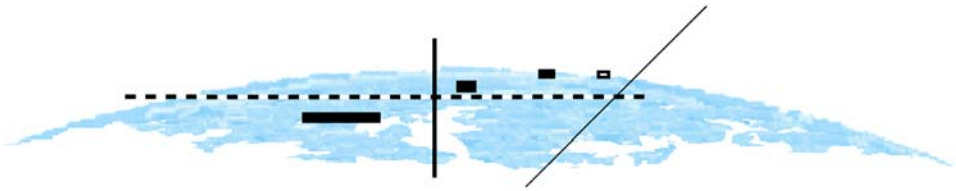


GeoForschungsZentrum Potsdam

STIFTUNG DES ÖFFENTLICHEN RECHTS

Scientific Technical Report

ISSN 1610-0956

MFGsoft

(Multi-Functional GPS/(Galileo) Software)

Software User Manual

Version of 2004

Guochang Xu

GeoForschungsZentrum Potsdam
Department 1: Geodesy and Remote Sensing
Telegrafenberg A17, 14473 Potsdam, Germany

October 2004

MFGsoft
(Multi-Functional GPS/(Galileo) Software)

Software User Manual

Contents

1. Introduction

MFGsoft, in his first version, is a real time multi-functional GPS software developed at the GeoForschungsZentrum Potsdam for following the development of the EU Galileo system and for simulating the measurement of the Galileo system. The goal is to develop a software which is able to process both the GPS and Galileo navigating and positioning data.

The so-called equivalent GPS data processing algorithm is used in this software, so that the un-differenced and differencing methods are unified into one algorithm and can be switched from one to another selectively. The software works in an sequential way which enables a real time epoch-block wise data processing or an one-step post-processing. To cover possibly application areas as more as possible, the navigating and positioning problem can be solved in ECEF coordinate system as soon as the orbit is considered as known. If orbit correction, dynamic orbit determination, or kinematic/dynamic combined orbit determination are required, the ECSF coordinate system is used. Data combinations are free selectable. A method of independent parameterisation of the equivalent observation model guarantees a regular normal equation so that the problem can be solved stably and the solution can be obtained homogeneously. A diagonalisation algorithm makes it possible to give up the parameters which are not any more actual in case of sequential data processing. Details of the functions and characters of the software are listed in the next section.

The documentation of the theoretical background of the applied algorithms are published in 2003 by Springer (Xu 2003). Descriptions are also carefully written in the source code. A small part of the source code and the features of the KSGSoft (Kinematic/Static GPS Software, Xu et al. 1998) are modified and absorbed in this new development.

The development of MFGsoft was started in 2002, first year for theoretical study and then for code design. The software is tested with many kinds of real data. Comparisons show good performance of the software. Due to the multi-functional abilities of the software, we hope it could be used in a broad area of applications and services for GPS and for simulation of the Galileo system in the future, and at the end it could be a basis of an excellent multi-functional GPS/Galileo software.

This manual outlines the characteristics and structure of the software and describes how to use the software. The principles and new features are outlined systematically and referred partly to existing references. Numerical examples of multi-functions and internal tests as well as external comparisons are given.

2. Characteristics of MFGsoft

The most important characteristics of MFGsoft are given as follows:

- MFGsoft is developed in C under Unix operating system and can be directly implemented on PCs under Linux without any change (computer independent).
- MFGsoft uses the equivalent GPS data processing algorithm. The un-differenced and differencing methods are free selectable by using switch.

- An independent parameterisation algorithm guarantees a regular normal equation (solvability). A diagonalisation algorithm makes it possible to keep the equations only with actual parameters (reduce the size of the problem).
- The software is able to be used for a real time data processing or post-processing modules.
- Broad application areas: can be used for data processing of local static/kinematic network, regional network monitoring, global network monitoring, and on board navigation.
- Multi-functions selectable: orbit as known, orbit has to be corrected, dynamic orbit determination, and kinematic/dynamic combined orbit determination.
- In case of orbit is known, solution in ECEF or ECSF coordinate system is selectable.
- Switches for all kinds of data combinations, all possible tropospheric models, IGS or broadcast ephemerides, cycle slips factor, gross error standard, debug functions.
- Switches for clock error solution or elimination, possible kinds of a priori information applications, fixed network, multi-reference stations, reference clock.
- Single point code positioning and differential Doppler velocity determination available by using phase and/or code measurements.
- Abilities to read or create the ephemerides of the Sun, the Moon and planets.
- Equipped with an extended software package of adjustment algorithms and filtering methods.
- Optimal ambiguity search criterion and algorithm.
- Equipped physical models: the Earth/ocean loading tide displacement, satellite mass centre correction, broadcast ionospheric model, relativity effects, solar radiation, atmospheric drag, etc.
- All possible orbit disturbance force models and orbit integration tools.
- The software is user-friendly designed for practical applications and scientific research.
- The software works under batch module and can run automatically.
- Many utilities programs.
- Detailed theoretical descriptions and references.
- User manual.

3. Run of MFGsoft

To start the MFGsoft, one just needs to enter the following command line:

MFGsoft <Input_parameter_file>

where Input_parameter_file is a file name in which consists all necessary information and control parameters for running the software. The Input_parameter_file has to be edited before running the program.

The definitions and descriptions of the Input_parameter_file will be given in the next section. The standard input parameter file may be created automatically.

4. Input Parameter File

The essential work to run the program for GPS data processing is to write an input parameter file which is defined in a flexible form. Definitions of the input parameter file, and standard input parameters as well as standard definitions of debug switches are given in following subsections.

4.1 Definition of Input Parameter Files

The detailed definitions of the input parameter file for the GPS data processing are given below in three parts: definitions of input parameter file, definitions of input parameters, and definitions of debug switches.

Definitions of the input parameter file:

- Any line with a character * (star) will be considered as a comment line.
- Every non-comment line should contain the character : (colon).
- The input contents are placed before the character : and the identifier of the input is placed after the : .
- Input parameters should be separated from each other by at least one blank space.
- Only the identifiers defined here will be accepted.
- Any line without known identifier will be considered as a comment line.
- Any text after the identifier will be considered as comment text.
- Any file name including its path name is allowed to be up to 80 characters.
- Blank line will be considered as a comment line.
- The orders of definition lines are optional if not specified.
- Parameters in < > are default or standard parameters.
- EOF marks the end of the input parameter file.

Definitions of the identifiers and their explanations:

- RINEX data file name, input Rinex GPS data file name.
- Coordinates xyz, sta_pro, input coordinates and property of the station at where above data are measured. This line should be put after the related line of Rinex data file name. Input parameter format: three float numbers and one integer. Definitions of station properties: 1, 2 for reference static, static, 3, 4 for kinematic, kinematic on the air, 6 for LEO onboard, 7, 8 for IGS reference, IGS. Above Rinex data file name and Coordinates line could be repeated until files and coordinates of all stations are listed.

- IGSorbit data file name, input file name of IGS precise orbit data. More files are allowed and can be given by several lines.
- Broadcast ephemerides file name, input file name of broadcast orbit data. More files are allowed and can be given by several lines.
- Document file name for output (also used to record the used input parameter file).
- IGSsta file name, input file of IGS station name list.
- IGSsitereceiver file name, input file of IGS site receiver type list.
- Satdat file name, input file of GPS satellite parameters.
- Polardat file name, input data file of polar motion.
- SelectSta file name, input file of selected stations for data processing.
- Oloadfile name, input file of ocean loading parameters.
- GPS_dat path name, the path name of the IGS GPS data, every day one line.
- Geopoten file name, the file name of geopotential model.
- InDataFile, data files used, 1, 2, 3 for using the GPS data given in this file, SelectSta file, both.
- Beginning date Year month day, measurement start date, format: three integers.
- DeltaDay, Days of total data, format: one integer (default input format).
- Begining time hour minute second, start epoch of the data processing, format: three integers.
- End time hour minute second, end epoch of the data processing, format: three integers.
- DeltaT, used data sampling rate, in seconds, format: one float number.
- Min_dat_win, minimum data window length, in seconds, format: one float number.
- DataInterval, data interval type, 1, 2, 3 for >=10sec., 1sec., 0.1sec.
- SKD, switch of kinematic/static/dynamic data processing modules (1/2/3).
- Icoordinates, <1, 2, 3, 4>, coordinates used by giving above, in Rinex file, by single point positioning, in station file. 3 is the standard option for kinematic data processing, 4 is for IGS network data processing.
- Dc, data combination type, <1, 2, 3, 4, 5, 6, 7> , 1 - L1, 2 - L2, 3 – L3, 4 - Lc (ion-free), 5 – int- ion-free 77%*(Fi(L1)-Fi(L2)*f(L2)/f(L1)), 6 - general combination: Factor1*Fi(L1) + Factor2*Fi(L2), 7 – code.
- DualF combination Factor1 Factor2, format : two float numbers.
- Cycleslipf, factor for cycle slips test, format : one float number.
- M_cyctest, cycleslip test methods, 1, 2 for using ionosphere residual, diff. Doppler fitting.
- Igs_v, igs data version number <0, 1, 2>, 0 - old version with x,y,z,xdot,ydot,zdot, 1 - new version with x,y,z,dt_clock, 2 - GFZ internal version with x,y,z,dt_clock,xdot,ydot,zdot.
- Trop_correction, <0, 1, 2> for <no, yes, as unknown>, 2 only used in static case, default <1>.
- Troposphere model used, <0, 1, 2, 3, 4, 5, 6, 7, 8> for <non, 8 models>.

- TroposHeight, |<1, 2, 3> for <standard, s_height, input parameters>.
- Lagrange polynomial order, standard value is <7>, less than 7 is not allowed.
- Inv_n, <1, 2> for <Cholesky, Gaussjordan> inverse method, if one failed, another will be automatically used, default <1>.
- Clock_correction, <0, 1, 2, 3> for <no, yes, ….>, should be <1>.
- InSMeph, 1, 2 for compute, read Smeph (Sun Moon planets orbit data).
- Tide_correction, <0, 1> for <no, yes> with the Earth tide correction, default <1>.
- Rela_correction, <0, 1> for <no, yes> with relativity correction, default <1>.
- Ioload_correction, <0, 1> for <no, yes> with ocean loading correction.
- Coord_ce (output forms), <1, 2, 3> for output in <Cartesian, geodetic ellipsoid, both> coordinates.
- CP_combine, <0, 1, 2> for <code, phase, phase-code>.
- C1P1, 1, 2 for C1, P1 code used in single point positioning.
- WcWp (code and phase weight factors), format : two float numbers. Weight standardization will be carried out in program.
- Iearthrot, <0, 1> for no, yes of the Earth rotational correction.
- Doppler_sign, <-1, 1> for Doppler data from <Berne_decoder, Landau_decoder>.
- Ioload_sign, <-1, 1>, theoretically, sign should be 1.
- Itidesign, <-1, 1>, theoretically, sign should be 1.
- Itropsign, <-1, 1>, static result shows Itropsign must be 1.
- Iclocksign1, <-1, 1>, theoretically, sign should be 1.
- Iclocksign2, <-1, 1>, theoretically, sign should be 1.
- Iclocksign3, <-1, 1>, theoretically, sign should be 1.
- Ambiguity fixing, <0, 1, 2> for <no, yes, equivalent> fixing.
- ReceivC, <0, 1> for no, yes with receiver clock error correction, must be <1>.
- Is_antenna_c, <0, 1> for <no, yes> correct satellite antenna centre offset.
- Ierot_tran (Ixyz) > 2 earthrot used in transmitting time correction.
- Itransrot (Iluisa), <0, 1> transmitting time rotation <no, yes>.
- K2run, further run for Run-n=4 or not, <0, 1> for no, yes.
- Ambiguity fixing search selector, 0 for no, else area factor (integer).
- Ifilter, <0, 1, 2, 3> for <sequential, Kalman, adaptive Kalman, intelligent Kalman> filtering.
- Equivalent method switch, 0 - zero differencing method, 1 - single differencing method, 2 - double differencing method, 3 - triple differencing method, 4 - user defined differencing method, 5 – equivalent method.
- I_C0 I_CS, order and grade of the geopotential model, format: two integers.

- Iorbit, input parameter for orbit data selection. 1, 3 for IGS in ECEF, broadcast orbit in ECEF, two integers.
- Isequential, 0, 1, 3 for epoch-block solution, sequential solution, accumulated total solution.
- I_sf_ef = 1, 2 for forming equation in ECSF or ECEF coordinate systems.
- I_nocoord = 1, 2 for fixed network or free network (no coordinates solved, coordinates solved).
- Ipostprocessing = 1, 2, 3, 4, 5 (see mfgsoft.c), post-processing switch.
- Idck_orbit = 4, 5, 6, 7 (orbit known, determine1,2, correction), orbit determination switch.
- Iapriori = 1, 2, 3, 11, 12, 13, 14, 15, 16, 17, a priori switch, 1, 2 for used for station coord., 6 orbit parameters, 3 for both, >=11 for IGS method: j1=Iapriori-10 for an*=(1+0.1**j1).
- Idatacondition = 0, 1 for no, yes to use the data conditions.
- Iinitial_update = 0, 1 for no, yes to update the initial values of the unknowns.

Definitions of the debug switches:

- Debug O_rinput =<0, 1, 2> : <read data, reprint data, check>
- Debug O_r_poten =<0, 1> : <no, yes, check>
- Debug O_r_oload =<0, 1> : <no, yes check>
- Debug O_oload_tide =<0, 1> : <no, yes, check>
- Debug O_igsdat =<0, 1, 2, 3> : <all, no, yes, check> in orbit.c defined switch
- Debug O_lagrange =<0, else> : <all, no>
- Debug O_lagrangef =<0, else> : <all, no>
- Debug O_rigs =<0, 1> : <all, no> (used in r_igs_d, r_igs_h)
- Debug O_broadcast =<0, 1> : <all, no, check> in broad2.c
- Debug O_borbc =<0, else> : <all, no>
- Debug O_eph_check =<0, else> : <all, no>
- Debug O_broadcast_orbit =<0, else> : <all, no>
- Debug O_r_eph =<0, else> : <all, no>
- Debug O_riono =<0, 1, 2> : <no, yes, check> print iono_residual
- Debug O_rinexd =<0, 1, else> : <all, part, no>
- Debug O_prepro =<0, else> : <all, no>
- Debug O_main =<0, 1, 2> : <no, yes, check> 1 gpsdat, 2 singlt
- Debug O_time0 =time switch (I==O_tome0 out, else else)
- Debug O_rinexh =<0, else> : <all, no> switch in rinex.c
- Debug O_refsat =<0, 1, 2, 3, 4, 5> : <all, ..., no> in orbit.c

- Debug O_getorb =<0, 1, 2> : <all, no, check>
- Debug O_getdat =<0, 1, else> : <all, part, no>
- Debug O_getdato =<0, else> : <all, no>
- Debug O_odat_tc =<0, else> : <all, no>
- Debug O_sdtdat =<0, else> : <all, no>
- Debug O_cycle =<0, 1, else> : <all, part, no> output
- Debug O_gpsdat =<-1, 0, 1, 2, 3, 4, 5, 6> : <all, o-, s-, d-, tdat, cyc, singl>
- Debug O_rcon =<0, else> : <all, no>
- Debug O_tro =<0, else> : <all, no>
- Debug O_rot =<0, else> : <all, no>
- Debug O_rela =<0, else> : <all, no>
- Debug O_Rotat =<0, else> : <all, no>
- Debug O_SL_eph =<0, 1, 2, 3, 4> : <all, ..., no>
- Debug O_tide =<0, 1, 2, 3, 4> : <all(tide_1day), all(interpo_rotat), ...no>
- Debug O_gcslsn =<0, 1, 2, 3> : <all, part, mini, no> in ad_core.c
- Debug O_gcsls =<0, 1, 2, 3> : <all, part, mini, no>
- Debug O_cls =<0, 1, 2, 3> : <all, part, mini, no>
- Debug O_rls =<0, 1, 2, 3> : <all, part, mini, no>
- Debug O_ls =<0, 1> : <all, no>
- Debug O_chol =<0, 1, 2, 3, 4, 5> : <all, part, mini, ..., no>
- Debug O_gaussj =<0, 1, 2, 3, 4, 5> : <all, part, mini, no>
- Debug O_eq =<0, 1, 2, 3, 4, 5, 6, 7> : in obs. equation
- Debug O_norm =<0, 1, 2, 3> : <all, part, mini, no> in normal equation
- Debug O_model =<0, 1, 2> : <all, part, no> in model function
- Debug O_t_level =<0, 1, 2, 3, 4, 5, 6, 7> : <all, part, mini, ..., no> in main.c
- Debug O_time1 =time switch end in main.c
- Debug O_Stop =time switch stop in main.c
- Debug O_norm_cp =<0, 1, 2, 3, 4, 5, 6> : <all, part, ..., no> in norm_dd_cp.c
- Debug O_kalman =<0, 1, 2> : <all, part, no> in ykalman.c
- Debug O_singlep =<0, 1, 2, 3, 4, 5, 6> : <all, part, ..., no>
- Debug O_velocity =<0, 1, 2, 3, 4, 5, 6> : <all, part, ..., no>

4.2 Standard Input Parameter file

Following is an example of a standard input parameter file for an IGS global network tectonic monitoring with GPS satellite orbit determination. Explanations will be outlined after this input parameter file. The Input_Parameter_File of all other numerical tests given in this manual can be obtained through minor modification from this standard input parameter file.

```
* to be defined input data file names and coordinates
*/home/sf/xu/1new/propro/ch2_121.rnxxx                : RINEX data file name
*              0.01            0.01    6356751.800 6      : Coordinates xyz, sta_pro
*/home/sf/xu/1new/propro/sylR122h.02ox                : RINEX data file name
* 3800689.6797  882077.3465  5028791.2898  1       : Coordinates xyz, sta_pro
*/home/sf/xu/1new/propro/sylM122h.02ox                : RINEX data file name
* 3800689.6797  882077.3465  5028791.2898  2       : Coordinates xyz, sta_pro
*/home/sf/xu/1new/propro/sylR122h.02ox                : RINEX data file name
* 3800689.6797  882077.3465  5028791.2898  3       : Coordinates xyz, sta_pro
*/home/sf/xu/1new/propro/sylM122h.02ox                : RINEX data file name
* 3800689.6797  882077.3465  5028791.2898  2       : Coordinates xyz, sta_pro
   * sta_pro : 1/2 static/_ref, 3/4 kinematic/_on_air, 6 LEO, 7/8 IGS ref /IGS
*/home/sf/xu/common_data/igs11642.sp3                 : IGSorbit data file name
/home/sf/xu/common_data/igs11643.sp3                  : IGSorbit data file name
*/home/sf/xu/common_data/igs11644.sp3                 : IGSorbit data file name
*/home/sf/xu/d120/ifag1200.02n                        : Broadcast ephemerides file name
*/home/sf/xu/d120/brdc1200.02n                        : Broadcast ephemerides file name
/home/sf/xu/d121/brdc1210.02n                         : Broadcast ephemerides file name
/home/sf/xu/d121/ifag1210.02n                         : Broadcast ephemerides file name
*/home/sf/xu/d122/ifag1220.02n                        : Broadcast ephemerides file name
*/home/sf/xu/d122/brdc1220.02n                        : Broadcast ephemerides file name
zout.1                                                : Document file name for output
/home/sf/xu/common_data/stako_gps_SNX_F_02     : IGSsta file name
/home/sf/xu/common_data/igs.snx_actual                : IGSsitereceiver file name
/home/sf/xu/common_data/gpssatdat_neu                 : Satdat file name
/home/sf/xu/common_data/pinit.SNX_B.02.001_999  : Polardat file name
/home/sf/xu/common_data/selected_station              : SelectSta file name
/home/sf/xu/common_data/oload.gps                     : Oloadfile name
*/home/sf/xu/d120/                                    : GPS_dat path name
/home/sf/xu/d121/                                     : GPS_dat path name
*/home/sf/xu/d122/                                    : GPS_dat path name
```

/home/sf/xu/common_data/POTFILET1_pgm055.cio : Geopoten file name

* to be defined parameters
2 : InDataFile, <1,2,3> datafiles used <above, file, both>
2002 5 1 : beginning date Year month day (2002 5 2)
1 : DeltaDay (total days, int)
0 30 0 : Beginning time hour minute sec (7 1 31) (0 35 0)
23 30 0 : End time hour minute sec (int) (8 1 30) (10 35 0)
30. : DeltaT (used sampling rate) (0.1) (10.)
900. : Min_dat_win (minimum data length in seconds) (double)
1 : DataInterval 1,2,3 for >=10sec, 1sec, 0.1sec (2)
2 : SKD, Static/Kinematic/Dynamic 0/1/2 (1/2)
4 : Icoordinates, coordinates used (3)
* 1,2,3,4 - use xyz <given above, in Rinex files, single point, in station file>
* 3 for kinematic by sigel point positioning
* 4 for IGS network data processing
4 <4>: Dc, data combination type, <1,2,3,4,5,6,7> , for L1, L2, L3, Lc (ion-free),
* int-ion-free, general combination: Factor1*Fi(L1) + Factor2*Fi(L2), code.
1. -1. : DualF combination Factor1 Factor2 if above dc=6 (Widelane)
4. : Cycleslipf, factor for cycle slip test, default 1.0
1 <1>: M_cyctest, cycleslip test methods
* 1 - ionosphere residuals
* 2 - differential Doppler fitting
1 <1>: Igs_v (igs data version), <0,1,2> for 3 versions
2 <1>: Trop_correction, <0,1,2> for <no, yes, as unknown>
7 <1>: Troposphere model used, <0,1,2,3,4,5,6,7,8> for <non, 8 models>
1 <1>: TroposHeight, <1,2,3> for <standard, s_height, input_parameters>
7 <7>: Lagrange polynorm order used, default 7
2 <2>: Inv_n, <1,2> for <Choleski,Gaussjordan> inverse method
1 <1>: Clock_correction, <0,1,2,3> for <no,yes,...,...>
2 <1>: InSMeph, <1,2> for <compute,read> Sun Moon planets orbits
1 <1>: Tide_correction, <0,1> for <no, yes>
1 <1>: Rela_correction, <0,1> for <no, yes>
1 <1>: Ioload_correction, <0,1> for <no, yes>
3 <3>: Coord_ce (output formats), <1,2,3> for <cartesian,ellipsoid,both>
2 : CP_combine, <0,1,2> for <code,phase,code-phase combination>
1 : C1P1, <1,2> for C1 or P1 used in single positioning.

```
1. 1000.    : WcWp (code and phase weight factors), default 1.0 1000.
1        <1>: Iearthrot, <0,1> for no,yes (int)
-1      <-1>: Doppler_sign, =<-1,1>
1        <1>: Ioload_sign <-1,1>
1        <1>: Itidesign, <-1,1>
1        <1>: Itropsign, <-1,1>, static shows Itropsign must be 1
1        <1>: Iclocksign1, <-1,1>
1        <1>: Iclocksign2, <-1,1>
1        <1>: Iclocksign3, <-1,1>
4        <4>: Ierot_tran (Ixyz) > 2 earthrot used in transmitting time correction
1        <1>: Itransrot (Iluisa), <0,1> transmitting time rotation <no,yes>(int)
1        <1>: ReceivC, =<0,1> no,yes for receiver clock correction (int)
1        <0>: K2run, =further run for Run-n=4 or not, <0,1> for no, yes (int)
2             : Ambiguity fixing search selector, 0,1,2 for no, yes, equivalent (int)
1             : Is_antenna_c, 0,1 for no,yes antenna mass centre correction
0             : Robust estimation for weight regulation, <0,1> for <no, yes>
50            : N_n (criterium for cancel N), default <20>
0             : Ifilter, <0,1,2,3> for <sequential,kalman,adaptive kalman,intelligent> filtering
5             : Equivalent method, 5
                * 0 - zero differencing method
                * 1 - single differencing method
                * 2 - double differencing method
                * 3 - triple differencing method
                * 4 - user defined differencing method
70 61         : I_C0 I_CS geopotential orders, maximum (70 61)
3             : Iorbit %d 1 igs, 2 igs_sf, 3 borb, 4 borb_sf (1)
1             : Isequential %d =0,1,3 for epoch, sequencial, accumulate solution
1             : I_sf_ef %d = 1,2 forming equation in ecsf or ecef coordinate systems (2)
2             : I_nocoord = 1,2 for fixed network or free network
5             : Ipostprocessing = 1,2,3,4,5 (see new983v.c)
5             : Idck_orbit = 4,5,6,7 (orbit known, determine1,2, correction)
0             : Ispriori = 1,2,3,11,12,13,14,15,16,17
1             : Idatacondition = 0,1 for no, yes to use the data conditions.
0             : Iinitial_update = 0,1 for no, yes to update the initial values of the unknowns.
EOF           : end of the input parameter file
```

In above file, the path name of the IGS data is given. The used stations are given in a list of selected stations. The list file of selected stations is defined one station per line, with station id,

number, continent, ECEF coordinates x,y,z, station property, internal number of the station. As soon as the firt character of the line is a blank, the station will be not included in the data processing. In this way to add or reduce a station may be realised easily by editing the file. Of course, the station and satellite related files, IGS and broadcast orbit data files, ocean loading parameters and geopotential files are given.

Data of May 1st 2002 is used for one day solution from 0:30:00 to 23:30:00. Station coordinates shall be solved for too. Orbit determination swithch is on, ECSF coordinate system must be used. Broadcast orbit data is used. Equivalent algorithm and sequential solution is preferred. No a priori information is used. Other parameters are standard.

4.3 Standard Debug Switches in Input Parameter File

The standard debug switches are defined as follows (default):

```
* to be defined standard debug switches
2     : Debug O_rinput   =<0,1> :<read data, reprint data>
2     : Debug O_r_poten  =<0,1> :<non, check>
2     : Debug O_r_oload  =<0,1> :<non, check>
2     : Debug O_oload_tide  =<0,1> :<non, check>
3     : Debug O_igsdat   =<0,1,2> :<all,non,check> in igsnew0.c defined switch
1     : Debug O_lagrange =<0,else>:<all,non>
1     : Debug O_lagrangef=<0,else>:<all,non>
1     : Debug O_rigs     =<0,1> :<all,non> (used in r_igs_d,r_igs_h)
3     : Debug O_broadcast=<0,1> : <all,check> in broad1.c
1     : Debug O_borbc    =<0,else>:<all,non>
1     : Debug O_eph_check=<0,else>:<all,non>
1     : Debug O_broadcast_orbit=<0,else>:<all,non>
1     : Debug O_r_eph    =<0,else>:<all,non>
2     : Debug O_riono    =<0,1> :<no,yes> print iono_residual
2     : Debug O_rinexd   =<0,1,else> :<all,part,non>
1     : Debug O_prepro   =<0,else> :<all,non>
2     : Debug O_main     =<0,1>:<non,yes> 1 gpsdat, 2 singlt
1     : Debug O_time0    =time switch (I==O_tome0 out, else else)
1     : Debug O_rinexh   =<0,else> :<all,non> switch in rinex.c
5     : Debug O_refsat   =<0,1,2,3,4,5>:<all,...,non> in orbit.c
2     : Debug O_getorb   =<0,else>:<all,non>
2     : Debug O_getdat   =<0,1,else> :<all,part,non>
1     : Debug O_getdato  =<0,else> :<all,non>
```

```
1     : Debug O_odat_tc  =<0,else> :<all,non>
1     : Debug O_sdtdat   =<0,else> :<all,non>
2     : Debug O_cycle    =<0,1,else> :<all,part,non> output
6     : Debug O_gpsdat=<-1,0,1,2,3,4,5>:<all,o-,s-,d-,tdat,cyc,singl>
1     : Debug O_rcon     =<0,else> :<all,non>
1     : Debug O_tro      =<0,else>:<all,non>
1     : Debug O_rot      =<0,else>:<all,non>
1     : Debug O_rela     =<0,else>:<all,non>
1     : Debug O_Rotat    =<0,else>:<all,non>
4     : Debug O_SL_eph   =<0,else>:<all,non>
4     : Debug O_tide=<0,1,2>:<all(tide_1day),all(interpo_rotat),non>
0     : Debug O_gcslsn   =<0,1,2,3>:<all,part,mini,non> in ad_core.c
0     : Debug O_gcsls    =<0,1,2,3>:<all,part,mini,non>
3     : Debug O_cls      =<0,1,2,3>:<all,part,mini,non>
0     : Debug O_rls      =<0,1,2,3>:<all,part,mini,non>
1     : Debug O_ls       =<0,1>:<all,non>
5     : Debug O_chol     =<0,,1,2,3>:<all,part,mini,non>
5     : Debug O_gaussj   =<0,1,2,3>:<all,part,mini,non>
7     : Debug O_eq =<0,1,2,3,4,5,6,7>: in obs. equation
3     : Debug O_norm     =<0,1,2>:<all,part,non> in normal equation
0     : Debug O_model    =<0,1,2>:<all,part,non> in model function
7     : Debug O_t_level  =<0,1,2,3>:<all,part,mini,non> in main.c
-1    : Debug O_time1    =time switch end in main.c
-1    : Debug O_Stop     =time switch stop in main.c
6     : Debug O_norm_cp  =<0,1,2>:<all,part,non> in norm_dd_cp.c
2     : Debug O_kalman   =<0,1,2>:<all,part,non> in ykalman.c
1     : Debug O_screen   =<0,1>:<all,non>
6     : Debug O_singlep  =<0,1,2,3,4,5,6>:<all,part,non>
6     : Debug O_velocity  =<0,1,2,3,4,5,6>:<all,part,non>
```

The debug switches are only needed for the advanced users to locate the errors and to regulate the test and intermidial output.

5. Multi-Functional GPS Data Processing

In this section, the multi-functinalities of the MFGsoft will be outlined with examples of the related input parameter files. They are: standard global network monitoring with dynamic orbit determination and with kinematic/dynamic combined orbit determination as well as with known orbit, regional network monitoring with orbit correction and with known orbit, local network static and kinematic positioning, onboard LEO satellite kinematic and kinematic/dynamic orbit determination.

5.1 Global Network Monitoring and Dynamic Orbit Determination

The input parameter file is exactly given in section 3.3. Significant alternative solutions may be obtained e.g. by extending the data length (up to three days), adjusting the data sampling rate, changing data combination methods, fixing the network, selecting different reference stations, or using different a priori information. The definitions of the reference stations have to be changed by editing the selected station file, and other related parameter lines are listed below:

```
2002  5  1 : beginning date Year month day (2002 5 2)
1           : DeltaDay (total days, int)
0 30 0      : Beginning time hour minute sec (7 1 31) (0 35 0)
23 30 0     : End time hour minute sec (int) (8 1 30) (10 35 0)
30.         : DeltaT (used sampling rate) (0.1) (10.)
4       <4>: Dc, data combination type, <1,2,3,4,5,6,7>
2           : CP_combine, <0,1,2> for <code,phase,code-phase combination>
2           : I_nocoord = 1,2 for fixed network or free network
0           : Ispriori = 1,2,3,11,12,13,14,15,16,17
```

5.2 Global Network Monitoring and Kinematic/Dynamic Orbit Determination

The only one switch different for such solution compared with the computation outlined in section 5.1, is that the Idck_orbit switch should be set to 6, i.e.

```
6           : Idck_orbit = 4,5,6,7 (orbit known, determine1,2, correction)
```

The significant alternative options mentioned in section 5.1 above are also applicable here (see §5.1). The advantage of a kinematic/dynamic combined orbit determination is that the orbit is fitted by a dynamic model and determined geometrically. No dynamic force models are used, so the processing is faster and easier.

5.3 Regional Network Monitoring with Orbit Correction

The only one switch difference for such solution compared with the computation outlined in section 5.1 is that the Idck_orbit switch should be set to 7, i.e.

```
7          : Idck_orbit = 4,5,6,7 (orbit known, determine1,2, correction)
```

The selected station file is of course different and includes only names of the regional stations. The significant alternative options mentioned in §5.1 are also applicable. Here the orbit correction model are used to fit the data so that the precision of the regional network monitoring can be modified. It is suggested that the IGS precise orbit data shall be used in this case.

5.4 Local Network Static and Kinematic Positioning

Generally speaking, the GPS data file names are given in the input parameter file in case of local network data processing. The InDataFile switch has to be changed to 1, i.e.

```
1          : InDataFile, <1,2,3> datafiles used <above, file, both>
```

Local network static data processing

In the standard input parameter file given in §5.1, there are two data files from two stations which have the station properties of 1 and 2. These two lines and the related two coordinate lines shall be switched on, i.e. (cancel the * mark)

```
 3800689.6797  882077.3465  5028791.2898  1      : Coordinates xyz, sta_pro
/home/sf/xu/1new/propro/sylM122h.02ox            : RINEX data file name
 3800689.6797  882077.3465  5028791.2898  2      : Coordinates xyz, sta_pro
/home/sf/xu/1new/propro/sylR122h.02ox            : RINEX data file name
```

The reference station has the station property of 1, and the unknown static station has the station property of 2. Therefore the coordinates give here for reference station shall be very precise. The approximated coordinates of unknown station shall be given. Because the data are measured on the May 2nd, 2002 with sampling rate of 0.1 seconds in the time duration from 7:01:31 to 8:01:30, the related parameters have to be changed. The IGS or broadcast orbit files have to be changed accordingly, i.e.

```
*/home/sf/xu/common_data/igs11643.sp3            : IGSorbit data file name
/home/sf/xu/common_data/igs11644.sp3             : IGSorbit data file name
```

or

```
*/home/sf/xu/d121/brdc1210.02n                : Broadcast ephemerides file name
*/home/sf/xu/d121/ifag1210.02n                : Broadcast ephemerides file name
/home/sf/xu/d122/ifag1220.02n                 : Broadcast ephemerides file name
/home/sf/xu/d122/brdc1220.02n                 : Broadcast ephemerides file name
```

and

```
2002  5  2 : beginning date Year month day (2002 5 2)
1          : DeltaDay (total days, int)
7 1 31     : Beginning time hour minute sec (7 1 31) (0 35 0)
8 1 30     : End time hour minute sec (int) (8 1 30) (10 35 0)
0.1        : DeltaT (used sampling rate) (0.1) (10.)
3          : DataInterval 1,2,3 for >=10sec, 1sec, 0.1sec (3)
1          : SKD, Static/Kinematic/Dynamic 0/1/2 (1/2)
1          : Icoordinates, coordinates used (1 3)
             * 1,2,3,4 - use xyz <given above, in Rinex files, single point, in station file>
             * 3 for kinematic by sigel point positioning
             * 4 for IGS network data processing
```

Local network kinematic data processing

In the standard input parameter file given in §5.1, there are two data files from two stations which have the station properties of 1 and 3. These two lines and the related two coordinate lines shall be switched on, i.e. (cancel the * mark)

```
/home/sf/xu/1new/propro/sylR122h.02ox          : RINEX data file name
 3800689.6797  882077.3465  5028791.2898  1    : Coordinates xyz, sta_pro
/home/sf/xu/1new/propro/sylR122h.02ox          : RINEX data file name
 3800689.6797  882077.3465  5028791.2898  3    : Coordinates xyz, sta_pro
```

Other changes of the parameters are the same as outlined above except the lines of run module and approximate coordinates:

```
2          : SKD, Static/Kinematic/Dynamic 0/1/2 (1/2)
3          : Icoordinates, coordinates used (1 3)
             * 1,2,3,4 - use xyz <given above, in Rinex files, single point, in station file>
             * 3 for kinematic by sigel point positioning
             * 4 for IGS network data processing
```

5.5 Onboard Kinematic/Dynamic Orbit Determination

The names of the LEO on board GPS data files are given in a file called leoorbitfiles. One file per line and the name has maximal 80 charaters. The data files will be read automatically according to the time duration in which the data will be processed. The example data used here are three days from April 30th 2002 to May 2nd 2004 and the leoorbitfiles are given below:

/home/sf/xu/champ_data/orbit/2002_118_22.dat
/home/sf/xu/champ_data/orbit/2002_119_10.dat
/home/sf/xu/champ_data/orbit/2002_119_22.dat
/home/sf/xu/champ_data/orbit/2002_120_10.dat
/home/sf/xu/champ_data/orbit/2002_120_22.dat
/home/sf/xu/champ_data/orbit/2002_121_10.dat
/home/sf/xu/champ_data/orbit/2002_121_22.dat

Many parameters are not relevant for this kind of data processing. The to be changed input parameters are listed as follows:

4 : InDataFile, <1,2,3,4> datafiles used <above, file, both,extra>
10. : DeltaT (used sampling rate) (0.1)
6 : Idck_orbit = 4,5,6,7 (orbit known, determine1,2, correction)

5.6 Other Functions

For reasons one needs to know the precise coordinates of the reference station of a local or regional network. In this case the station can be combined into the IGS network for a common data processing. To do so, one just needs to list the interested data file name in the input parameter file, and sets the InDataFile to 3, i.e.

3 : InDataFile, <1,2,3> datafiles used <above, file, both>

For some special purposes of global or regional network data processing, such as troposphere or ionosphere sounding, the problem may also be solved in a orbit fixed or network fixed ways. The network fixing switch and the orbit fixing switch can be used selectively. They are:

1 : I_nocoord = 1,2 for fixed network or free network
4 : Idck_orbit = 4,5,6,7 (orbit known, determine1,2, correction)

Other significant possibilities are listed follows:

- Switch on and off the data conditions
- Switch on and off the a priori information in different modulus
- Switch between epoch-block solution, sequential solution and total solution.
- Switch between all kinds of data processing algorithms and dada combinations
- Switch on to update the initial values of the unknowns (a kind of iteration)

In case of orbit fixed solution, the problem can be solved in ECEF or ECSF coordinate systems.

6. Structure and Diagram of MFGsoft

The most important software packages of MFGsoft are the physical models, algorithms and tools. They are listed below (figures are given in the appendicies).

Physical Models:

1. Tropospheric models
2. Ionospheric model
3. Relativity model
4. Earth tide model
5. Ocean loading tide model
6. Satellite mass centre model
7. Solar radiation model
8. Atmospheric drag model
9. Geopotential disturbance
10. Tidal potential disturbance
11. Sun, Moon, planets orbit models
12. Multi-bodies disturbance
13. Dynamic orbit fitting model

Algorithms:

1. Equivalent algorithm ---- undifferenced algorithm
   ---- differencing algorithms
1. Diagonalisation algorithm
2. Independent parameterisation --- optimal baselines
   --- data conditions
3. Cycle slip detection
4. Ambiguity --- initialisation
   --- search
5. Adjustment and filtering --- Least Squares
   --- Block-wise elimination
   --- Sequential
   --- A priori information
   --- Kalman filter
   --- intelligent Kalman filter
6. Differentiator/integrators --- variance equation

--- orbit

7. Differential Doppler

Tools:

1. Coordinates transformers
2. Time systems transformers
3. Sun-Moon-planets orbit creator
4. Broadcast orbit transformer
5. Interpolation and integration
6. Matrix inverse --- Gauss-Jordan
   --- Cholesky
7. Helmert transformation
8. Mapping functions
9. Flight state computation
10. Spectral analysis methods
11. Statistic analysis
12. Graphic representation

Data processing diagram:

1. Program start
2. Common part --- Read input parameter file
   --- Read other files given
   … Read/create Sun-Moon-planets orbits
   --- Compute Earth/ocean loading tide
   … Orbit data transformation
   … Data pre-processing if possible
   --- initialisation … optimal baselines
   --- ambiguity
   … variation equation
3. Sequential time do loop
   --- get real time data
   --- models and parameters acquisition
   --- single point positioning
   … velocity determination
   --- ambiguity check and set
   --- modulus switch … local net
   … on board
   … global/regional net
4. summary part … forecast
   … iteration
   … analysis

modulus switch:

modulus switch --- global/regional net --- ambiguity check
--- models acquisition
--- observation equation
… Fi and Kepler/Jacoby matrices
--- unknown check with data conditions
--- normal equation
--- exchange, elimination, accumulation

--- adjustment/filter
… diagonalisation

7. Strategies and Principles Used

The MFGsoft is based on a new theoretical background and is a realisation of several new algorithms. The priciples will be outlined in this section. As soon as there exists the references, the theories will be described briefly.

7.1 Equivalent GPS Data Processing Algorithm

In GPS data processing practice, the commonly used methods are so-called zero-difference (un-differential), single-difference, double-difference and triple-difference methods (Bauer 1994; Hofmann-Wellenhof et al. 1997; King et al. 1987; Leick 1995; Remondi 1984; Seeber 1993; Strang and Borre 1997; Wang et al. 1988). It is well-known that the observation equations of the differencing methods can be obtained by carrying out a related linear transformation to the original equations. As soon as the weight matrix is similarly transformed according to the law of covariance propagation, all methods are equivalent, theoretically. A theoretical proof of the equivalence between the un-differential and differential methods can be found in Schaffrin and Grafarend (1986). An algebra proof is given in Xu (2002) where a unified GPS data processing method based on equivalently eliminated equations is proposed. By selecting the eliminated unknown vector as a vector of zero, a vector of satellite clock error, a vector of all clock error, a vector of clock and ambiguity parameters, or a vector of user-defined unknowns, the selectively eliminated equivalent observation equations can be formed, respectively. The equations are equivalent to the zero-, single-, double-, triple-, or user-defined differenced equations. The advantage of such a method is that the observational vector remains the original one and the weight matrix keeps the un-correlated diagonal form. The detailed description can be found in Xu (2003, §6.7 pp.107-118, §7.6 pp.132-135).

7.1.1 Formation of Equivalent Observation Equations

For the convenience of readers, the method to form an equivalently eliminated equation system is outlined here. The theory is given in Xu (2002) in detail. In practice, sometimes only one group of unknowns is of interest; it is better to eliminate the other group of unknowns (called nuisance parameters), for example, because of their size. In this case, using the so-called equivalently eliminated observation equation system could be very beneficial (Wang et al. 1988; Xu and Qian 1986; Zhou 1985). The nuisance parameters can be eliminated directly from the observation equations instead of from the normal equations.

The linearised observation equation system can be represented using the matrix:

$$V = L - \begin{pmatrix} A_1 & A_2 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}, \qquad P \tag{1}$$

where L is the observational vector of dimension m; A_1 and A_2 are coefficient matrices of dimension $m\times(n-r)$ and $m\times r$; X_1 and X_2 are unknown vectors of dimension n-r and r; V is a residual vector of dimension m; n and m are numbers of total unknowns and observations respectively; P is a symmetric and definite weight matrix of dimension $m\times m$. For un-correlated observation vector L, P is a diagonal matrix.

Least squares normal equation of Eq.1 can be formed then by

$$\begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix}\begin{pmatrix} X_1 \\ X_2 \end{pmatrix}=\begin{pmatrix} W_1 \\ W_2 \end{pmatrix}, \tag{2}$$

where

$$\begin{pmatrix} A_1^T PA_1 & A_1^T PA_2 \\ A_2^T PA_1 & A_2^T PA_2 \end{pmatrix}=\begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix}=M$$
$$M^{-1}=Q=\begin{pmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{pmatrix} \quad . \tag{3}$$
$$W_1=A_1^T PL, \quad W_2=A_2^T PL$$

Eliminating X_2 and X_1 in Eq.2 respectively, the related equivalent normal equations of X_1 and X_2 can be obtained. Combining them together one has the so-called diagonalised normal equation (Xu 2003a)

$$\begin{pmatrix} M_1 & 0 \\ 0 & M_2 \end{pmatrix}\begin{pmatrix} X_1 \\ X_2 \end{pmatrix}=\begin{pmatrix} B_1 \\ B_2 \end{pmatrix}, \tag{4}$$

where (e.g. Cui et al. 1982, Gotthardt 1978)

$$Q_{11}=M_1^{-1}, \quad Q_{22}=M_2^{-1}, \tag{5}$$

$$\begin{aligned} M_1 &= M_{11}-M_{12}M_{22}^{-1}M_{21} \\ B_1 &= W_1-M_{12}M_{22}^{-1}W_2 \end{aligned}, \tag{6}$$

$$\begin{aligned} M_2 &= M_{22}-M_{21}M_{11}^{-1}M_{12} \\ B_2 &= W_2-M_{21}M_{11}^{-1}W_1 \end{aligned}. \tag{7}$$

It is obvious that Eq.2 and Eq.4 are two equivalent normal equations. The solutions of the both equations are identical. The derivation of Eq.4 from Eq.2 is not a simple linear transformation and is irreversible.

The related equivalent observation equations of the diagonal normal Eq.4 can be written (Xu 2003a)

$$\begin{pmatrix} U_1 \\ U_2 \end{pmatrix}=\begin{pmatrix} L \\ L \end{pmatrix}-\begin{pmatrix} D_1 & 0 \\ 0 & D_2 \end{pmatrix}\begin{pmatrix} X_1 \\ X_2 \end{pmatrix}, \quad \begin{pmatrix} P & 0 \\ 0 & P \end{pmatrix}, \tag{8}$$

where U_1 and U_2 are residual vectors which have the same property as V in Eq.1, and

$$D_1=(I-K)A_1, \quad D_2=(I-J)A_2, \tag{9}$$

where I is an identity matrix, and (Xu 2003a, Wang et al. 1988, Xu and Qian 1986, Zhou 1985)

$$J=A_1M_{11}^{-1}A_1^T P \ , \quad K=A_2M_{22}^{-1}A_2^T P \ . \tag{10}$$

Matrices J, K, (I-J) and (I-K) are idempotent, $(I-J)^T P$ and $(I-K)^T P$ are symmetric. These properties are used to derive Eq.8 from Eq.4, and of course, can be used to form the normal equation of Eq.8 to get Eq.4. Again, the derivation of Eq.8 from Eq.1 is not a simple linear transformation and is irreversible. For reasons, we define the equivalent cofactor matrix Q_e by

$$Q_e = \begin{pmatrix} M_1 & 0 \\ 0 & M_2 \end{pmatrix}^{-1} = \begin{pmatrix} Q_{11} & 0 \\ 0 & Q_{22} \end{pmatrix}. \tag{11}$$

Which has the same diagonal element blocks as the original cofactor matrix Q and guarantees that the precision relation between the unknowns remains the same. Such a definition is indeed implicitly used in the traditional block-wise least squares adjustment (cf. Xu 2003 p127-132).

A GPS data processing algorithm that uses the second equation of Eq.8 (denoted by Eq.8_2) is said to use the method of equivalent observation equation selectively. Selecting X_1 in the first equation of Eq.8 (denoted by Eq.8_1) as zero vector, the algorithm is the same as the undifferenced method. Selecting X_1 as the satellite clock error vector, as the vector containing all clock errors, as the vector containing all clock errors and ambiguities, or as the vector containing any user defined parameters, then the algorithm is equivalent to the single-difference method, the double-difference method, the triple-difference method, or the user defined eliminating method, respectively. The unknown X_1 can be computed separately with Eq.8_1 if desired (Xu 2002).

The equivalence property is valid under three implicit assumptions. The first one is that the identical observation vector L is used. The second is that the parameterisation of X_2 is identical. The third is that the vector X_1 should be able to be eliminated. Otherwise the equivalence does not hold (Xu 2002).

The advantages of this method are (compared with un-differential and differential methods):

- The un-differential and differential GPS data processing can be dealt with in an equivalent and unified way. The data processing scenarios can be selected by a switch and used in a combinative way;
- The eliminated parameters can be also solved separately with the same algorithm;
- The weight matrix remains the original diagonal one;
- The original observations are used; no differencing is required.

It is obvious that the described algorithm meanwhile has all the advantages of all un-differential and differential GPS data processing methods.

7.2 Diagonalisation Algorithm

In the sequential adjustment and Kalman filtering, there is a problem which has not been described exactly and theoretically in literature before. That is how to give up the nuisance parameter related information from the past. For a real time data processing, it is specially important to keep the updated problem as small as possible. The so-called diagonalisation algorithm is proposed initially for deriving an equivalent ambiguity search criterion and then used for giving up the nuisance parameter related information (Xu 2003a).

Suppose the past surveying information is presented in a normal equation system

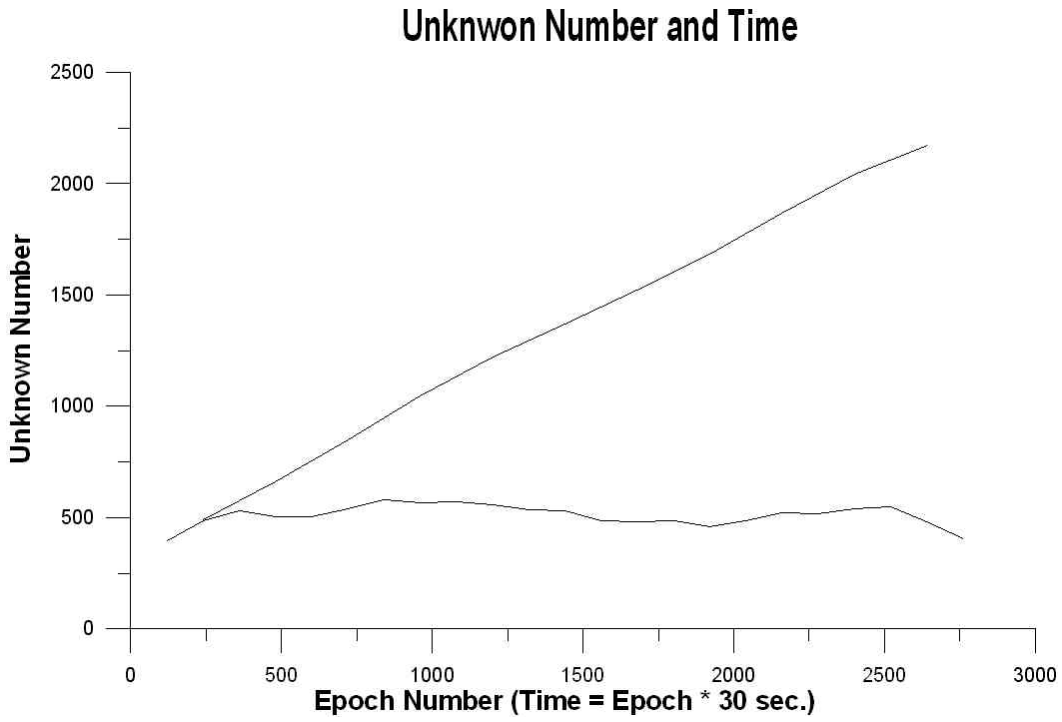
$$\begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} W_1 \\ W_2 \end{pmatrix}, \tag{12}$$

and the normal equation of the present epoch-block consists only the parameter sub-vector X_2. Then the Eq.12 can be diagonalised by

$$\begin{pmatrix} M_1 & 0 \\ 0 & M_2 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} B_1 \\ B_2 \end{pmatrix}, \tag{13}$$

so that one just needs to accumulate the X_2 related part in Eq.13 into the present normal equation and then to solve the problem. In such way the X_1 is considered as nuisance parameter sub-vector and is given up during the sequential data processing so that the data processing problem will be able to be kept as small as possible.

Following graphic shows the relationship between unknown number and time for a sequential solution of IGS network with 47 stations and fixed orbit for a network monitoring problem. Without diagonalisation algorithm, the parameter number increases to 2300 whereas with diagonalisation algorithm, the parameter number remains around 500. Without such an algorithm an exact and effective real time data sequential processing and filtering is not realistic.



7.3 Optimal Ambiguity Search Criteria

The ambiguity search criterion is a contradict topic in international GPS research community since a few years.

Suppose GPS observation equation is Eq.1 and its least squares normal equation is Eq.2, where $X_2 = N$ (N is the ambiguity sub-vector) and $X_1 = Y$ (Y is the rest unknown sub-vector). The least squares ambiguity search (LSAS) criterion (Teunissen 1995, Leick 1995, Hofmann-Wellenhof et al. 1997, Euler and Landau 1992; Han and Rizos 1997) is

$$\delta(dN) = (N_0 - N)Inv(Q_{22})(N_0 - N), \tag{14}$$

where N_0 is the float solution of the ambiguity sub-vector, $dN = N_0 - N$. The ambiguity search is a process to find out a vector N in the searching area so that the value of $\delta(dN)$ reaches the minimum.

The so-called general ambiguity search criterion is derived in Xu (2002a) and has a form of

$$\delta(dX) = (X_0 - X)^T Inv(Q)(X_0 - X), \tag{15}$$

where $X = (Y \quad N)^T$, $X_0 = (Y_0 \quad N_0)^T$, $dX = X_0 - X$, index 0 denotes the float solution. The search is a process to find out a vector X (includes N in the searching area and Y computed) so that the value of $\delta(dX)$ reaches the minimum. The optimal property of this criterion can be found in Xu (2002a).

For the equivalent normal equation 4, the related equivalent criterion is

$$\begin{aligned}\delta_1(dX) &= (\mathrm{Y}_0 - Y)^T Inv(Q_{11})(Y_0 - Y) + (N_0 - N)Inv(Q_{22})(N_0 - N) \\ &= \delta(dY) + \delta(dN)\end{aligned}, \tag{16}$$

where index 1 is used to distinguish the equivalent criterion from the general one given in Eq.15.

It is worth mentioning that the well-known ambiguity function (AF) method is mathematically incorrect (cf. Xu 2003 pp.280-283) and the so-called least squares (LS) criterion is generally not optimal (cf. Xu 2003 pp153-172). For information of the readers, a brief discussion is given below concerning the LS criterion.

An excellent summary of the derivation of the ambiguity search LS criterion is given in Verhagen (2004). In terminology of Teunissen (1995) the observation model is

$$y = A\,a + B\,b + e, \quad Q_y \tag{17}$$

and the solution of Eq.17 can be obtained by the following minimization problem

$$\min_{a,b} \|y - Aa - Bb\|^2_{Q_y}, \quad a \in Z^n, b \in R^n. \tag{18}$$

Eq.18 can be orthogonalised by

$$\|y - Aa - Bb\|^2_{Q_y} = \|\hat{e}\|^2_{Q_y} + \|\hat{a} - a\|^2_{Q_y} + \|\hat{b}(a) - b\|^2_{Q_y}. \tag{19}$$

The explanations of the symbols can be found in Verhagen (2004). The solution of Eq.18 is then obtained by three steps: 1). compute the float solution of Eq.17; 2). minimize the second term (denoted by δ_a for later convenience) on the right-hand side of Eq.19 and set the third term (denoted by δ_b) to zero to get the fixed ambiguity; 3). compute the fixed coordinates using fixed ambiguity.

The key problem is that the δ_b is not allowed to be set to zero. The significant opposite arguments are listed below:

- To set δ_b to zero or any constant are equivalent, however the δ_b is neither zero nor constant for all coordinate vectors computed from all to be searched ambiguity candidate vectors. Therefore to set δ_b to zero is indeed to omit the term δ_b in Eq.19 (in other words, in the second step the effect of the term δ_b has never been taken into account in this way).
- To omit the term δ_b in the minimisation problem Eq.19 is allowed only if following assumption is true, i.e. a minimum δ_a will lead to a minimum δ_b and therefore lead to a minimum of $(\delta_a + \delta_b)$. However, theory and numerical examples shown that such an assumption cannot be generally true (cf. Xu 2002). The omission of the term δ_b in Eq.19 destroys the equivalence between Eq.18 and Eq.19. The problem of minimisation of $(\delta_a + \delta_b)$ is degraded to a problem of minimisation of δ_a. In this way the obtained solution cannot be generally the same as the solution obtained directly from Eq.18.
- Noting optimality und uniqueness properties of the solution of Eq.18, the solution obtained by minimising δ_a cannot generally be the optimal ones of Eq.18. Any indication obtained by such non optimal results may not be really true.
- To set δ_b to zero is the same as to set b as float solution (then δ_b is zero). However, b cannot be the float solution. Proof: accordingly to the summarised method select b = float coordinate solution; through the minimisation of δ_a one gets the vector a, then one can compute b using a; because a is not the float ambiguities, therefore b is not the float coordinate solution. So the result of b states that b does not equal float coordinate solution; this is in conflict with the assumed starting value.
- To set δ_b to a constant is not allowed either. Proof: for b = float coordinate solution + any constant vector (e.g. b = 0 or b = float solution + constant vector of 1000 km) one has δ_b = constant, then following $\text{mini}(\delta_a+\delta_b) = \text{mini}(\delta_a)$; the sought for results are the same by using the summarised method, no matter the computation is started from given float position or one 1000 km farther away. Therefore setting δ_b = constant would be incorrect.

7.4 Independent Parameterisation Method

The parameterisation problem of the equivalent GPS data processing method is studied in details in Xu (2004). As mentioned in Section 7.1, the equivalence property is valid under three implicit assumptions. The first one is that the identical observation vector L is used. The second is that the parameterisation of X_2 is identical. The third is that the vector X_1 should be able to be eliminated. Otherwise the equivalence does not hold (Xu 2002).

The first equivalent condition is necessary for the exactness of the equivalence because of the fact that through forming differences the un-paired data will be cancelled out in the differencing data processing.

The second equivalent condition states that the parameterisation of the undifferenced and differencing models should be the same. This may be interpreted as: the rank of the undifferenced and differencing problem should be the same if the differencing is formed by a full rank linear transformation. If only the differencing equations are taken into account, then the rank of the undifferenced model should equal the rank of the differencing model plus the number of eliminated independent parameters.

It is well-known that one of the clock error parameters is linearly correlated with the others. This may also be seen in the proof of the equivalent property of the double differences, where the two receiver clock errors of the baseline may not be separated from each other and have to be transformed to one parameter and then eliminated (Xu 2002). This indicates that if in

undifferenced model all clock errors are modeled, the problem will be singular (i.e. rank defect). Indeed, Wells et. al. noticed early in 1987 that the equivalence is valid if measures are taken to avoid rank defect in the bias parameterisation (Wells et al. 1987). Which clock error has to be kept fixed is arbitrary. Because of the different qualities of the satellite and receiver clocks, a good choice is to fix a satellite clock error (the clock is called reference clock). In practice, the clock error is an unknown, therefore there is no way to keep that fixed except to fix it to zero. In such a case, the meaning of the other bias parameters will be changed and may represent the relative errors between the other biases and that of the reference (Xu 2003 pp.279-280). The third equivalent condition is important to ensure a full ranked parameterisation of the to be eliminated parameter vector X_1.

The undifferenced Eq.1 is solvable if the parameters X_1 and X_2 are not over-parameterised. In case of single differences, X_1 includes satellite clock errors and is able to be eliminated. Therefore, to guarantee that the undifferenced model Eq.1 is not a singular one, X_2 in Eq.1 must be not over-parameterised. In case of double differences, X_1 includes all clock errors except the reference one. Here we notice that the equivalent observation Eq.8_2 (denotes the second equation of Eq.8) is equivalent to the double differencing observation equation and Eq.4_2 (denotes the second equation of Eq.4) is the related normal equation. In traditional double differencing observation equation, the ambiguity parameters are represented by double differencing ambiguities. Recall the equivalent property, the number (or rank) of not linearly correlated ambiguity parameters in X_2 must be equal to the number of the double differencing ambiguities. In case of triple differences, X_1 includes all clock errors and ambiguities. The fact that X_1 should able to be eliminated leads again to the conclusion that the ambiguities should be linearly independent.

The two equivalent linear equations should have the same rank. Therefore, if all clock errors except the reference one are modelled, the number of independent undifferenced ambiguity parameters should be equal to the number of double differencing ambiguities.

According to the definition of the double differencing ambiguity and a special example of observation equation, the method of independent parameterisation and the so-called data conditions as well as the extended way of double differencing forming are derived (Xu 2004).

7.4.1 Sequential Data Dealing and Geometric Illustration

In case of sequential data dealing, as soon as the configuration of the observed satellites do not change greatly and the receivers work ideally, the linear correlations between the bias parameters remain the same. With the rotation of the earth and changing of the configurations of the satellites, the independent bias parameter set and the to be kept fixed parameter set are changing too. If the to be fixed parameters are determined beforehand, then of course they should be fixed to the real values, or in other words, they should be relaxed from fixing conditions. Therefore one may include all bias parameters in the model first and meanwhile notice which parameters are over-parameterised ones, then modify the record of over-parameterisation through accumulation of the sequential notes of the sequential equations. In this way the over-parameterised biases can be exactly identified and kept fixed if sequential solutions are needed. The reference clock parameter and the ambiguities of the reference station have to be kept fixed in any way. An artificial changing of the reference clock and reference station will make the sequential equations inconsistent to each other.
The reason why the reference parameters have to be fixed lies in the nature of distance measurements, which cannot provide the datum origin (cf. e.g. Wells et al. 1987 p9). Suppose d is the direct measurement of clock errors of satellite k and receiver i, i.e. $d_i^k = \delta_i + \delta_k$, no matter how many observations were made and how the indices were changed, one parameter (i.e. reference

clock) is inseparable from the others and has to be fixed. The reason why the ambiguities and the clock biases are partly linear correlated can be similarly analysed. Suppose h is the direct measurement of clock errors of satellite k and receiver i and ambiguity N, i.e. $h_i^k = \delta_i + \delta_k + N_i^k$, the number of over-parameterised biases is exactly the number of total observed satellites and used receivers. This ensures again that the parameterisation method to fix the reference clock and one ambiguity of every satellite as well as one ambiguity of the reference satellite of every station is reasonable. The case of combination of d and h (as code and phase observations) will be discussed below.

7.4.2 Correlation Analysis in Case of Phase-Code Combination

A phase-code combined observation equation can be written by (cf. Xu 2003 p131):

$$\begin{pmatrix} V_1 \\ V_2 \end{pmatrix} = \begin{pmatrix} L_1 \\ L_2 \end{pmatrix} - \begin{pmatrix} A_{11} & A_{12} \\ A_{11} & 0 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} \quad \text{and} \quad P = \begin{pmatrix} w_{\mathrm{p}} P_0 & 0 \\ 0 & w_{\mathrm{c}} P_0 \end{pmatrix} . \tag{20}$$

where L_1 and L_2 are the observational vectors of phase (scaled in length) and code, respectively; V_1 and V_2 are related residual vectors; X_2 and X_1 are unknown vectors of ambiguity and others; A_{12} and A_{11} are related coefficient matrices; P_0 is a symmetric and definite weight matrix; w_{p} and w_{c} are weight factors of the phase and code observations.

The phase, code and phase-code normal equations can be formed respectively by:

$$\begin{pmatrix} N_{11} & N_{12} \\ N_{21} & N_{22} \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} R_1 \\ R_2 \end{pmatrix}, \quad N_{11} X_1 = R_c, \quad \begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} B_1 \\ B_2 \end{pmatrix}, \tag{21}$$

where

$$\begin{aligned} M_{11} &= (w_{\mathrm{p}} + w_{\mathrm{c}}) A_{11}^T P_0 A_{11} = (w_{\mathrm{p}} + w_{\mathrm{c}}) N_{11} \ , \\ M_{12} &= M_{21}^T = w_{\mathrm{p}} A_{11}^T P_0 A_{12} = w_p N_{12} \ , \\ M_{22} &= w_{\mathrm{p}} A_{12}^T P_0 A_{12} = w_p N_{22} \ , \\ B_1 &= A_{11}^T P_0 (w_{\mathrm{p}} L_1 + w_{\mathrm{c}} L_2) = w_p R_1 + w_c R_c \quad \text{and} \\ B_2 &= w_{\mathrm{p}} A_{12}^T P_0 L_1 = w_p R_2 \ . \end{aligned} \tag{22}$$

Covariance matrix is denoted

$$Q = Inv \begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix} = \begin{pmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{pmatrix}, \tag{23}$$

where (Gotthardt 1978; Cui et al. 1982)

$$\begin{aligned} Q_{11} &= (M_{11} - M_{12} M_{22}^{-1} M_{21})^{-1} \ , \\ Q_{22} &= (M_{22} - M_{21} M_{11}^{-1} M_{12})^{-1} \ , \\ Q_{12} &= M_{11}^{-1} (-M_{12} Q_{22}) \quad \text{and} \\ Q_{21} &= M_{22}^{-1} (-M_{21} Q_{11}) \ . \end{aligned} \tag{24}$$

i.e.

$$Q_{11} = ((w_p + w_c)N_{11} - w_p N_{12} N_{22}^{-1} N_{21})^{-1} \ ,$$

$$Q_{22} = (w_p N_{22} - w_p^2 (w_p + w_c)^{-1} N_{21} N_{11}^{-1} N_{12})^{-1} \text{ and} \tag{25}$$

$$Q_{21} = -N_{22}^{-1} N_{21} ((w_p + w_c)N_{11} - w_p N_{12} N_{22}^{-1} N_{21})^{-1} \ .$$

So the correlation coefficient C_{ij} is a function of w_p and w_c, i.e.

$$C_{ij} = f(w_p, w_c) \,, \tag{26}$$

where indices i and j are unknown indices in X_1 and X_2, respectively. For $w_c = 0$ (only phase is used, X_1 and X_2 are partly linear correlated) and $w_c = w_p$ (X_1 and X_2 are uncorrelated) there exists indices ij, so that

$$C_{ij} = f(w_p, w_c = 0) = 1 \quad \text{and} \quad C_{ij} = f(w_p, w_c = w_p) = 0 \,. \tag{27}$$

For phase and code combination, $w_c = 0.01 w_p$ can be selected, so that

$$C_{ij} = f(w_p, w_c = 0.01 w_p) \,. \tag{28}$$

Eqs.26, 27 and 28 indicate that for the correlated unknown pairs ij the correlation situation may not change much by combining the code to the phase because of the lower weight of the code related to the phase. A numerical test confirmed this conclusion.

7.5 Differential Solution of the Variance Equation

We denote the range and range rate function generally by ρ; their partial derivatives with respect to the orbit state vector are given in Xu (2003 §6.3 pp.86-90) and have the forms of

$$\frac{\partial \rho}{\partial \vec{r}}, \frac{\partial \rho}{\partial \dot{\vec{r}}}, \quad or \quad \frac{\partial \rho}{\partial \vec{X}} \,. \tag{29}$$

Therefore, the orbit parameter related parts in the linearised GPS observation equations are then

$$\frac{\partial \rho}{\partial (\vec{r}, \dot{\vec{r}})} \frac{\partial (\vec{r}, \dot{\vec{r}})}{\partial \vec{y}} \Delta \vec{y}^T, \quad or \quad \frac{\partial \rho}{\partial \vec{X}} \frac{\partial \vec{X}}{\partial \vec{y}} \Delta \vec{y}^T \,, \tag{30}$$

where

$$\vec{y} = \left(\vec{X}_0, \vec{Y}\right), \quad \Delta \vec{y}^T = \left(\Delta \vec{X}_0, \Delta \vec{Y}\right)^T, \quad \frac{\partial \vec{X}}{\partial \vec{y}} = \frac{\partial \vec{X}}{\partial (\vec{X}_0, \vec{Y})} \ .$$

$\vec{X}$, $\vec{Y}$ are the state vector of satellite and the parameter vector of the force models, and index $_0$ denotes the related initial vectors of time t_0. $\vec{y}$ is the total unknown vector of the orbit determination problem, the related correction vector is $\Delta \vec{y} = \vec{y} - \vec{y}_0$, and $\Delta \vec{X}_0$ is the correction vector of the initial state vector. The partial derivatives of $\vec{X}$ with respect to $\vec{y}$ is called transition matrix which has the dimension of $6 \times (6+n)$, where n is the dimension of vector $\vec{Y}$. The equation of motion of the satellite is (cf. Xu 2003 pp.254 Eq.11.134)

$$\begin{cases} \dot{\vec{X}}(t) = \vec{F} \\ \vec{X}(t_0) = \vec{X}_0 \end{cases}. \tag{31}$$

The partial derivatives of the equation of motion of the satellite with respect to the vector $\vec{y}$ are

$$\frac{\partial \dot{\vec{X}}(t)}{\partial \vec{y}} = \frac{\partial \vec{F}}{\partial \vec{y}} = \frac{\partial \vec{F}}{\partial \vec{X}} \frac{\partial \vec{X}}{\partial \vec{y}} + \left(\frac{\partial \vec{F}}{\partial \vec{y}} \right)^*, \tag{32}$$

where the superscript * denotes the partial derivatives of $\vec{F}$ with respect to the explicit parameter vector $\vec{y}$ in $\vec{F}$, and

$$D(t) = \left(\frac{\partial \vec{F}}{\partial \vec{X}} \right) = \begin{pmatrix} 0_{3\times3} & E_{3\times3} \\ \frac{1}{m}\frac{\partial \vec{f}}{\partial \vec{r}} & \frac{1}{m}\frac{\partial \vec{f}}{\partial \dot{\vec{r}}} \end{pmatrix} = \begin{pmatrix} 0_{3\times3} & E_{3\times3} \\ A(t) & B(t) \end{pmatrix},$$

$$C(t) = \left(\frac{\partial \vec{F}}{\partial \vec{y}} \right)^* = \begin{pmatrix} 0_{3\times6} & 0_{3\times n} \\ 0_{3\times6} & \frac{1}{m}\frac{\partial \vec{f}}{\partial \vec{Y}} \end{pmatrix} = \begin{pmatrix} 0_{3\times(6+n)} \\ G(t) \end{pmatrix}, \tag{33}$$

where E is an identity matrix; the partial derivatives are discussed and derived in Xu (2003, pp.267-276). It is notable that the force parameters are not functions of t. Therefore the order of the differentiations in Eq.32 can be exchanged. Denoting transition matrix by $\Phi(t, t_0)$, then Eq.32 turns out to be

$$\frac{d\Phi(t,t_0)}{dt} = D(t)\Phi(t,t_0) + C(t). \tag{34}$$

Eq.34 is called differential equations of the transition matrix or variational equations (cf., e.g., Montenbruck and Gill 2000). Denoting

$$\Phi(t,t_0) = \begin{pmatrix} \Psi(t,t_0) \\ \dot{\Psi}(t,t_0) \end{pmatrix}, \tag{35}$$

an alternate expression of Eq.34 can be obtained by substituting Eqs.35 and 33 into Eq.34

$$\frac{d^2\Psi(t,t_0)}{dt^2} = A(t)\Psi(t,t_0) + B(t)\frac{d\Psi(t,t_0)}{dt} + G(t). \tag{36}$$

The initial value matrix is (initial state vector does not depend on force parameters):

$$\Phi(t_0,t_0) = \begin{pmatrix} E_{6\times6} & 0_{6\times n} \end{pmatrix}. \tag{37}$$

That is, in the GPS observation equations, the transition matrix has to be obtained by solving initial value problem of differential equation 34 or 36. The problem is traditionally solved by integration. However, by carefully studying the integrator, one may notice that by integration from t_0 to t_1 several function values at the pints between t_0 to t_1 have to be computed. This is not the same as the orbit integration, in which the right function is the function of known forces. Here the right function includes the unknown function $\Phi(t, t_0)$. So without approximation, the integrator cannot be applied directly for solving the variation equation. This fact leads to a new development of a differentiation method to solve the variation equation and is given first time in Xu (2003 pp.254-257). For convenience of readers, the algorithm is outlined below.

Eq.36 is a matrix differential equation system of size 3×(6+*n*). Because *A(t)* and *B(t)* are 3×3 matrices, the differential equations are independent from column to column. That is, we need just to discuss the solutions of the equations of one column. For column *j*, the Eqs.36 and 37 are

$$\frac{d^2\Psi_{ij}(t)}{dt^2}=\sum_{k=1}^{3}\left(A_{ik}(t)\Psi_{kj}(t)+B_{ik}(t)\frac{d\Psi_{kj}(t)}{dt}\right)+G_{ij}(t),\quad i=1,2,3\,, \tag{38}$$

$$\begin{pmatrix}\Psi_{ij}(t_0)\\ \dot{\Psi}_{ij}(t_0)\end{pmatrix}=\begin{pmatrix}\delta_{ij}\\ \delta_{(i+3)j}\end{pmatrix},\quad i=1,2,3\,,\qquad \delta_{kj}=\begin{cases}1 & if\ k=j\\ 0 & if\ k\neq j\end{cases},$$

where index *ij* denotes the related element of the matrix. For time interval $[t_0,\ t]$ and differentiation step $h=(t-t_0)/m$, one has $t_n=t_0+nh,\ n=1,\ldots,m$ and

$$\left.\frac{d^2\Psi_{ij}(t)}{dt^2}\right|_{t=t_n}=\frac{\Psi_{ij}(t_{n+1})-2\Psi_{ij}(t_n)+\Psi_{ij}(t_{n-1})}{h^2},\quad i=1,2,3\,,$$

$$\left.\frac{d\Psi_{ij}(t)}{dt}\right|_{t=t_n}=\frac{\Psi_{ij}(t_{n+1})-\Psi_{ij}(t_{n-1})}{2h},\quad \Psi_{ij}(t)\big|_{t=t_n}=\Psi_{ij}(t_n),\qquad i=1,2,3\,. \tag{39}$$

Then Eq.38 turns out to be

$$\begin{aligned}&\Psi_{ij}(t_0)=\Psi_{ij}(t_0),\qquad \Psi_{ij}(t_1)=\Psi_{ij}(t_0)+h\,\dot{\Psi}_{ij}(t_0),\quad i=1,2,3\,.\\ &\frac{\Psi_{ij}(t_{n+1})-2\Psi_{ij}(t_n)+\Psi_{ij}(t_{n-1})}{h^2}=\\ &\sum_{k=1}^{3}\left(A_{ik}(t_n)\Psi_{kj}(t_n)+B_{ik}(t_n)\frac{\Psi_{kj}(t_{n+1})-\Psi_{kj}(t_{n-1})}{2h}\right)+G_{ij}(t_n),\quad i=1,2,3\end{aligned}\,, \tag{40}$$

where $n=1,\ 2,\ \ldots,\ m-1$. For $i=1,\ 2,\ 3$ and the sequential number *n*, there are three equations and three unknowns of time t_{n+1}; so that the initial values problem has a set of unique solutions sequentially. Eq.40 can be rewritten as

$$\left(\frac{E}{h^2}-\frac{B(t_n)}{2h}\right)\begin{pmatrix}\Psi_{1j}(t_{n+1})\\ \Psi_{2j}(t_{n+1})\\ \Psi_{3j}(t_{n+1})\end{pmatrix}=\begin{pmatrix}R_1\\ R_2\\ R_3\end{pmatrix}, \tag{41}$$

where

$$\begin{pmatrix}R_1\\ R_2\\ R_3\end{pmatrix}=\left(\frac{2E}{h^2}+A(t_n)\right)\begin{pmatrix}\Psi_{1j}(t_n)\\ \Psi_{2j}(t_n)\\ \Psi_{3j}(t_n)\end{pmatrix}-\left(\frac{E}{h^2}+\frac{B(t_n)}{2h}\right)\begin{pmatrix}\Psi_{1j}(t_{n-1})\\ \Psi_{2j}(t_{n-1})\\ \Psi_{3j}(t_{n-1})\end{pmatrix}+\begin{pmatrix}G_{1j}(t_n)\\ G_{2j}(t_n)\\ G_{3j}(t_n)\end{pmatrix}. \tag{42}$$

For $n=1,\ \ldots,\ m-1$, above equations are solvable. It is notable that the three matrices

$$\left(\frac{E}{h^2}-\frac{B(t_n)}{2h}\right),\quad \left(\frac{2E}{h^2}+A(t_n)\right),\quad \left(\frac{E}{h^2}+\frac{B(t_n)}{2h}\right)$$

are independent from the column number *j*. The solutions of Eq.41 are vectors

$$\begin{pmatrix}\Psi_{1j}(t_{n+1})\\ \Psi_{2j}(t_{n+1})\\ \Psi_{3j}(t_{n+1})\end{pmatrix}\quad \text{and}\quad \begin{pmatrix}\dot{\Psi}_{1j}(t_{n+1})\\ \dot{\Psi}_{2j}(t_{n+1})\\ \dot{\Psi}_{3j}(t_{n+1})\end{pmatrix},\quad n=1,\ldots,m-1\,, \tag{43}$$

where the velocity vector can be computed using definition of Eq.39. Solving the equations of all column *j*, the solutions of the initial value problem of Eqs.36 and 37 can be obtained. The needed results are related to time t_n, which can be obtained by averaging the values of t_{n+1} and t_{n-1}.

This differential algorithm for solving the variation equation can be used as initiator of the integration method for solving the same problem or can be used to replace the integration method and to solve the variation equation directly.

7.6 Adjustment Models of the Solar Radiation and Atmospheric Drag

Based on the commonly used solar radiation and atmospheric drag models of the orbit theory, a so-called disturbance coordinate system is introduced and the to be adjusted models are proposed. A simulation study shown that it is very advantageous to describe the models in the disturbance coordinate system. The traditional models to be adjusted are partly not well formulated and the bias parameters are partly over-parameterised. The new models owns much better ability to absorb the un-modelled errors with fewer parameters. The new models are then implemented in a software and shown excellent performance.

7.6.1. Introduction of the models

Solar radiation pressure model

Solar radiation pressure is force acting on the satellite's surface caused by the sunlight. The radiation force can be represented as (Seeber 1993)

$$\vec{f}_{\text{solar}} = m\gamma P_s C_r \frac{S}{m}\frac{r_{\text{sun}}^2}{\left|\vec{r}-\vec{r}_{\text{sun}}\right|^2}\vec{n} \quad \text{and} \quad \vec{n} = \frac{\vec{r}-\vec{r}_{\text{sun}}}{\left|\vec{r}-\vec{r}_{\text{sun}}\right|} \tag{44}$$

where γ is the shadow factor, P_s is the luminosity of the Sun, C_r is the surface reflectivity, (S/m) is the surface to mass ratio of the satellite, $\vec{r}$ and $\vec{r}_{sun}$ are the geocentric vectors of the satellite and the Sun, the related distances are r and r_{sun}, $\vec{n}$ is the identity vector pointed from the Sun to the satellite. Usually, P_s has the value of 4.5605×10^{-6} Newton/meter, C_r has values from 1 to 2, 1 is for the complete absorption of the sunlight, and for aluminium, $C_r = 1.95$.

Because of the complex shape of the satellite, the use of constant reflectivity and homogenous luminosity of the Sun, as well as the existence of indirect solar radiation (reflected from the Earth's surface), the model of Eq.44 is not accurate enough for precise purposes and will be used as a first order approximation.

An adjustment model for fitting solar radiation effects can be represented as (cf. Beutler et al. 1994; Rothacher & Mervart 1996)

$$\vec{f}_{\text{solar-force}} = \vec{f}_{\text{solar}} + \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} 1 \\ \cos u \\ \sin u \end{pmatrix}. \tag{45}$$

Where 9 parameters are used to model the solar radiation force error for every satellite. That is, one bias and two periodic coefficients are used in every coordinate direction to model and absorb the un-modelled errors.

Atmospheric drag model

Atmospheric drag is the disturbance force acting on the satellite's surface caused by the air. Air drag force can be represented as (Seeber 1993, Liu and Zhao 1979)

$$\vec{f}_{\text{drag}} = -m\frac{1}{2}\left(\frac{C_{\text{d}}S}{m}\right)\sigma\left|\dot{\vec{r}}-\dot{\vec{r}}_{\text{air}}\right|^2\vec{p} \qquad \text{and} \qquad \vec{p}=\frac{\dot{\vec{r}}-\dot{\vec{r}}_{\text{air}}}{\left|\dot{\vec{r}}-\dot{\vec{r}}_{\text{air}}\right|} \tag{46}$$

where S is the cross section (or effective area) of the satellite, C_{d} is the drag factor, m is the mass of the satellite, $\dot{\vec{r}}$ and $\dot{\vec{r}}_{air}$ are the geocentric velocity vectors of the satellite and the atmosphere, and σ is the density of the atmosphere, $\vec{p}$ is the atmospheric drag identity vector of the $\dot{\vec{r}}-\dot{\vec{r}}_{air}$. Usually, S has a value of $1/4$ of the outer surface area of the satellite, and C_{d} has labour values of 2.2 ±0.2. The velocity vector of the atmosphere can be modelled by

$$\dot{\vec{r}}_{\text{air}} = k\vec{\omega}\times\vec{r} = k\omega\begin{pmatrix}-y\\ x\\ 0\end{pmatrix}, \tag{47}$$

where $\vec{\omega}$ is the angular velocity vector of the Earth's rotation, and $\omega=\left|\vec{\omega}\right|$, k is the atmospheric rotation factor, and $\vec{r}=(x, y, z)^{\text{T}}$. For the lower layer of the atmosphere, $k=1$, i.e., the lower layer of the atmosphere, is considered rotating with the Earth. For the higher layer, $k=1.2$, because the higher ionosphere is accelerated by the Earth's magnetic field.

An adjustment model for fitting the atmospheric drag force model can be represented as (similar with the Eq.2 (cf. Xu 2003 p238 Eq.11.85))

$$\vec{f}_{\text{air-drag}} = \vec{f}_{\text{drag}} + \begin{pmatrix}b_{11} & b_{12} & b_{13}\\ b_{21} & b_{22} & b_{23}\\ b_{31} & b_{32} & b_{33}\end{pmatrix}\begin{pmatrix}1\\ \cos v\\ \sin v\end{pmatrix}. \tag{48}$$

Where $v = \omega + \beta$, ω is the angle of perigee of the satellite, β is the angle between radial and tangential directions of the satellite orbit.

Because β is not a constant for any ellipse orbit, cos$u$ and sin$u$ are independent from $\cos v$ and $\sin v$. However, the six bias parameters are pair-wise linearly correlated. Only three of the biases can be determined. In this way the two models are partly inseparable. Further problem is that the models are very empirical ones which set the main disturbance frequency as that of the motion of the satellite.

7.6.2 The disturbance coordinate system and error models

Solar radiation force vector is pointed from the Sun to the satellite. If the shadow factor is computed exactly, the luminosity of the Sun is a constant, and the surface reflectivity of the

satellite is a constant, then the length of the solar force vector can be considered as a constant, because

$$\frac{r_{sun}^2}{(r_{sun}+r)^2} \le \frac{r_{\mathrm{sun}}^2}{\left|\vec{r}-\vec{r}_{\mathrm{sun}}\right|^2} \le \frac{r_{sun}^2}{(r_{sun}-r)^2}, \tag{49}$$

and

$$\frac{r_{sun}^2}{(r_{sun}\pm r)^2} = \left(\frac{r_{sun}}{r_{sun}\pm r}\right)^2 \approx \left(1\mp\frac{r}{r_{sun}}\pm\ldots\right)^2 \approx 1\mp\frac{2r}{r_{sun}} \approx 1\mp 3\times 10^{-5}, \tag{50}$$

where r and r_{sun} are the geocentric distances of the satellite and the Sun, respectively. Any bias error in P_{s}, C_{r} and (S/m) may cause a model error of $\alpha\vec{f}_{solar}$, where α is a parameter. So the $\alpha\vec{f}_{solar}$ can be considered as main error model of the solar radiation. Because the ratio of the geocentric distances of the satellite and the Sun is so small so that the direction and distance changes of the Sun-satellite vector are negligible. With the motion of the Sun, the solar radiation force vector changes its direction with the time in the ECSF (Earth-Centred-Space-Fixed) coordinate system ca. 1 degree per day. Such an effect can only be considered as a small drift, not a periodical changing for the orbit determination. To model such an effect in ECSF system one needs three bias parameters in three coordinate axes, and three drift terms instead of a few periodical parameters. It is obvious that to model such an effect in the direction of $\vec{n}$, just one parameter α is needed. And we see that the traditional solar radiation adjustment model may not able to fit the main error model well in order to absorb the effect.

Therefore, it is very advantageous to define a so-called disturbance coordinate system as follows: the origin is the geo-centre, the three axes are defined by $\vec{r}$ (redial vector of the satellite), $\vec{n}$ (the Sun-satellite identity vector) and $\vec{p}$ (the atmospheric drag identity vector). These three axes are always in the main disturbance directions of the indirect solar radiation (reflected from the Earth's surface), direct solar radiation and atmospheric drag, respectively. This coordinate system is not a Cartesian one and the axes are not orthogonal to each other. The parameters in individual axes are mainly used to model the related disturbance effects, and meanwhile to absorb the remained error of other un-modelled effects.

In the atmospheric drag model Eq.3, the velocity vector of the atmosphere is always perpendicular to the z-axis of the ECSF coordinates and the satellite velocity vector is always in the tangential direction of the orbit. The variation of the term $\left|\dot{\vec{r}}-\dot{\vec{r}}_{\mathrm{air}}\right|^2$ (denoted by g^2) is dominated by the direction changes of the velocity vectors of the satellite and the atmosphere. Any bias error in S (effective area of the satellite), C_{d} (drag factor) and σ (density of the atmosphere) may cause a model error of $\mu\vec{f}_{drag}$, where μ is a parameter. So the $\mu\vec{f}_{drag}$ can be considered as main error model of the un-modelled atmospheric drag. For simplification our discussion, we consider the velocities of the satellite and atmosphere are constants, and call the satellite positions with max(z) and –max(z)) the highest and lowest points, respectively. Satellite at the lowest point, the two velocity vectors are in the same direction and therefore the g reaches the minimum. At the ascending node the two vectors have the maximum angle of inclination i and the g reaches the maximum. Then g reaches the minimum again at the highest point and reaches the maximum again at the descending node, and at the end reaches the minimum at the lowest point. It is obvious, besides the constant part, g has a dominant periodical component of cos2f and sin2f. Where f is the true anomaly of the satellite. So we see that the traditional atmospheric drag adjustment model may not able to fit the main error model well in order to absorb the effect.

7.6.3 Numerical simulation and models analysis

Numerical simulated data of the solar radiation error model $\alpha \vec{f}_{solar}$ and the atmospheric drag error model $\mu \vec{f}_{drag}$ are carried out based on a simulated GPS satellite orbit and solar orbit data. Keplerian elements for GPS satellite are selected: semi-major axis a = 2600km, inclination i = 55°, ascending node Ω = arbitrary, angle of perigee ω = arbitrary, eccentricity e = 0.005 ~ 0.0221, true anomaly f = 0 ~ 2π. The orbital period T is assumed as 12 hours and Keplerian elements for the Sun are selected: semi-major axis a = 1 AU (astronomical units), inclination i = 23°, eccentricity e = 0.0167, true anomaly $f = f_0 + (0 \sim 2\pi T)$/(year in hours), f_0 is an arbitrary constant between 0 ~ 2π.

Theoretically speaking, it is obvious that the errors computed from above two error models can be completely absorbed by the two model parameters α and μ . Then the dada in three axes of the ECSF coordinate system are spectral analysed to find out the frequency properties. FFT (fast Fourier transformation) method are used. The theoretical analysis about the properties of the main error models are very well shown in the FFT results.

Based on numerous analysis of the data, conclusions are obtained. The solar radiation pressure error model can be represented alternatively by

$$\alpha \vec{f}_{solar} = \begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \\ a_3 & b_3 \end{pmatrix} \begin{pmatrix} 1 \\ t \end{pmatrix}, \tag{51}$$

where b-terms are very small. The atmospheric drag error model can be represented alternatively by

$$\mu \vec{f}_{drag} = \left[a + b\varphi(2\omega)\cos(2f) + c\varphi(3\omega)\cos(3f) + d\varphi(\omega)\cos f \right] \vec{p}, \tag{52}$$

where

$$\varphi(k\omega) = \begin{cases} \sin k\omega & if \cos k\omega = 0 \\ \dfrac{1}{\cos k\omega} & if \cos k\omega \neq 0 \end{cases}, \qquad k = 1, 2, 3 \tag{53}$$

where ω is the angle of perigee and f is the true anomaly of the satellite, a, b, c and d are model parameters to be determined. According to the simulation, a-term and b-term are the most significant terms. The amount of d is just about 1% of the amount of c, and the amount of c is about 1% of that of b.

7.6.4 Summary and Conclusions

It is advantageous to represent the solar radiation pressure and atmospheric drag error models in the so-called disturbance coordinate system. The main error models are considered theoretically to be able to be presented by models $\alpha \vec{f}_{solar}$ and $\mu \vec{f}_{drag}$ respectively. The simulation study shown that the traditional models are very empirical ones and are not able to be used even for

fitting the main errors, and the periodical terms are either not reasonable ones or with wrong frequencies. Two alternative adjustment models represented by parameters in three axes of the ECSF system are proposed based on intensive analysis. The parameters used are fewer and without problem of over-parameterisation. For the remaining error model, the modulation shall be based on an residual analysis instead of empirical setting.

The proposed models are implemented in software and shown good performance.

7.7 Intelligent Kalman Filtering Technique

Based on the following introduction of Kalman filter and the filter using velocity information, the needs of the intelligent Kalman filtering will be discussed and the algorithm will be outlined.

7.7.1 Introduction of Kalman Filter

The principle of the classical Kalman filter can be summarised as below (Yang et al. 1999):

The linearised observation equation system can be represented by

$$V_i = L_i - A_i X_i \ , \quad P_i \ , \tag{54}$$

where

L : observational vector of dimension m,
A : coefficient matrix of dimension $m \times n$,
X : unknown vector of dimension n,
V : residual vector of dimension m,
n : number of unknowns,
m : number of observations,
i : sequential index, $i = 1, 2, 3, \ldots$, and
P_i : weight matrix of index i.

Suppose system equations are known and can be presented as

$$U_i = X_i - F_{i,i-1} X_{i-1} \ , \qquad i = 2, 3, \ldots \ , \tag{55}$$

where

F : transition matrix of dimension $n \times n$, and
U : residual vector of dimension n.

U and V are un-correlated and have zero expectations. Using the covariance propagation law, one has from Eq.55

$$Q(X_i) = F_{i,i-1} Q(X_{i-1}) (F_{i,i-1})^T + Q_U \ . \tag{56}$$

The normal Eq.54 can be formed as

$$M_i X_i = B_i \ . \tag{57}$$

For the initial step or epoch, i.e., $i = 1$, Eq.54 has the solution under the least squares principle

$$\widetilde{X}_i = Q_i B_i \ , \qquad \text{where} \qquad Q_i = M_i^{-1} \ , \tag{58}$$

and here one will assume

$$\widetilde{Q}_i = Q_i \ , \tag{59}$$

where $\widetilde{X}_i$ and $\widetilde{Q}_i$ are called estimated values. Using the estimated values and transition matrix, one can predict the unknown values and covariance matrix of the next epoch (say $i = 2$):

$$\underline{X}_i = F_{i,i-1}\,\widetilde{X}_{i-1} \quad \text{and} \tag{60}$$

$$\underline{Q}_i = F_{i,i-1}\widetilde{Q}_{i-1}(F_{i,i-1})^T + Q_U \ , \tag{61}$$

where $\underline{X}_i$ and $\underline{Q}_i$ are called predicted values (vector and matrix). Then estimated values of this epoch can be calculated by

$$\widetilde{X}_i = \underline{X}_i + K(L_i - A_i\underline{X}_i) \ , \tag{62}$$

$$\widetilde{Q}_i = (E - KA_i)\underline{Q}_i \quad \text{and} \tag{63}$$

$$K = \underline{Q}_i A_i^T (A_i \underline{Q}_i A_i^T + Q_V)^{-1} \ , \tag{64}$$

where K is the gain matrix, $Q_V = P_i^{-1}$.

For the next sequential step i, the predicted values have to be computed by using Eqs.60 and 61, and the estimated values can be computed by using Eqs.62 and 63. Such an iterative process is called Kalman filtering.

In classical Kalman filtering, it is assumed that for the problem of Eq.54 there exists a system transition matrix $F_{i,i-1}$ in Eq.55 and the cofactor Q_U. Therefore, the estimated values in the Kalman filter process are dependent on $F_{i,i-1}$ and Q_U. The transition matrix shall be based on strengthened physical models, and the cofactor shall be well-known or reasonably given. If the system description is accurate enough, of course Kalman filtering will lead to a more precise solution. However, if the system is not sufficiently well-known, the results of Kalman filter will sometimes not converge to the true values (divergence). Furthermore, a kinematic process is generally difficult to be precisely represented by theoretical system equations. However, for a dynamic process (like on-board GPS for satellite to satellite tracking or orbit determination) the system equation can be well-formulated (by an orbital equation of motion). Another problem of Kalman filtering is the strong dependency of the given initial values. Many studies have been made in this area to overcome the above-mentioned shortages.

The development of the theory leads to a so-called adaptive robust Kalman filter in which Eq.62 can be presented by (Yang et al. 2001)

$$\widetilde{X}_i = \underline{X}_i + Q_{\underline{X}_i} A_i^T (A_i Q_{\underline{X}_i} A_i^T + \alpha Q_V)^{-1}(L_i - A_i \underline{X}_i) \ . \tag{65}$$

Where α is the adaptive factor or adaptive diagonal matrix and Q_V is the robust weight matrix. Robust weight can be used to adjust the weight of observations according to the residuals and the adaptive matrix can be used to adjust how much the past information shall be taken into account in the filtering.

7.7.2 Kalman Filtering Using Velocity Information

As applied in KSGSoft (Xu et al. 1998), velocity information from the differential Doppler can be used to describe the system which is needed in Kalman filtering. Whether the receiver is moving or resting, the differential Doppler includes information about the motion state of the receiver. Therefore, using velocity information as a system description should be significantly better than any empirical model.

The principle of Kalman filtering using velocity information can be outlined as follows (Xu 2003):

For the initial (or predicted) vector $\overline{Z}$, the normal equation of the phase observation equation can be formed by

$$M_z Z = B_z \ , \quad Z = \begin{pmatrix} X \\ N \end{pmatrix} \ , \tag{66}$$

where M_z is the normal matrix, and B_z is the vector on the right side of the equation. These are formed by using initial vector $\overline{Z}$; Z includes sub-vector X (coordinates) and N (ambiguities and others). The estimated solution of Eq.66 is then

$$\widetilde{Z} = \widetilde{Q}_z B_z \ , \quad \widetilde{Q}_z = M_z^{-1} \ . \tag{67}$$

The normal equation of the differential Doppler observation equation (cf. Xu 2003, Eq.9.50 pp.197, only the velocity vector is unknown) can be formed by

$$M_{\dot{x}} \dot{X} = B_{\dot{x}} \ , \tag{68}$$

where $\dot{X}$ is the velocity vector of the receiver; it is also used as an index to denote the related normal matrix and vector on the right side of the equation. The solution of Eq.68 is then

$$\dot{X} = Q_{\dot{x}} B_{\dot{x}} \ , \quad Q_{\dot{x}} = M_{\dot{x}}^{-1} \ . \tag{69}$$

Thus for the next epoch, denoted as k, the predicted vector turns out to be

$$\overline{Z}(k) = \widetilde{Z}(k-1) + \dot{Z}(k-1) \cdot \Delta t \ , \tag{70}$$

where Δt is the time interval of the epoch $k-1$ and k, and

$$\dot{Z}(k-1) = \begin{pmatrix} \dot{X}(k-1) \\ 0 \end{pmatrix} . \tag{71}$$

Equation 70 indicates that the differential Doppler has to be used in Eq.69 as observations, because the velocity is considered an average one here. The related covariance matrix of the predicted vector is then

$$\overline{Q}_z(k) = \widetilde{Q}_z(k-1) + (\Delta t)^2 \begin{pmatrix} Q_{\dot{x}} & 0 \\ 0 & 0 \end{pmatrix} . \tag{72}$$

The weight matrix is

$$\overline{P}_z(k) = \overline{Q}_z^{-1}(k) \ . \tag{73}$$

The normal Eq.66 of epoch k is

$$M_z(k)Z(k) = B_z(k) \ , \tag{74}$$

and the Kalman filter solution of Eq.74 is then

$$\widetilde{Z}(k) = \widetilde{Q}_z(k)B_z(k) \ , \quad \widetilde{Q}_z(k) = (M_z(k) + \overline{P}_z(k))^{-1} \ . \tag{75}$$

It is notable that the normal equation of Eq.74 must be computed using the predicted vector $\overline{Z}(k)$ of Eq.70.

Repeating the steps from Eq.68 to 75 for the further epoch is a process of Kalman filtering using velocity information. The algorithm outlined above is suitable both for the kinematic and static data processing. This is true especially for static data processing, because the station has not been exactly assumed as fixed (as described by Eq.68); such an algorithm will modify the property of the strong dependency on the initial value of the Kalman filter. The forming of normal Eq.68 is an iterative process, i.e., the velocity information has to be used for forming the equation. Equation 68 represents a realistic system description.

7.7.3 The Theoretical Problem vs. Practical Requirement

Kalman filter has been broadly used in real time or sequential data processing, however almost all of them are case studies in navigation area. The reason why the on-hand Kalman filtering techniques cannot be generally used in the daily GPS navigation and monitoring is a theoretical one. The GPS observation equation describes that the position unknowns will be determined through measurements, whereas the system equation says that, well, we do not know the position unknowns, but we do know the position changing regulation. And this is not true in most cases. If one considers the coordinate change as a motion, then only the motion under the acting of the conservative force will obey the motion law of Newton. As soon as there exists non conservative and non negligible unknown disturbance, the motion will be very hard to be described by an equation given by persons.

For example, a person drives a car along a strait street without any traffic lamp and other traffics with a constant velocity. The motion can be described easily and the accumulated past information represents the smooth motion and can be used to estimate the motion in the future. However, now, the man sees the name of a side street and remembers that one of his old friends lives in the near; then the man decided to make a visit to his friend and slowed down and stoped the car quickly and derived his car in backward direction a few meters and stoped, then turned on the right. In such a case, the motion is not decided by any equation, but by the idea of the driver. It is nearly impossible to try to use some equations to describe the fantasy of the people. The attempt to try to forget the past information (adaptive) in such a case is a correct measure, however after the event, the filtering procedure will start again to use some kinds of empiric equations to describe the motion in the future; and this cannot be generally work well.

To use the velocity information to describe the system equation of the motion is a great step forward in the theoretical development of the Kalman filtering technique. This technique was developed by a cooperation between author and Dr. Yang during Yang's guest stay in GFZ in 1997 (Xu 2003). The GPS phase observable came always from an internal Doppler measurement, no matter the Doppler is outputted for use or not. So that as soon as there is no cycle slip happened, the differential Doppler can always be derived from the phase measurements afterward, and the differential Doppler described an average motion of receiver during the data sampling epochs. That is the system equation in such way is determined meanwhile by the additional measurements. For a non accelerated motion such a description would be good enough,

however, in case of high kinematic motion, the acceleration or even higher order acceleration have to be taken into account.

We do not have a few universal equations which could be used to describe any kind of motion, so the Kalman filtering technique cannot be used generally in our navigation and monitoring practice. However, if we by the way of surveying, meanwhile use the possibilities of GPS or other additional sensors to measure and determine the system equation, then we have designed a universal Kalman filter for GPS data processing and we call such a filter as an intelligent Kalman filter.

The more and more measurement are now made in a real time manner to monitor the real time motion. The development of an intelligent Kalman filter is an urgent desire of the navigation and positioning practice especially for the application of GPS and Galileo systems.

7.7.4 Algorithm of Intelligent Kalman Filter

The adaptive robust Kalman filter possess already a kind of intelligent. It can adjust the weight automatically due to the real residuals of the observation equations (called robust). It can decide if or not and how to adapt the past accumulated system information (called adaptive). However, the system equations are assumed known and fixed, and no matter how the motion is, the system equations will be always used. This is physically not correct and therefore may not be valid universally. Kalman filter using velocity information in GPS application makes it possible first time to determine the system equation instead of to define the system equation empirically. Even this is just an approximation to an average motion during the sampling epochs, however, this is a first order approximation and could be well fit to the non accelerated motion. The key point is that in this way the system equation could be determined and updated automatically. For a general motion, especially the high kinematic and high dynamic disturbed motion, the system described by an average motion could be not enough. Therefore, an intelligent Kalman filtering is to be able to adjust the weight (robust) and to adapt accordingly the past information (adaptive), and to determine the system equation generally (intelligent).

The average velocity is a kind of smoothed velocity. Within the two sampling epochs, the average motion is not precise enough for an accelerated motion. However, the acceleration derived from the two adjacent average motion may have high quality because of the smoothing. So we may use the past information of the average velocity to derive the acceleration or even higher order ones. The problem is that in this case the system equation will be not a linear one. This will be not a problem for predicting the next values of the unknowns, however a problem for predicting the next covariance matrix of the unknowns.

The method of system equation determination by using GPS for general navigation and positioning problem has to be carefully studied and the concept of such an intelligent filtering has to be worked out. How the statistic information will be propagated and the precision can be estimated exactly have to be studied. In general, the so-called intelligent Kalman filtering is a no linear process. The precision estimation is important because it is the criterion for the robust weight and adaptation as well as the judge of goodness of the system determination.

An well designed MFGsoft which provide stable and homogenous solutions without using empiric and partly un-physical a priori information requires a careful study of such an intelligent Kalman filter for its applications.

7.8 Forming Independent Baselines and Extended Double Differencing

In order to extend the way of double differencing forming, we have to discuss first how to form an independent and optimal baseline network.

It is well-known that for a network with n stations there are n-1 independent baselines. An independent baseline network can be stated in words: all stations are connected through these baselines, and the shortest way from one station to any other station through these baselines is unique. Generally speaking, a shorter baseline leads to a better common view of the satellites. Therefore the baseline should be formed so that the length of the baseline falls as short as possible. For a network, an optimal choice should be that the summation of weighted lengths of all independent baselines should be minimum. This is exactly a mathematic problem called minimum spanning tree (cf. e.g. Wang et al. 1977).

There exists algorithms and software to solve this minimum spanning tree problem. Therefore we just show an example here. An IGS network with ca. 100 stations and the related optimal and independent baseline tree is shown in Fig1. The average length of the baselines is ca. 1300 km. The maximum distance is ca. 3700 km.

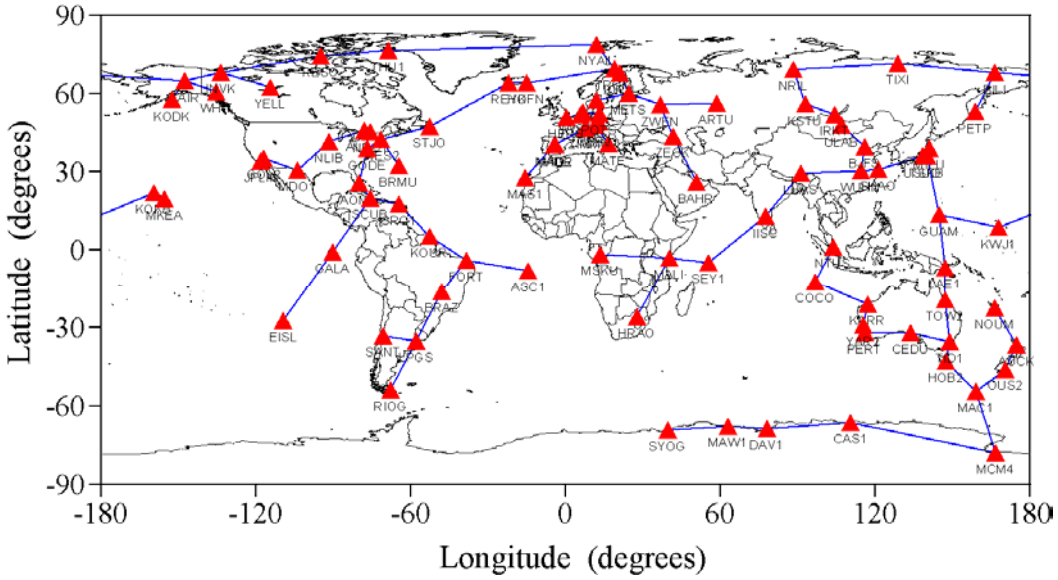


Fig.1: Optimal independent baseline network

In the traditional double differencing model, the un-paired GPS observations of every designed baseline have to be omitted because of the requirement of differencing forming. The example in Xu (2004) shown that, if the differencing is not limited by baseline design, many observations have not to be cancelled out by differencing forming. An optimal way of double differencing forming should be based on an optimal baseline design to form the differencing first, then, without limitation of the baseline design, to check for the un-paired observations in order to form possible differencing. This measure is useful for rising the data using rate by differencing method and for a better fulfilment of the first equivalent condition (the same data should be used).

7.9 Standard Models and Algorithms

The standard models and algorithms used are summarised and theoretically carefully verified in Xu (2003) according to the experience of KSGSoft and later intensive study. For convenience of the readers, the related contents in the literature are listed as follows.

Models: tropospheric models (pp.51-57); ionospheric model (pp.39-50); relativity model (pp.57-61); Earth tide model (pp.62-65); ocean loading tide model (pp.68-71); satellite mass centre model (pp.78-81); solar radiation model (pp.232-234); atmospheric drag model (pp.355-237); geopotential disturbance (pp.223-226); tidal potential disturbance (pp.228-230); multi-bodies disturbance (pp.237); dynamic orbit fitting model (pp.250-253).

Algorithms: equivalent algorithm (pp.132-134, 107-117,193); undifferenced algorithm (pp.100, 109, 191-192); differencing algorithms (pp.101-107); diagonalisation algorithm (pp. 172); independent parameterisation (see above §7.4, Xu 2004); optimal baselines and data conditions (see above §7.4, Xu 2004); cycle slip detection (pp.151-153); ambiguity initialisation (see source code); ambiguity search criterion (pp.153-171); least squares adjustment (pp.119-121); block-wise elimination (pp.127-131); sequential least squares (pp.123); a priori information (144-147); Kalman filter (pp.135-143); intelligent Kalman filter (see above §7.7); differential solution of variance equation (pp.256-257); orbit integrators (pp.257-259); differential Doppler and velocity determination (pp.98-99, 153, 195), single point positioning (pp.186-190).

Tools: coordinates transformers (pp.3-13); time systems transformers (pp.13-15); the Sun-Moon-planets orbit creator (pp.240-243); broadcast orbit transformer (pp.28-30); interpolation and integration (pp.30, 257); Gauss-Jordan and Cholesky inversions; Helmert transformation (pp.6-7); mapping functions (pp.47-49, 55-56); flight state computation (pp.208-209); spectral analysis methods (Xu 1992); statistic analysis; graphic representation;

8. Numerical Examples

8.1 Data and Station Network

For test run of the MFGsoft, the IGS GPS data of 47 stations measured on May 1st 2002, and static/kinematic GPS data as well as CHAMP on-board GPS data are used. A graphic of the used IGS station network is given below (Fig.2).

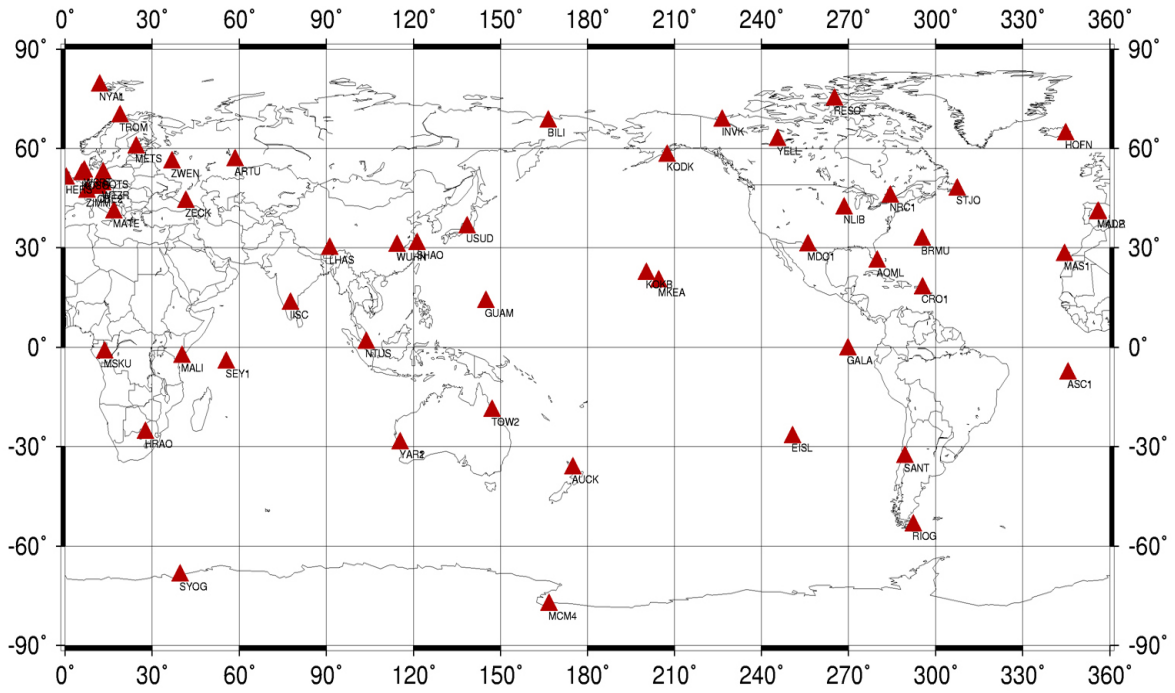


Fig.2: Used IGS Station Network

The numerical example of an IGS network with 47 stations and one day's observations shown that 87.9% of all data is used in double differencing forming based on the optimal baseline design, whereas 99.1% of all data is used in the extended method of double differencing forming.

8.2 Numerical Examples and Internal Test

Numerous examples are computed using MFGsoft to have broad internal tests. They are listed below and outlined briefly.

8.2.1 Solutions in ECEF and ECSF coordinate systems

For the global network monitoring problem with fixed orbit, the solutions are computed in ECEF and ECSF coordinate systems, respectively. The solutions are the same except negligible differences. This indicates that the coordinate and variables transformation as well as the corrections are well done in the two systems.

8.2.2 Equivalent Properties

For the global network monitoring problem with fixed orbit, the solutions are computed with equivalent algorithm, un-differential algorithm and double differencing algorithm, respectively. The solutions are generally the same except negligible differences. These results ensure that the equivalent theory hold numerically too.

8.2.3 Data Condition and Independent Parameterisation

For the global network monitoring problem with fixed orbit, the solutions are computed with and without the data condition and independent parameterisation, respectively. The inverse process are stable with data condition and independent parameterisation and the results are homogenous. Whereas without the condition and independent parameterisation, the inverse process is very instable and quit often fails by inversion of the normal matrix and the results are inhomogeneous and obviously incorrect.

8.2.4 Phase and Phase-Code Solutions

For the global network monitoring problem with fixed orbit, the solutions are computed for phase only solution and phase-code combined solution, with and without data condition, respectively. The solutions shown that the phase-code combination does not change much the singularity problem and confirmed the theoretical analysis mentioned in Section 7.4.2. With the data condition and independent parameterisation, the solutions are a little bit different in an order of ca. 1--2%. This may be explained as the consequence of the lower weight of the code.

8.2.5 Velocity Determination

Kinematic and static results of a local network are not given here due to the reasons. Velocity determination of a static station is given in Fig.3. The mean value and standard deviation are both less than 2cm/second.

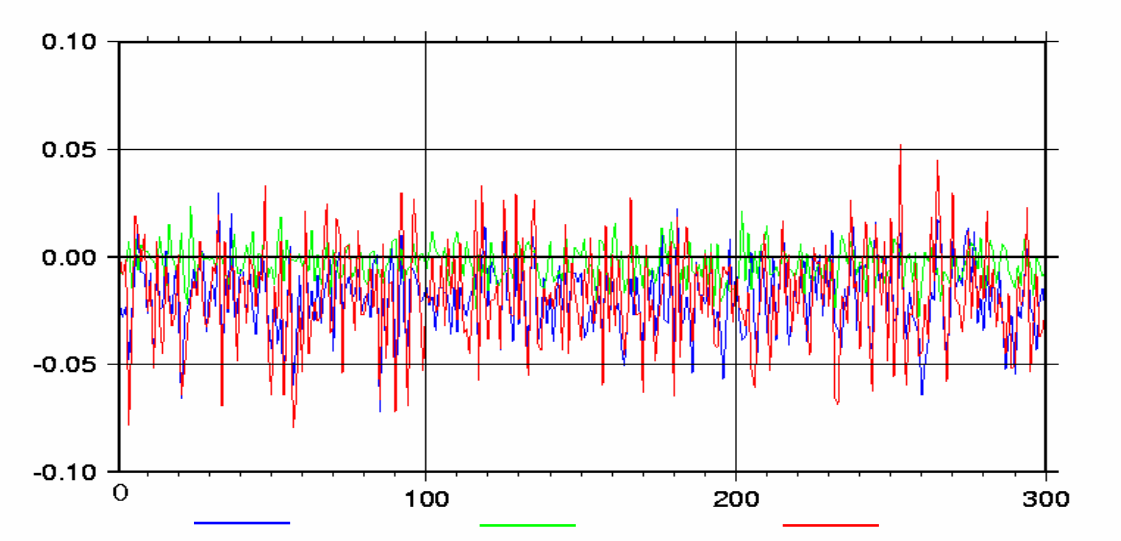


Fig.3: Kinematic velocity of a static station (blue-V_x, green-V_y, red-V_z, units: m/s)

LEO on-board GPS determined velocity is given in Fig.4 without external comparison.

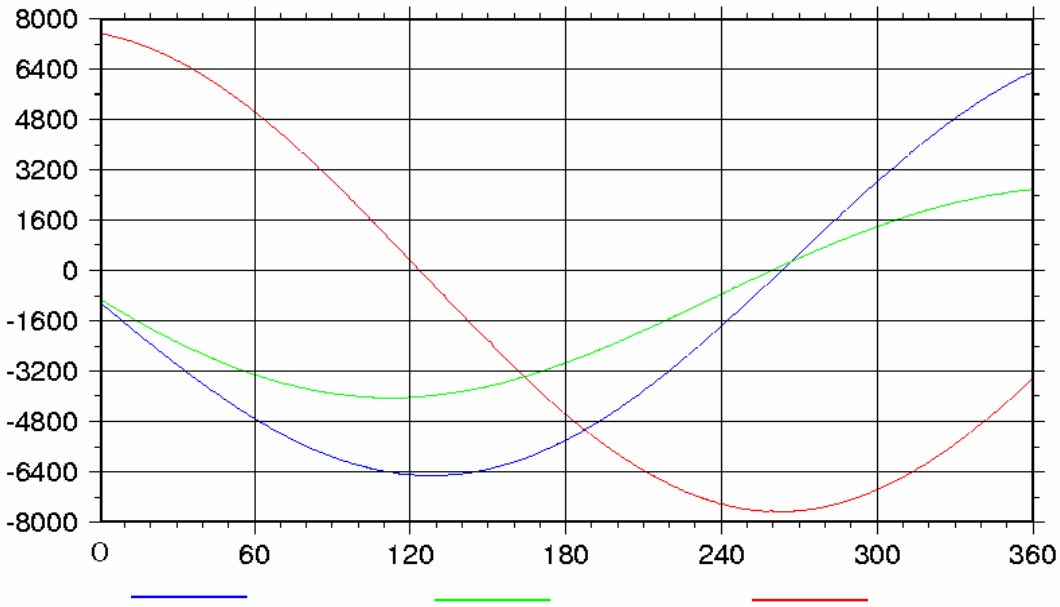


Fig.4: LEO satellite on-board GPS determined velocity (blue-V_x, green-V_y, red-V_z, units: m/s)

8.2.6 Regional Network Monitoring with Orbit Correction

Solving a regional (a part of a global) network monitoring problem with orbit correction is realised by using MFGsoft. The solution process is stable and the results are homogenous. However, the results are obviously depended on the model used for the orbit correction. How to model the orbit correction of a regional network problem has to be further studied and the experience has to be added to the software.

8.2.7 Global Network Monitoring with Kinematic/Dynamic Orbit Determination

Solving a global network monitoring problem with kinematic/dynamic orbit determination is realised by using MFGsoft. The orbit is determined by a geometric way and fitted with a dynamic model. Generally speaking, the orbit solutions are smoother than that of the kinematic ones. The data processing is simpler (without the complicated dynamic force model and integration). The results are not compared externally.

8.2.8 Global Network Monitoring with Dynamic Orbit Determination

For the global network monitoring problem with dynamic orbit determination, the function is realised in MFGsoft. The variance equation are solved using a differential method outlined in Section 7.5. The solution shown the problems of the adjustment model of the solar radiation. A study is given in Section 7.6 and the model is implemented in the MFGsoft and shown good performance. The solution is not external compared.

The software is able to run in a sequential way, an epoch-block wise way and a total post processing way. The epoch-block wise solution is needed for filtering.

8.2.9 Time Consuming

The global network monitoring problem with fixed orbit can be solve in three ways. The time consuming is listed here. For an epoch-block solution, pro hour one solution, it took 4 -- 5 minutes. For data of one day it took 1.5 -- 2 hours, that is ca. 2 seconds for pro epoch data processing. For an sequential solution, pro 2 hour one solution, it took 10 – 15 minutes. For sata of one day it took 2 – 3 hours, that is ca. 3 seconds pro epoch data processing. For a total solution pro day one solution, it took 4 – 5 hours, that is ca. 6 seconds pro epoch data processing. The time consuming results have only relative sense of comparison. It depends on the computer and the user number of the computer etc. The data is recorded before introducing the data condition and implementing the independent parameterisation algorithm. Checking for the data condition and parameterisation are time consuming very intensive.

8.3 External Comparisons

External comparisons are most interested topic and important task for a new software. The basis for a meaningful comparison is that the adjustment and filtering principles used shall be equivalent, the data used shall be nearly the same, the parameterisation methods shall be equivalent, and if the a priori information is used then they shall be similarly used.

The a priori information such as coordinate precision, may be obtained by analysis of a long term stable solutions obtained without a priori information. However as known, most un-differential GPS scientific software are dealing with a singular equation because of the parameterisation problem, therefore without a priori information the software are not able to produce stable solutions in practice. So the chain between solutions and the used a priori information is not closed. Before the software can stably solve the problem without a priori information, one may generally do not able to have the a priori information from the solutions. In turn, without a priori information the software are not able to solve the problem stably to obtain homogenous solutions. In this aspect, MFGsoft is the first software which is able to provide the stable solutions without a priori information and inturn is able to get the a priori information through a long term data processing.

Because of different parameterisation methods, the parameters have partly also different physical meanings. The clock bias is the bias of the clock related to the reference one and includes the related ambiguity bias. For satellite clock, the ambiguity bias is depended on the selection of the related reference station of every satellite, and for receiver clock, the ambiguity bias is depended on the selection of the reference satellite of every station. All parameters of the reference station are considered as error-free or correction-free in MFGsoft due to the nature of the distance measurement of the GPS.

Due to above reasons a successful external comparison is not possible up to now. The numerical results confirm above discussion. Indeed, arbitrary results can be produced by using different empirical information. Therefore the a priori information must be a reasonable ones.

9. Summary

A multi-functional GPS software based on the equivalent data processing algorithm is developed. The software, equipped with an algorithm of independent parameterisation, may provide a stable solution and homogenous results without any a priori information. The functional abilities of the software reached from the local, to the regional and global network applications and is able to process the data with fixed orbit, orbit correction, kinematic/dynamic combined and dynamic orbit determination in real time and post processing modules. The precision of the results, compared with the common reported ones, are still not satisfactory. The software has to run with much more numerical data to learn the intelligence and to modify its quality of functionality.

This software surely can be a good basis for an excellent GPS software. And surely is a good simulation software for the Galileo system study to simulate the data and then to solve the navigation and positioning problem of the Galileo applications. And surely will be a good basis for a GPS and Galileo combined software.

10. Acknowledgements

Gratefully acknowledged are the support from Prof. Dr. Dr. Ch. Reigber. Without his encouragement the development work may never reach this status.

Scientist Abbas Khan of KMS Denmark is thanked for many discussions concerning the ocean loading computation and for his kindly providing us the ocean loading computing routines.

Prof. Dr. JianCheng Li and Dr. ZhengTao Wang of University Wuhan are thanked for their co-work as scientists in GFZ from 2002 to 2003. They are mainly responsible for the research and software development of the force models of dynamic orbit determination and the numerical integration tools as well as the integration and extensive numerical tests of this software. Many valuable and intensive discussions have been held within the team. Specially worth mentioning is that through their work a mistake in the theoretical description of the variation equation is covered. They also pointed out that there are many initial values cannot be computed by solving the variation equation using integration method as traditionally did. This leads to the new development of the algorithm of differential solution of the variation equation. They reviewed also parts of the textbook “GPS – Theory, Algorithms and Applications”. Some of the formulae of the integration methods are also corrected and checked through numerical tests. They contributed greatly for this research and development with their excellent work.

The theoretical study of the software development is carried out in 2001 and at least leads to a detailed description of the dynamic orbit determination theory by using GPS. After one and a half years work, the program development is finished at the middle of 2003, however, due to the parameterisation problem, which has never been carefully studied before, the software runs instable and the results are inhomogeneous. Under the extreme pressure of failing to be able to produce results with the software, the time which I got to research and finally leads to an understanding and solving of the problem at the middle of 2004 must be acknowledged. Without understanding and solving the parameterisation problem, this manual will be never born.

This study was carried out under the grant of the German Federal Ministry of Education and Research (BMBF) No. 50EP9578.

11. References

Abramowitz M, Stegun IA (1965) ***Handbook of mathematical functions***. Dover Publications, Inc., New York

Andersen OB (1994) ***M,2, and S,2, ocean tide models for the North Atlantic Ocean and adjacent seas from ERS-1 altimetry, Space at the service of our environment***. In: Proceedings of the second ERS-1 symposium, Hamburg, 11–14 October 1993, Vol. 2., January 1994, Noordwijk, pp 789–794

Balmino G, Schrama E, Sneeuw N (1996) ***Compatibility of first-order circular orbit perturbations theories: Consequences for cross-track inclination functions***. J Geodesy 70,9:554–561

Beutler G (1996) ***GPS satellite orbits***. In Kleusberg A, Teunissen PJG (eds) GPS for geodesy. Springer-Verlag, Berlin

Beutler G (1996) ***The GPS as a tool in global geodynamics***. In: Kleusberg A, Teunissen PJG (eds) GPS for geodesy. Springer-Verlag, Berlin

Brown R.G. and P.Y.C. Hwang (1992) ***Introduction of Random Signals and Applied Kalman Filter***, Second Edition, John Wiley & Sons, Inc., New York

Bauer M. (1994) ***Vermessung und Ortung mit Satelliten***, Wichmann Verlag, Karslruhe, in German.

Cannon M.E., Lachapelle G., Szarmes M., Herbert J., Keith J. and Jokerst S. (1997) ***DGPS Kinematic Carrier Phase Signal Simulation Analysis for Precise Velocity and Position Determination***, Proceedings of ION NTM 97, Santa Monica, CA.

Cui X., Yu Z., Tao B. and Liu, D. (1982) ***Adjustment in Surveying***, Surveying Publishing House, Peking, in Chinese

Davis P, Rabinowitz P (1984) ***Methods of numerical integration***, 2nd Ed. Academic Press, INC

Euler H.J., and Landau H. (1992) ***Fast GPS ambiguity resolution on-the-fly for real-time applications***, Proceedings of 6th Int. Geod. Symp. on satellite Positioning, Columbus, Ohio, 17-20.

Goad C., Remondi B. (1984) ***Initial Relative Positioning Results Using the Global Positioning System***, Bulletin Geodesique, 58, 193-210.

Gotthardt E. (1978) ***Einfuehrung in die Ausgleichungsrechnung***, Herbert Wichmann Verlag Karlsruhe.

Featherstone W, Dentith M, Kirby J (1998) ***Strategies for the accurate determination of orthometric heights from GPS***. Survey rev 34(267):278–296

Han S. and Rizos C. (1995) ***On-The-Fly Ambiguity Resolution for Long Range GPS Kinematic Positioning***, IAG Symposia 115, edited by Beutler, Hein, Melbourne and Seeber.

Han S. and Rizos, C. (1997) ***Comparing GPS Ambiguity Resolution Techniques***, GPS World, Oct. 1997, pp54-61.

Hofmann-Wellenhof B., Lichtenegger H. and Collins, J. (1997) ***GPS Theory and Practice***, Springer-Verlag, Wien

Hostetter G. H. (1987) ***Handbook of Digital Signal Processing***, Engineering Applications, Academic Press, INC.

King R.W., Masters E.G. Rizos C., Stolz, A. and Collins J. (1987) ***Surveying with Global Positioning System*** **FERD**, Duemmler Verlag, Bonn.

Leick A. (1995) ***GPS Satellite Surveying***, New York: John Wiley & Sons.

Mohamed AH, Schwarz KP (1999) ***Adaptive Kalman filtering for INS/GPS***. J Geodesy 73:193–203

Parkinson B.W., Spilker J.J. (Eds) (1996) ***Global Positioning System: Theory and Applications***, Volume I, II, Progress in Astronautics and Aeronautics, Volume 163

Reigber C, Koenig R (1995) ***On the accuracy of IGS coordinate solution***. In: Proceedings of the First Turkish International Symposium on Deformation “Istanbul-94”, Istanbul, Sept. 5–9, 1995, pp 445–453

Reigber C, Schwintzer P, Luehr H (1996) ***CHAMP – a challenging mini-satellite payload for geoscientific research and application***. Erste Geodaetische Woche, Stuttgart, 7.-12. Oktober 1996, 4 p)

Remondi B. (1984) ***Using the Global Positioning System (GPS) phase observable for relative geodesy: modelling, processing, and results***, University of Texas at Austin, Center for Space Research.

Rizos C, Han S, Chen HY (2000) ***Regional-scale multiple reference stations for carrier phase-based GPS positioning: A correction generation algorithm***. Earth Planets Space 52(10):795–800

Rothacher M, Mervart L (1996) ***Bernese GPS Software Version 4.0***. Astronomical Institute of University of Bern

Rothacher M, Schaer S (1995) ***GPS-Auswertetechniken***. Schriftenreihe des Deutschen Vereins für Vermessungswesen, Bd. 18, pp 107–121

Schafrin B. and Grafarend E. (1986) ***Generating classes of equivalent linear models by nuisance parameter elimination, Applications to GPS observations***, Manuscripta Geodetica (11:262-271)

Schaffrin B (1991) ***Generating robustified Kalman filters for the integration of GPS and INS***. Technical Report, No. 15, Institute of Geodesy, University of Stuttgart

Schaffrin B (1995) ***On some alternative to Kalman filtering***. In: Sanso F (ed) Geodetic theory today. Springer-Verlag, Berlin, pp 235–245

Schwintzer P, Kang Z, Reigber C (1995) ***GPS satellite-to-satellite tracking for TOPEX/Poseidon precise orbit determination and gravity field model improvement***. J Geodyn 20(2):155–166

Seeber G. (1993) ***Satellitengeodäsie***, Walter de Gruyter, in German

Sjoeberg L.E. (1998) ***On the estimation of GPS phase ambiguities by triple frequency phase and code data***, ZfV, 123(1998)5, pp. 162-163, Stuttgart, 1998.

Sjoeberg L.E. (1999) ***Unbiased vs biased estimation of GPS phase ambiguities from dual-frequency code and phase observables***, Journal of Geodesy (1999) 73: 118-124

Strang G. and Borre K. (1997) ***Linear Algebra, Geodesy, and GPS***, Wellesley-Cambridge Press

Teunissen P.J.G. (1995) ***The least-squares ambiguity decorrelation adjustment: a method for fast GPS integer ambiguity estimation***, Journal of Geodesy, 70, 65-82.

Verhagen S. (2004) ***Integer ambiguity validation: an open problem?*** GPS Solutions (2004) 8:36-43

Wang G., Chen Z., Chen, W. and Xu G. (1988) ***The Principle of the GPS Precise Positioning System***, Surveying Publishing House, Peking, in Chinese

Wang LX, Fang ZD, Zhang MY, Lin GB, Gu LK, Zhong TD, Yang XA, She DP, Luo ZH, Xiao BQ, Chai H, Lin DX (1977) ***Mathematic handbook***. Educational Press, Peking, ISBN 13012-0165

Wells D., Lindlohr W., Schaffrin B., Grafarend E. (1987) ***GPS Design: Undifferenced Carrier Beat Phase Observations and the Fundamental Differencing Theorem***, University of New Brunswick

Xu G., Qian Z. (1986) ***The Application of Block Elimination Adjustment Method for Processing of the VLBI Data***, Crustal Deformation and Earthquake, Vol.6, No.4, in Chinese

Xu G., Hehl K., Angermann D. (1994) ***GPS Software Development for Use in Aerogravimetry: Strategy, Realization, and First Results***, Proceedings of ION GPS-94, p1637-1642

Xu G., Bastos L., Timmen L. (1997) ***GPS Kinematic Positioning in AGMASCO Campaigns—Strategic Goals and Numerical Results***, Proceedings of ION GPS-97 Conference in Kansas City (USA), p1173-1183

Xu G., Schwintzer P., Reigber Ch. (1998) ***KSGSoft (Kinematic/Static GPS Software) --- Software User Manual***, Scientific Technical Report 19/1998, GeoForschungsZentrum Potsdam

Xu G. (2002) ***GPS data processing with equivalent observation equations***, GPS Solutions, Vol. 6, No. 1-2, 6:28-33

Xu G. (2002a) ***A general criterion of integer ambiguity search***, Journal of GPS, Vol.1 No.2: 122-131

Xu G. (2003a) ***A diagonalisation algorithm and its application in ambiguity search***, Journal of GPS, Vol.2 No.1:37-43

Xu G. (2003) ***GPS – Theory, Algorithms and Applications***, Springer Verlag, Heidelberg

Yang Y (1997) ***Robust Kalman filter for dynamic systems***. Journal of Zhengzhou Institute of Surveying and Mapping 14:79–84

Yang Y (1999) ***Robust estimation of geodetic datum transformation***. J Geodesy 73:268–274

Yang Y, He H, Xu GC (2001) ***Adaptively robust filtering for kinematic geodetic positioning***. J Geodesy 75:109–116

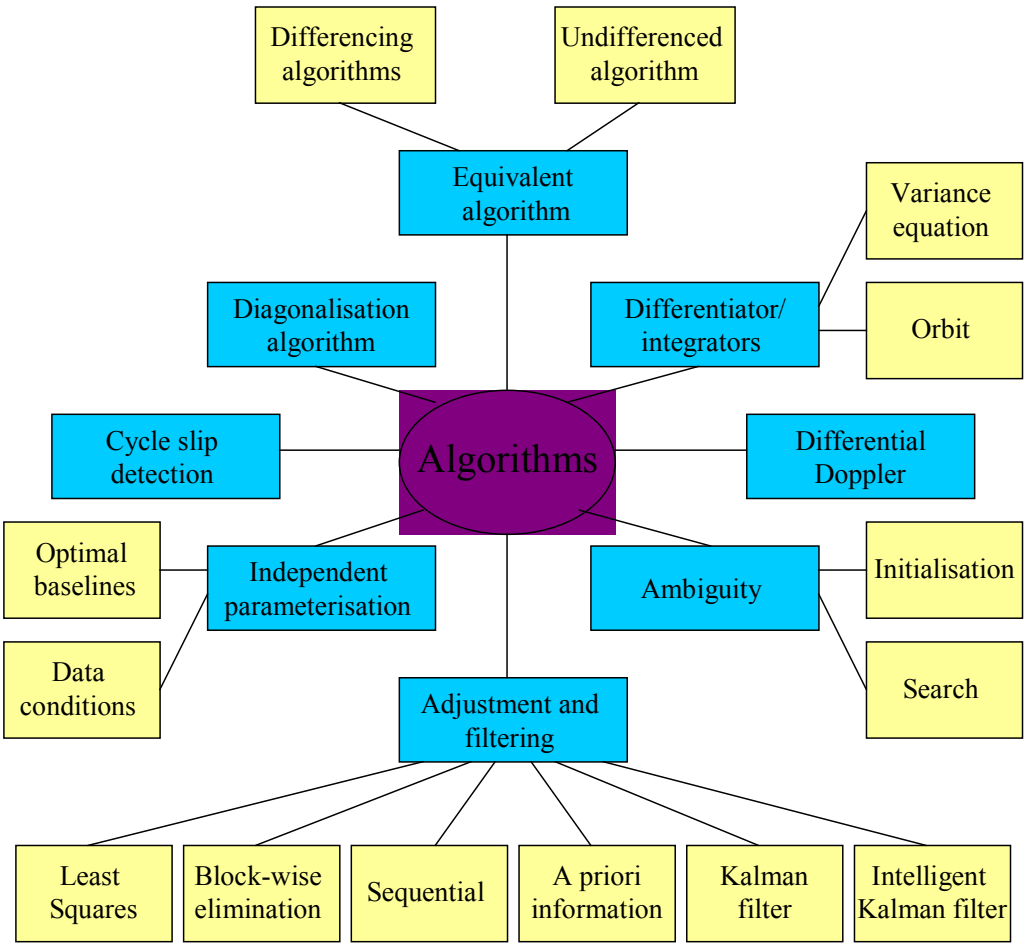
Zhou J. (1985) On the Jie factor, ***Acta Geodaetica et Geophysica***, No.5, in Chinese

Zhou J., Huang J., Yang Y. and Ou, J. (1997) ***Robust least squares method***, Publishing House of Huazhong University of Science and Technology, Wuhan, in Chinese

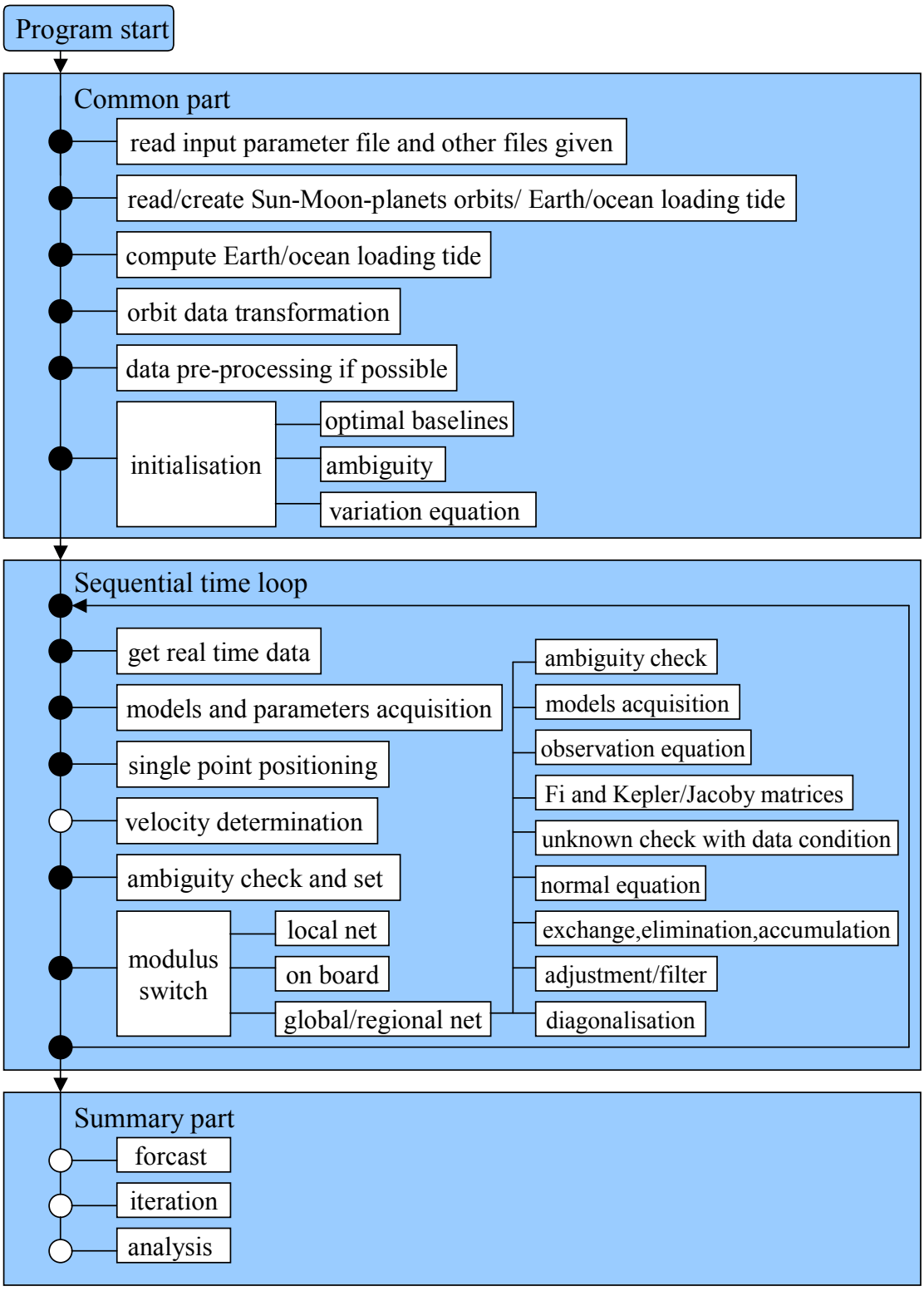
12. Appendixes

12.1 Appendix 1: Diagrams of the Software

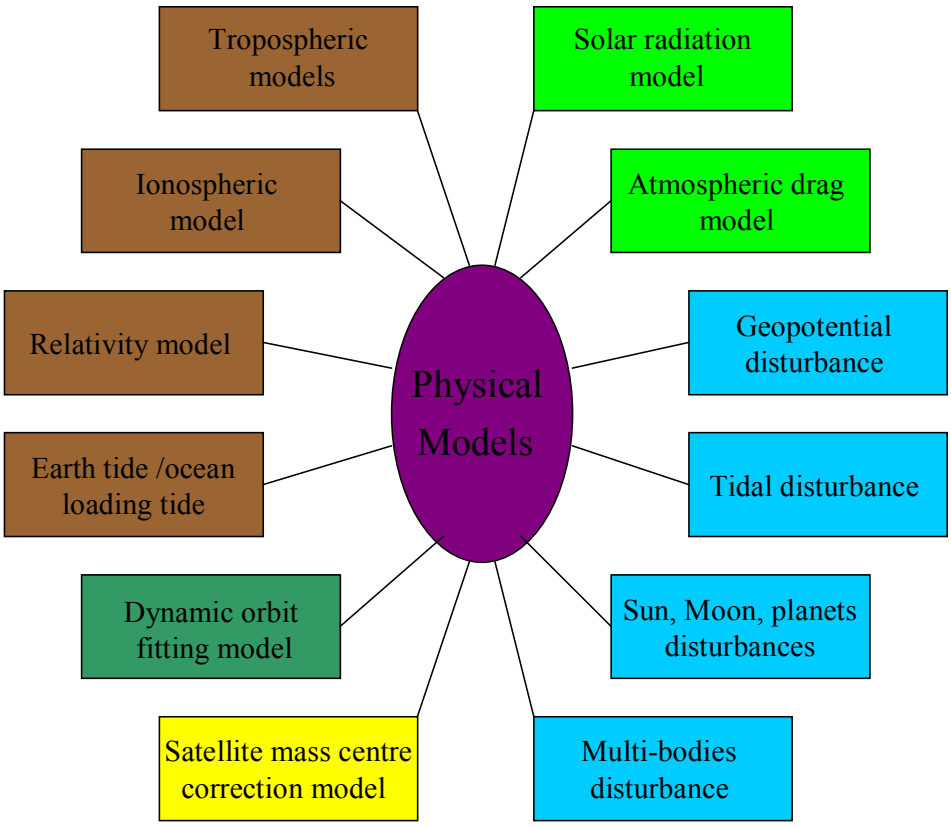
Algorithms diagram



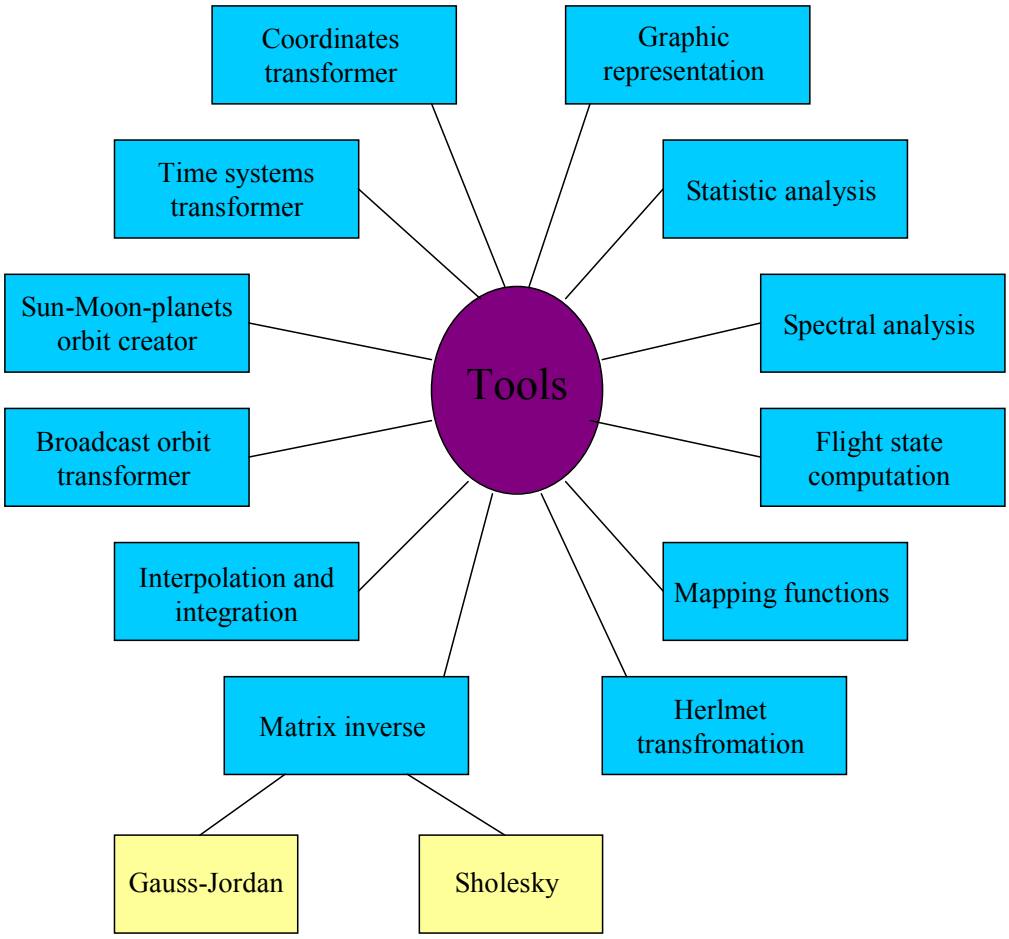
Program diagram



Models diagram



Tools diagram



12.2 Appendix 2: List of Functions of the Software

Complete functions used in this software are listed bellow:

models.c --- software package of models

- double rela_tivity() --- relativity models
- double ion_mapping() --- ionospheric mapping functions
- double tro_mapping() --- tropospheric mapping functions
- double ion_model() --- broadcast inospheric model
- double tropospheremodel() --- tropospheric models and their combination
- double niellis() --- Niels tropospheric model
- double NHMF2() --- mapping function in Niels model
- double NWMF2() --- mapping function in Niels model
- double yiono() --- Viono tropospheric model
- double davis() --- Davis tropospheric model
- double saasta() --- Saastamonie tropospheric model
- double hopfield() --- Hopfield tropospheric model
- double tro_saasta() --- modified Saastamonie tropospheric model
- double tro_hopfield() --- modified Hopfield tropospheric model

tools.c --- software package of tools

- double distance() --- distance computing function
- void helmert_t() --- Helmert transformation
- double zenithAl() --- zenith distance and azimuth computing function
- double atan2() --- arctangent function
- void xyzrphila() --- coordinates transformation between x,y,z and phi,lambda,h
- void rotationR() --- rotation matrix between global and local coordinate systems
- void rotation() --- rotational function
- double zenithA() --- zenith distance and azimuth computing function
- void TRANF() --- coordinates transformation between x,y,z and geodetic phi,lambda,h
- double lagrangef() --- float Lagrange interpolating function
- double lagrange() --- float Lagrange interpolating function with integer step
- void lagrange_cha() --- Lagrange function for Fortran call

- double line2_integrate() --- numerical integration with 2nd order linear fitting model
- double line2_interpo() --- numerical interpolation with 2nd order linear model
- double line_interpo() --- linear interpolating function
- void root2polynom() --- solution of the 2nd order polynomial
- void root3polynom() --- solution of the 3rd order polynomial
- void root4polynom() --- solution of the 4th order polynomial

cood_time.c --- software package of time and coordinate systems

- void t_hms() --- transform function of time from float to hour, min, second
- void rotat() --- rotate a vector
- void rotatm() --- rotate a matrix
- void Rotationm() --- the pole motion rotation matrix
- void Rotations() --- the Earth rotation matrix
- void Rotationn() --- the nutation matrix
- void Rotationn1() --- the nutation matrix for Fortran call
- void Rotationp() --- the precession rotation matrix
- int RotationT() – rotation matrix test function
- double MJD() --- compute JD from year, moth and day
- double JD() --- compute Julian date
- double JD2000() --- compute the Julian date from JD2000.0
- double JD_GPST() --- JD and GPST transformation
- double TDT_GPST() --- TDT and GPST transformation
- void tdt_gpst1() --- TDT and GPST transformation called by Fortran
- double UTC_GPST() --- UTC and GPST transformer
- double Hutc() --- compute UTC from GPST
- double Hut1() --- compute UT1 from GPST
- void YMDHNW_JD() --- compute year, month, day, hour, day_of_week, GPS_week, day_of_year from JD
- void GPSwdow() --- from year, month, day, hour compute GPS_day_of_week, week, day_of_year
- int DayofYear() --- compute day_of_year from year, month, day, hour
- int ntable() --- leap seconds table

help.c --- help software package

- void DdE_e() --- change DdE to e for read data
- void G2blank() --- chang G to blank

- int strstrx() --- find a string in a string
- void blank_2() --- change special characters to blank
- void blank_68() --- change special characters to blank
- int Fix() --- rounding function
- double dfix1() --- double rounding function
- long lfix() --- long rounding function
- int ifix() --- int rounding function
- void filenam() --- attach a number to file name
- void check_balance() --- check balance in c code
- void rem_comment() --- remove the comment lines
- void count() --- count for info in c code
- void search_text() --- search for text in c files
- int index() --- find index of a string in a string
- long count_line() --- count for lines in c code
- int cp() --- copy from file to file
- int lower_f() --- change file name to lower case
- int upper_f() --- change file name to upper case
- void lower_s() --- change string to lower case
- void upper_s() --- change string to upper case
- int getline() --- read a line from a file
- char lower_c() --- change a character to lower case
- char upper_c() --- change a character to upper case
- void adds1tos() --- attach 01 to string
- int strc() --- find a character in a string
- void r_grid_kms() --- read kms format grid data
- void w_grid_kms() --- write kms format grid data
- void w_grid_srf() --- write surfer format grid data
- void D_e() --- change D to e in a data line

chol_12.c --- software package of Cholesky decomposition

- void cp_alow() --- copy a lower case matrix
- void cp_av() --- copy a vector
- void ini_av() --- initial a vector
- void checkE() --- identity matrix check
- void totalinv() --- total inverse

- void checktotal() --- check total inverse matrix
- void checkinvdecompose() --- check inverse decomposition
- void checkdecompose() --- check decomposition
- void invdecompose() --- check decompose
- void decompose() --- decompose
- void pm_low() --- write lower case matrix
- void pm_2d() --- write 2d matrix
- void pv_1drow() --- write a float vector
- void pv_1drowlong() --- write a long vector
- void pv_1drowint() --- write a integer vector
- void pv_1d_v2d() --- write a float vector
- void pv_1d_v2long() --- write a long vector
- void pv_1d_v2int() --- write a integer vector
- void p2v_v2d() --- write two float vectors
- void p3v_() --- write three float vectors
- void p2v_v2long() --- write two long vectors
- void p2v_v2int() --- write two int vectors

adjust.c --- software package of adjustment

- double ls() --- least squares algorithm
- int chol_inv() --- cholesky inverse function
- int gauss_jordan() --- gauss-jordan inverse function
- void pm_low_cl_v2() --- write a matrix in lower case with info
- void rm_low_cl_v2() --- read a matrix in lower case with info
- void pm_2d_cl_v2() --- write 2d matrix with info
- void rm_2d_cl_v2() --- read 2d matrix with info
- void p2v_v2() --- write 2 vectors with info
- void r2v_v2() --- read two vectors with info
- void pstr1v_v2() --- print a variable with info
- double rstr1v_v2() --- read a variable with info
- void rstr1v_v1() --- read a variable with a string
- void pv_1d_v2() --- write a vector
- void rv_1d_v2() --- read a vector

tidkep.c --- software package of tidal computation etc

- void SMP_eph_lday() --- compute one day Sun Moon planets eph
- void SMP_eph() --- compute a epoch Sun Moon planets eph
- void planet_eph0() --- compute planets eph
- double omegapf() --- omega and f
- void SM_eph() --- Sun Moon eph creator
- void SM_eph0() --- Sun Moon eph creator0
- void kepler_el() --- kepler elements and xyz transformer
- void kepler_el_1() --- kepler elements transformer for Fortran call
- double Kepler_equation() --- kepler equation
- void tide_t() --- interpolate the tide for t
- void tide_lday() --- compute 1 day tidal effects
- void tide() --- tide creator
- void interpo_rotat() --- interpolation and rotation
- int SL_eph() --- read Sun Moon eph
- void Rotation_M() --- compute 4 rotational matrices

r_input.c --- software package for read files and parameters

- void r_oload() --- read ocean loading parameters
- void r_poten() --- read potential model parameters
- void pot_str() --- special operation on a string
- void r_selected_sta() --- read selected station file
- void check_sta_dat() --- check station data file
- void r_igssite_r_dat() --- read igs station info
- void get_site_eccentricity() --- read station eccentricity file
- void get_antenna_phasecenter() --- read antenna phase centre data
- void get_site_receiver() --- read receiver type file
- void get_site_dome_info() --- read station dome file
- void r_station_dat() --- read station info
- void get_sta_idt_code() --- read station id code
- void get_plate_id() --- read plate info of stations
- void blank_4id() --- special operation on a string
- void r_satdat() --- read satellite related info
- void r_polar() --- read polar motion file
- void r_input() --- read input parameter file

- int r_str() --- string operation
- void time_debug() --- time debug function
- void wr_condat() --- read condition data
- int indexx() --- search for a character
- void DOP() --- DOP compute function
- void w_info() --- write info
- int overread() --- over read function
- double twoi2f() --- combine two integer to a float

main.c --- software package used in main program

- void sm_ef_t() --- compute the Sum, Moon coordinates in ECEF system
- void set_satid_index() --- set satellite identifier vector
- void rotate_sf_ef() --- coordinates transformation between ECSF and ECEF coordinate systems
- void V_rotate_sf_ef() --- velocity transformation between ECSF and ECEF systems
- void rotate_a_ef_sf() --- transformation of 3*3 matrix from ECEF to ECSF system
- void rotatm_r() --- multiplication of transformation matrix on the right
- void rotatm_l() --- multiplication of transformation matrix on the left
- void rotate_sf_ef1() --- void rotate_sf_ef() for Fortran call
- void rotate_matrices_t() --- get the rotation matrices of time
- void rotate_matrices_t1() --- void rotate_matrices_t() for Fortran call
- void smp_sf_t() --- get Sum, Moon, planets data in ECSF of time
- void smp_sf_t1() --- void smp_sf_t() for Fortran call
- void check_Tr1_2() --- tidal correction check according to the station property
- void out_Tr1() --- output tidal corrections
- void sta_xyz() --- prepare station number and coordinates for tidal computation
- void oload_t1() --- get ocean loading tide correction of time
- void polar_xyt_t() --- interpolate polar motion
- void polar_xyt_t1() --- void polar_xyt_t() for Fortran call

oload_tide.c --- software package of ocean tide loading effect (Fortran originator: Khan)

- void oload() --- call Fortran or C routine to compute oload effect
- void oload_sta_() --- Sun version of oload C call Fortran
- void oload_sta() --- Vampire version of oload C call Fortran
- void oload_sta_c() --- subroutine translated from Fortran
- void etaste_c() --- subroutine translated from Fortran (originator: Wenzel)

- void doodson_c() --- subroutine translated from Fortran, set Doodson coefficients
- double dast_a_c() --- subroutine translated from Fortran
- double ast_a_c() --- subroutine translated from Fortran
- void time_seri_c() --- subroutine translated from Fortran
- void etmutc_c() --- subroutine translated from Fortran
- void etjuln_c() --- subroutine translated from Fortran

broad2.c --- software package of broadcast orbit

- void rotate_2orb() --- rotate IGS and broadcast orbit to ECSF system
- void check_2orb() --- statistic analysis of the two orbits
- void broadcast() --- broadcast orbit computation from given parameters
- void borbc() --- broadcast orbit computation from given parameters
- int findisat() --- find out the satellite identifier
- void broadcast_orbit() --- broadcast orbit computation from given parameters
- double Kepler_equation() --- solving Kepler equation
- void r_eph() --- read eph data
- void eph_check() --- eph data check

orbit.c --- software package of orbit

- int refsat() --- decide the reference station
- int getorb_t() --- get orbit data of time
- void getorb_t1() --- int getorb_t() for Fortran call
- int getorb() --- get orbit data due to sat id and time
- void rd_leo_orb() --- read LEO orbit data
- void get_leoorb_t() --- get LEO orbit data of time
- void igs() --- combine IGS data and read
- void igsdat() --- get IGS data
- void r_igs_h() --- read IGS file header part
- void r_igs_d() --- read IGS file data part

pre67.c --- software package of GPS data pre-processing

- void getdata_rt() --- get GPS data of one epoch
- void open_r_header() --- open GPS data files and read the headers
- void r_rinexheader_only() --- read the header part of all files
- void prepro_gps_dat_0() --- pre-processing GPS data

- void prepro() --- pre-processing
- void r_rinexdata() --- read Rinex data
- void r_rinexheader() --- read Rinex file header
- void d_sat() --- delete some data due to sat id
- void read_afile() --- read so-called a file
- void id_obs() --- set observation id
- void id_i12j12() --- integer transformer
- void internalobs_type() --- set internal observation type
- void d_dat() --- delete the data started no on integer epoch

sp_pv.c --- software package of single point positioning

- void singlep_c() --- single point positioning
- void transrot() --- transmission and rotation correction1
- void transrot1() --- void transrot() for Fortran call
- void earthrot() --- earth rotation correction
- double distance2() --- distance function
- void velocity() --- velocity function

ambi.c --- software package of ambiguity

- void separateNXgroup() --- separate N and X groups
- void determinnNarea() --- determine N area
- void getN_new() --- get N new
- void min2m0_() --- minimum of two m0
- void ambifixc() --- ambiguity fixing
- void search() --- ambiguity search

ambi_ion04.c --- software package of ambiguity-ionosphere equation

- void ambi_ion1() --- ambi-iono equation
- void shift_add() --- shift-and-add operation of matrix
- void shift_add_2() --- shift and add 2 operation
- void d2_low() --- 2d and lower case
- void exchangeij_1() --- exchange two row and column
- void exchangeij_21() --- exchange two row and column
- void exchangeij_2() --- exchange two row and column
- void ambitropref() --- ambi, trop, ref decision

- void in_ambi() --- internal function
- void out_anambi() --- output in picture
- void anambi() --- analysis ambiguity
- void search_1() --- search
- void VWS2blank() --- string operation

another.c --- software package of another routines

- void test8() --- test function
- void theory_ambi() --- theoretical ambiguity
- void sat_view() --- output viewed satellite in picture
- void diagonal_Q() --- Q matrix diagonalisation
- void cp_combine() --- code-phase combination
- void exchange_mx() --- exchange unknown order of the normal matrix
- void b_eli_mx() --- elimination of unknowns
- void out_int_vector() --- output integer vector
- void out_d_vector() --- output float vector
- void out_intG_vector() --- output with condition
- double weight_robust() --- robust weight function
- double weight_zdis() --- weight function of zenith distance
- void out_sta_dt() --- output dt of stations
- int j_x_id() --- integer operation
- int j_x_id4() --- integer operation
- void form_norm_eq() --- form normal equation
- void form_norm_eq1() --- for Fortran call
- void accum_norm_eq() --- accumulation of normal equations

dynamic_orb.c --- software package of dynamic orbit

- void s_variation_eq() --- solving the variation equation
- void get_fi_matrix() --- get Fi matrix
- void get_fi_matrix9() --- get Fi matrix
- void a_f_geo2() --- so-called a matrix of forces
- void air_drag() --- atmospheric drag model
- void air_drag1() --- for Fortran call
- void air_drag2() --- for Fortran call
- double air_density() --- air density function

- void solar_radiation() --- solar radiation model
- void solar_radiation1() --- for Fortran call
- void g_matrix_c() --- get so-called g matrix
- void solar_radiation2() --- for Fortran call
- double shadowf() --- shadow function
- void s_m_ratio_1() --- Surface and mass ratio function
- double min() --- minimum function
- void mc_correction_ef() --- GPS satellite mass centre correction
- void masscenter() --- mass centre function
- void b_air_drag1() --- b matrix of air drag for Fortran call
- void a_solar_radiation1() --- a matrix of solar radiation for Fortran call
- void f_nbody1() --- n-body force
- void a_f_nbody1() --- a matrix of n-body force
- double beta_1() --- beta1 function
- double beta_2() --- beta2 function
- int kronecker_delta() --- kronecker-delta function
- int gauss_jordan3() --- gauss-jordan inverse
- void R_weight_f() --- weight function

xyz.c --- software package of additional functions

- void m33m33() --- matricies multiplication
- void drrdotdkep6() --- Jacorby matrix of Kepler elements and xyz_xyzdot
- void get_f_vector() --- get force vector
- void runge_kutta0() --- Runge_Kutta integration initial
- void adams_cowell_beta() --- Adams-Cowell beta function
- void runge_kutta1() --- Runge_Kutta integration1
- void runge_kutta2() --- Runge_Kutta integration2
- void adams_cowell() --- Adams-Cowell integration function
- void theory_baseline() --- theoretical baselines
- void pm_2d_1v() --- output matrix in 2d
- void minimum_tree3() --- minimum tree problem
- void apriori() --- a priori function
- void eli_rfx() --- elimination of reference coordinates
- void out_int4_vector() --- output vector with condition
- void b_eli_refclock() --- eliminate reference clock

- double ls8() --- ls function modified version
- int chol_inv8() --- Cholesky inversion modified version
- int gauss_jordan8() --- Gauss-Jordan inverse modified version
- void dpdxgRef2sf() --- partial differentiations of orbit fitting model

etc. software package

- kalman.c --- Kalman filtering functions
- data_conditions.c --- data conditions

mfgsoft.c --- main program of MFGsoft

- main.c --- main program
- void position1() --- LEO kinematic/dynamic orbit determination
- void position2() --- local network static/kinematic positioning
- void position3() --- regional network monitoring with orbit correction
- void position4() --- global network monitoring with kinematic/dynamic orbit determination
- void position5() --- global network monitoring with dynamic orbit determination