boston, 1993.

[Bie 77] G. J. Bierman : *Factorisation Methods for Discrete Sequential Estimations*; Acedemic Press, New York, 1977.

[Boz 79] S. M. Bozic : *Digital and Kalman Filtering*; Edward Arnold, University of Birmingham, 1979.

[Bry 75] A. E. Bryson, Y.-C. Ho : *Applied Optimal Control : Optimisation, Estimation and Control*; Hemisphere Publishing Corporation., New York, 1975.

[Clo 60] W. H. Clohessy, R. S. Wiltshire : *Terminal Guidance System for Satellite Rendez-Vous*; Journal of Aerospace Scientists, 653-656, 1960.

[Dag 92] P. Dagnelie : *Statistique Theorique et Appliquée*; Presses Agronomiques de Gembloux, 1992.

[Fra 92] P. Francken : *Analysis of Orbit Modelling Errors and their Impact on Orbital Elements*; SES internal report, Betzdorf, 1992.

[Fra 94] P. Francken : *Software Development Standards : Coding Standards*; SES internal report, Betzdorf, 1994.

[Fra 95] P. Francken : *Comparison of Orbit Interpolation Schemes*; SES Internal Notes, Betzdorf, 1995.

[Fra 96] P. Francken : *The Tracking Data Simulator (TRACKSIM) V 3.0 Detailed Software Description*; SES Internal Notes, Betzdorf, 1995.

[Fra 98] P. Francken : *Least Squares and Kalman Filtering techniques for orbit estimation*; SES Notes, Betzdorf, 1998.

[Gel 74] A. Gelb, editor : *Applied Optimal Estimation*; MIT Press, Cambridge, 1974.

[Giv 58] J. W. Givens : *Computation of plane unitary rotations transforming a general matrix to triangular form*; SIAM J.Appl.Math.6, 26-50,1958.

[Gol 70] G. H. Golub, C. Reinsch : *Singular Value Decomposition and Least Squares Solutions*; Handbook Series Linear Algebra, Numer.Math.14, 403-420, 1970

[Gre 93] M. S. Grewal, A. P. Andrews : *Kalman Filtering : Theory and Practice*; Prentice Hall, New Jersey, 1993.

[Hal 98] J. P. Halain, P. Francken, G. Krier, T. Welter, P. Waulthier and P. Rochus : *Performances of Least Squares and Kalman Filter Algorithms for Orbit Fetermination using Single- and Multi-Station Tracking of Geostationnary Satellites*; 13th International Symposium on Space Flight Dynamics, Goddard Space Flight Center, NASA, 1998.

[Jaz 70] A. H. Jazwinski : *Stochastic Processes and Filtering Theory*; Academic Press, New York, 1970.

[Kal 60] R. E. Kalman : *A New Approach To Linear Filtering And Prediction Problems*; ASME Journal of Basic Engineering, Serie D, Vol. 82, 1960.

[Kri 98] G. Krier : *Improvement of the ORBIT solar radiation pressure modelisation*; SES Internal Report, Betzdorf, 1998.

[Law 74] C. L. Lawson, R. J. Hanson : *Solving Least Squares Problems*; Prentice Hall, 1974.

[McC 96] D. D. McCarty (ed.) : *IERS Technical notes 21, IERS conventions*; Central Bureau of IERS, Observatoire de Paris, 1996.

[Mon 96] O. Montenbruck: *General Description of the Multi-Purpose Orbit Determination Program ORBIT*; DLR German Space Operations Centre Oberpfaffenhofen, 1996.

[Nag 90] *The Nag Fortran Library Guide, Mark 17*; The Numerical Algorithms Group, Oxford, 1995.

[Pap 91] A. Papoulis : *Probability, Random Variables and Stochastic Processes*; Mc Graw-Hill, 1991.

[Pre 92] W. H. Press, A. A. Teukolsky, W. T. Veterling, B. P. Flannery : *Numerical Recipes*; Cambridge University Press, Cambridge, 2nd edition, 1996.

[Roc 98] P. Rochus : *Mécanique Céleste*; Notes de Cours, 1998.

[Soo 94] E. M. Soop : *Handbook of Geostationnary Orbits*; Kluwer Academic Publishers, Dordrecht, 1994.

[Spi 61] M. R. Spiegel : *Theory and Problems of Statistics*; Schaum Publishing Company, New York, 1961.

[Wau 95] P. Wauthier : *Trilateration : SATRE Ranging Error Preliminary Analysis*; SES Internal Report, Betzdorf, 1995.

[Wei 97] T. Weidig : *The performances of the Kalman Filter versus the Least Squares Fit for trilateration*; SES internal report, Betzdorf, 1997

[Wel 97] T. Welter : *Development and Implementation of a ontinuous-discrete Kalman Filter for Dynamical Orbit Determination*; Diploma Thesis, Société Européenne des Satellites and Universität Karlsruhe, 1997

[Zar 87] O. Zarrouati : *Trajectoires spatiales*; Centre National d'Etudes Spatiales, 1987.