



Matlab Algorithm Availability Simulation Tool (MAAST) Software Developer's Guide

Version 1.1

January 30, 20002

Table of Contents

<i>i. Preface.....</i>	<i>3</i>
<i>ii. About this Manual</i>	<i>4</i>
<i>iii. MAAST Overview.....</i>	<i>4</i>
<i>iv. MAAST files most commonly modified.....</i>	<i>5</i>
<i>1 Graphical User Interface.....</i>	<i>6</i>
<i>1.1 Key GUI File Information</i>	<i>7</i>
<i>2 SVM Directory Files and Data Matrixes</i>	<i>8</i>
<i>2.1 Customizing MAAST Data Files.....</i>	<i>8</i>
2.1.1 MAAST Data File Modification Procedure:.....	8
2.1.2 MAAST Data File Modification Example.....	8
2.1.3 YUMA Data file Selection Procedure	9
<i>2.2 Satellite Data Calculations.....</i>	<i>9</i>
<i>3 Algorithm and Simulation</i>	<i>9</i>
<i>3.1 SVM Simulation</i>	<i>9</i>
<i>3.2 WMS Processing.....</i>	<i>11</i>
<i>3.3 Custom WMS Processing</i>	<i>12</i>
3.3.1 Procedure for Custom WMS algorithms (CNMP, TROPO or UDRE).....	13
3.3.2 Custom WMS Modification Example	13
<i>3.4 User Processing.....</i>	<i>13</i>
<i>3.5 User processing Modifications</i>	<i>14</i>
<i>3.6 Output Processing</i>	<i>15</i>
<i>4 References</i>	<i>16</i>
<i>4.1 Text/Publication References</i>	<i>16</i>
<i>4.2 Website References.....</i>	<i>16</i>
<i>5 List of Figures.....</i>	<i>17</i>
<i>6 List of Tables.....</i>	<i>17</i>
<i>7 List of Appendixes.....</i>	<i>17</i>
Appendix #1: MAAST File Listing	18
Appendix #2: MAAST Data File Formats	19
Appendix #3: Table of MAAST Constants	20
Appendix #4: MAAST Function Map.....	21
Appendix #5: Matrix Mapping.....	23
Appendix #6: Block Diagrams.....	27

i. Preface

The Wide Area Differential GPS (WADGPS) Laboratory of Stanford University has developed the Matlab Algorithm Availability Simulation Tool, MAAST, to provide a tool for availability simulation of the Wide Area Augmentation System. This software is available in the public domain for users wishing to simulate the impact on WAAS availability as a result of proposed changes in the system.

To support the public use of this tool, the MAAST Software Developer's Guide is provided to assist those wishing to use, modify or tailor MAAST for their own specific purposes. The purposes of this guide are provided below:

- Identify the key features of the MAAST program that allows users to customize and modify the code for their own application.
- Provide necessary information to the user to simplify the code and subroutine customizations.
- Provide detailed reference information of files, functions and data matrixes to facilitate end-user software modifications.

Key references are listed in Section 4. Feedback on MAAST is welcome and encouraged. Please direct comments to The Wide Area Differential GPS (WADGPS) Laboratory of Stanford University.

ii. About this Manual

Easy customization of MAAST was a key design philosophy in developing this code. This manual is provided to support the customization of MAAST and is organized to follow the organization of the code itself. The general approach and key ideas for customizing MAAST are summarized below:

1. Ensure a complete set of MAAST files is available before starting. Refer to Appendix #1 for a complete listing of files provided.
2. This manual and MAAST codes are provided for familiarization with the functional flow, basic assumptions and customization procedures of MAAST.
3. Knowledge of the input/output requirements of the desired algorithms can be quickly found by checking the feval lines in the codes. The use of identical names for inputs/outputs will simplify customization. Summary tables are included for major computation algorithms.
4. MAAST uses global variables to initialize variable names as part of a philosophy to minimize variable name changes and simplify variable use within the multiple .m files of MAAST. The global variable appear in all capital letters at the beginning of the MAAST .m files. Global declarations of all *init_xxx.m* files are located within the associated .m file.
5. MAAST utilizes initialization files (*init_xx.m*) as a “#define” approach so MAAST can call different functions and/or data without having to modify the original init.m files.
6. MAAST uses the following system to name .m files:
 - af_ : Algorithm .m files
 - init_ : Initialization #define .m files described in #5 above.
 - calc_ : Calculation .m files
 - gui_ : GUI related .m files

iii. MAAST Overview

MAAST was designed for easy modification and changes to the code for user specific customization. Modified algorithms can be integrated quickly into MAAST by modifying only a few lines of code.

To facilitate flexibility, MAAST is organized into the following functional areas:

1. **Graphical User Interface (GUI):** User specified inputs.
2. **SVM directory files:** Containing WRS, satellite, IGP and other data files.
3. **Algorithm and Simulation:** Primary error algorithm processing
 - Satellite/User computations: Computed at each time interval
 - WMS processing: CNMP,TROPO,UDRE and GIVE computations
 - User processing: User error computations
 - Output processing: Availability, HPL and VPL processing as requested
4. **Output display:** Color displays of availability, HPL and VPL contour plots.

This guide provides all the necessary information to make rapid customization to functional areas one, two and three above. Modifications of the output display are not provided in this document. A top-level block diagram is provided below in Figure #1 showing the functional areas and the top-level algorithm flow for MAAST.

SVM Analysis Block Diagram (Top Level View)

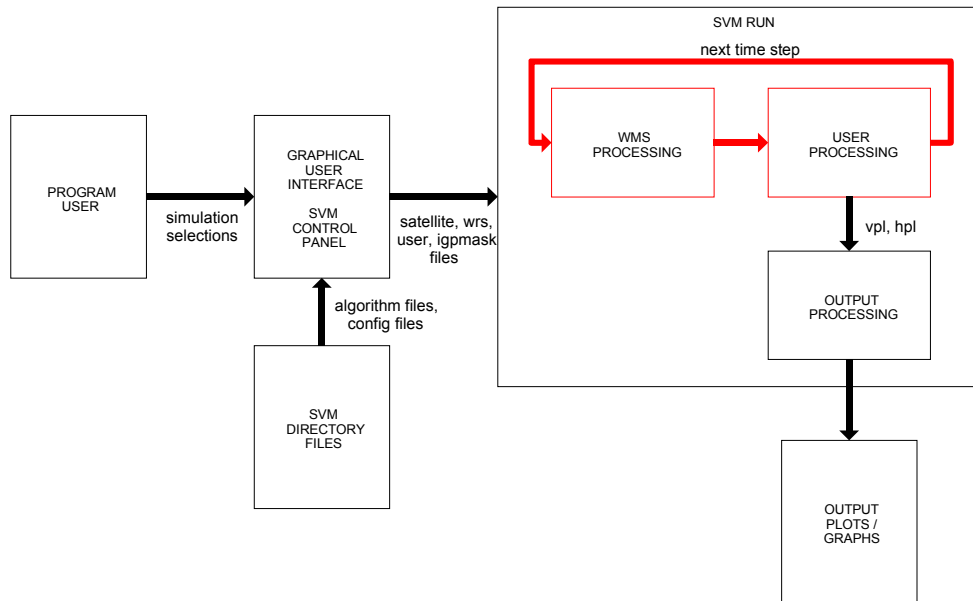


Figure #1: SVM Analysis Block Diagram

iv. MAAST files most commonly modified

The following files are frequently referenced and/or modified during the customization of MAAST:

- *initgui.m* – Primary GUI file where algorithm changes are implemented.
- *guicbfun.m* – GUI file that assigns name strings to selected algorithm functions.
- *svmrn.m* – Main computational algorithm.
- *wmsprocess.m* - WAAS master station processing algorithm.
- *usrprocess.m* – User processing algorithm
- *outprocess.m* – File that formats and displays results of simulation
- *init_col_labels.m* – File that initializes column names for matrixes in MAAST.

1 Graphical User Interface

The MAAST Graphical User Interface (GUI) allows the user to specify algorithms and simulation parameters for analysis. The GUI can be easily customized for user specific modifications. Figure #2 shows the GUI displayed by MAAST.

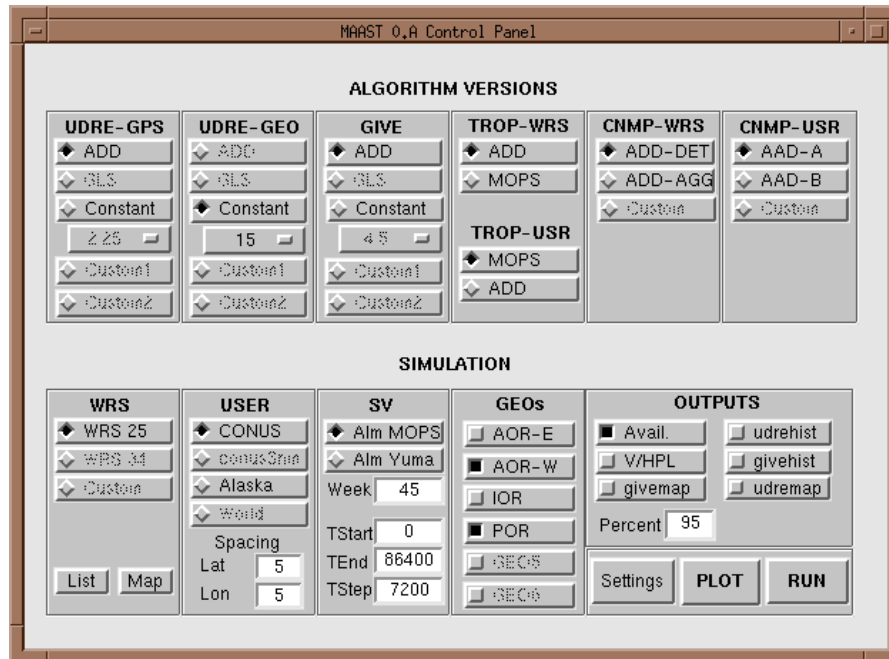


Figure #2: MAAST Control Panel GUI

“Custom” selections buttons are pre-programmed into the GUI. The GUI may be modified for additional buttons if desired. Selection buttons appear “grayed-out” when there is no associated algorithm in MAAST. When a custom algorithm or simulation is integrated into MAAST, the GUI button will reflect the name of the customized subroutine and will no longer be ‘greyed-out’. This feature allows users to add subroutines and selection buttons to build the flexibility of MAAST. Table #1 summarizes the selections currently available in MAAST.

Table #1: MAAST Control Panel Selections

Algorithms	Selections available
UDRE-GPS	CONSTANT option activates a list of indexed UDRE values from which to choose from. This sets all GPS satellite UDREs to the specified constant.
UDRE-GEO	CONSTANT option activates a list of indexed UDRE values from which to choose from. This sets all GEO UDREs to the specified constant.
GIVE	CONSTANT option activates a list of indexed GIVE values from which to choose from. This sets all IGP GIVES to the specified constant.
TROPO-WRS	WAAS MOPS
TROPO-USR	WAAS MOPS
CNMP-WRS	Selects WRS model used to describe the airborne error contribution as a function of elevation and/or time tracked.
CNMP-USR	Selects WRS model used to describe the airborne error contribution as a function of elevation and/or time tracked.

Simulation	
WRS	1. WRS 25 uses the 25 reference station used by the NSTB testbed. 2. LIST / MAP buttons show a list/map of the selected WRS network.
USER	1. CONUS selects users within the continental United States. 2. ALASKA selects users in Alaska. 3. LAT / LON fields specify user grid resolution in degrees.
SV	1. YUMA uses almanac file corresponding to the specified week. 2. TSTART, TEND, and TSTEP specify the time steps for the simulation in seconds.
GEOs	1. AOR-E 2. AOR-W 3. IOR 4. POR
Outputs	Availability contour VPL contour HPL contour UDRE map GIVE map UDRE / σ_{fit} histogram GIVE / UIVE histogram
Other	1. PERCENT field specifies the availability percentage on which the VPL and HPL plots are based. 2. RUN button runs simulation and plots selected outputs. 3. PLOT button plots selected outputs for latest simulation run. 4. SETTINGS button opens new window for simulation settings. (Not available as of this writing.)

1.1 Key GUI File Information

Table #2 provides a summary description of key GUI files used in MAAST.

Table #2: GUI Summary Table

File	Description
init_gui.m	Main routine, builds GUI • Input: Global function/routines • Output: Global function/routines
gui_mexclude.m	Function gui_mexclude(obj_list,sel_obj) • Obj_list: list of handles of objects not selected (to be mutually excluded). • Sel_obj: list of handles selected in the GUI
gui_readnum.m	Function num=gui_readnum(hndl,llim,ulim,errmsg) • Reads the value of a numerical text box.
gui_readselect.m	Function iselect = gui_readselect(objlist) • Returns the index of the selected option/s (Value field = 1) in a list of object handles.
guicbfun.m	Function guicbfun(hndl) • Assigns the working variable name to the algorithm selected by the GUI.

2 SVM Directory Files and Data Matrixes

MAAST draws upon several data files for data used in computations. Table #3 lists the eight data files provided with MAAST. Appendix #2 provides detailed file information and formats for reference.

Table #3: MAAST data files

Data File	Description
usrconus.dat	Longitude and Latitude of boundary of user region in continental U.S.
usralaska.dat	Longitude and Latitude of boundary of user region in Alaska.
wrs25.dat	Longitude and Latitude of location of WAAS reference stations
igpaor.dat igppor.dat igpjoint.dat	Mask grid of active ionospheric grid points in Longitude and Latitude for Atlantic (aor) Pacific (por) and joint (both).
geo.dat	Data for geostationary satellites
almmops.dat	Weekly almanac data

MAAST takes data from these files and generates matrixes used for satellite, user and reference station computations. Critical MAAST matrixes are mapped and provided in Appendix #4 contains a function map that provides information on inputs and outputs of the major functions of MAAST. Appendix #5 for reference on matrix organization, size, data units and relations within MAAST.

2.1 Customizing MAAST Data Files

The following procedure is provided to customize data files used by MAAST.

2.1.1 MAAST Data File Modification Procedure:

1. Modify existing data file with new information.
2. Save the modified data file with a custom name or for minor changes save over the old file.
3. Make the following modification to the `init_gui.m` file:
 - Modify the `GUI_XXX_DAT` line of code with the name of the new data file.
 - Modify the `GUI_XXX_MENU` line of code with to the desired name for the button used in the GUI for the routine
4. Save changes to the `init_gui.m` file and run MAAST.

2.1.2 MAAST Data File Modification Example

A user has generated a custom WRS data file named “`wrsspecial.data`” To integrate this into MAAST, the user would open the `init_gui.m` file and go to the following lines:

```
GUI_WRS_DAT = {'wrs25.dat','wrs34.dat','wrscustom.dat'};  
GUI_WRS_MENU = {'WRS 25','WRS 34','Custom'};
```

Substitute '`wrscustom.dat`' with '`wrsspecial.dat`' and 'Custom' with 'WRS Special' to change the code as follows:

```
GUI_WRS_DAT = {'wrs25.dat','wrs34.dat','wrsspecial.dat'};  
GUI_WRS_MENU = {'WRS 25','WRS 34','WRS Special'};
```

2.1.3 YUMA Data file Selection Procedure

The MAAST user has the option of selecting YUMA almanac data for analysis. This is accomplished as follows: modify the data file prior to maast stip off blank lines

1. Obtain recent YUMA almanac data from the following web site <http://www.navcen.uscg.gov/ftp/GPS/almanacs/yuma/> and save the file as 'almyuma_xxx_dat' in the same directory as MAAST.
2. When running MAAST, select 'Alm Yuma' on the GUI to utilize YUMA data for analysis.

2.2 Satellite Data Calculations

MAAST takes raw almanac data from the almmops.dat file and stores it into matrixes *alm_param* and *satdata* for use in MAAST as follows:

1. Recent almanac data is pulled from a file and stored into *alm_param* matrix(n x 12)
2. *init_satdata.m* initializes the satdata matrix (n x 27)
3. *init_satdata.m* calls *alm2satposve.ml* function to take raw almanac data at time *t* and calculates the ECEF XYZ position and velocity for each satellite and stores the data into the satdata matrix.

At each increment of time, *t*, *init_satdata.m* calls *alm2satposvel.m* and recalculates satellite position for the next increment of time *t*. This data is written over previous data in the satdata matrix. Please refer to Appendix #5 for data mapping of *alm_param* and *satdata* matrixes.

3 Algorithm and Simulation

A description of the major simulation and analysis computations is provided below. Primary MAAST computations are performed by *svrun.m*. Custom algorithms and simulations can be integrated into MAAST easily. When customizing algorithms it is essential to match the input/output information and data formats.

3.1 SVM Simulation

The primary computational algorithm for MAAST is the SVM simulation routine located in the *svrun.m* (see Figure #3 below). SVM's performs the following key functions:

1. Initializes *satdata*, *wrsdata* and *igpdata* matrixes.
2. Calculates LOS rise times for CNMP calculations
3. Establishes *tstart* at *tstart - cnmp-tl3* (below this CNMP is at floor value)
4. For the time period and time steps specified by the GUI, SMV calls the following algorithms for calculations
 - *wmsprocess.m* for UDRE and GIVE calculations
 - *usrprocess.m* vpl and hpl calculations
 - *outputprocess.m* for output displays.

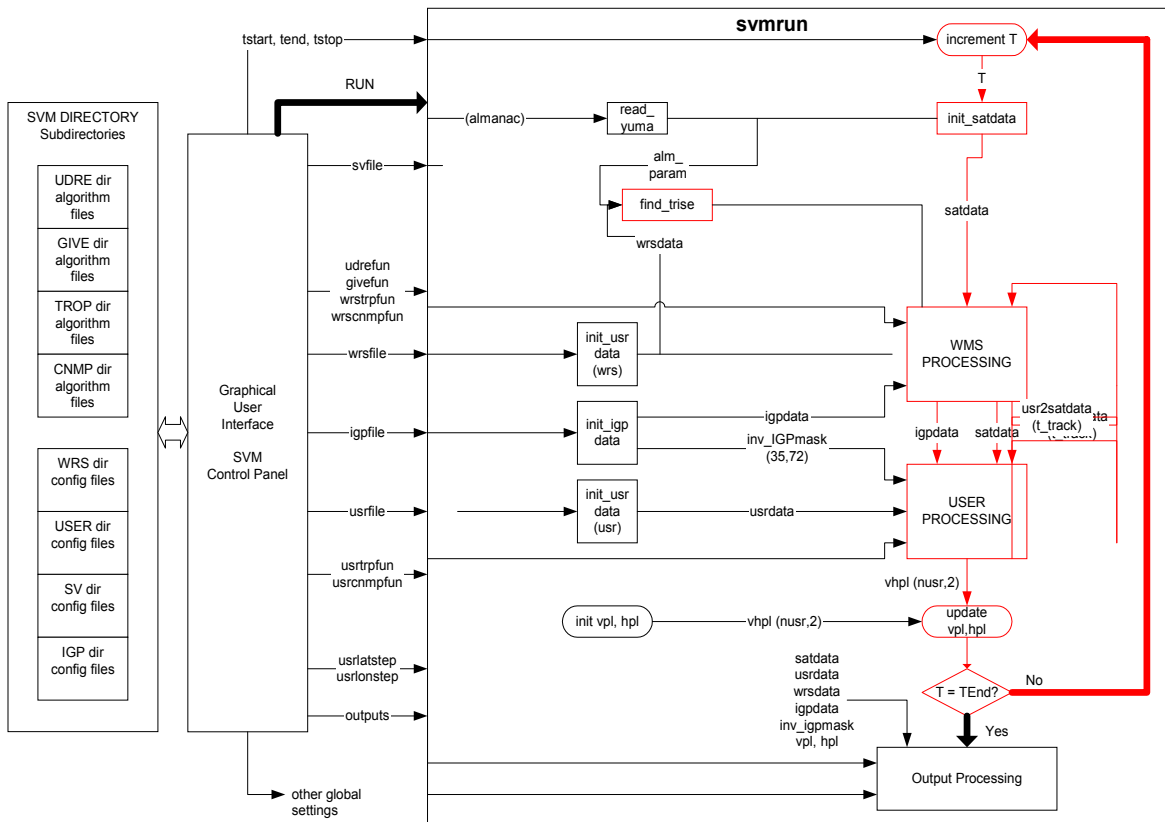


Figure #3: SVM Processing Flow

Table #4: SVMRUN Summary Table

Svmrun.m	
Function Line	
Function svmrun(gpsudrefun, geoudrefun, givefun, usrtrpfun, usrcnmpfun, wrstrpfun, wrsgpscnmpfun, wrsggeocnmpfun, wrsfile, usrfile, igpfile, svfile, geodata, tstart, tend, tstep, usrlatstep, usrlonstep, outputs, percent);	
Inputs	Outputs
gpsudrefun- udre calculation for GPS satellites geoudrefun – udre calculation for GEO satellites givefun – give calculation usrtrpfun - tropo delay calculation for user usrcnmpfun – cnmp delay calculation for user wrstrpfun - tropo delay calculation for wrs wrsgpscnmpfun – cnmp delay calculation for wrs wrsfile – wrs position data usrfile – user position boundary polygon igpfile- IGP mask points svfile – Yuma almanac file if Tstep not zero Static satellite position file if Tstep is zero tstart – start time of simulation (for almanac option) tend-end time of simulation (for almanac option) tstep- time step of simulation (for almanac option), should be 0 for static satellite position option	Array of ON-OFF flags (1 for ON, 0 for OFF) for output options, corresponding to: 1) Availability 2) UDRE map 3) GIVE map 4) UDRE histogram 5) GIVE histogram 6) VPL and HPL

3.2 WMS Processing

When called by *svmrn*, the *wmsprocess* algorithm simulates the computations performed by the WAAS master stations for each time interval using data from the WAAS reference stations. See Figure #4 below. These computations include the following:

1. CNMP: Noise/Multi-path error variance
 - Carrier phase is continuous while the satellite-to-reference station elevation angle exceeds the visibility limit (5°) by the WAAS MOPS
 - Cycle slips do not occur.
 - Satellite rise times are calculated to 1-second accuracy.
2. TROPO: Troposphere error variance
3. UDRE/GIVE calculations

This data is stored in the *wrs2sat_data* matrix that includes the type 28 covariance matrixes.

WMS PROCESSING (UDRE/GIVE)

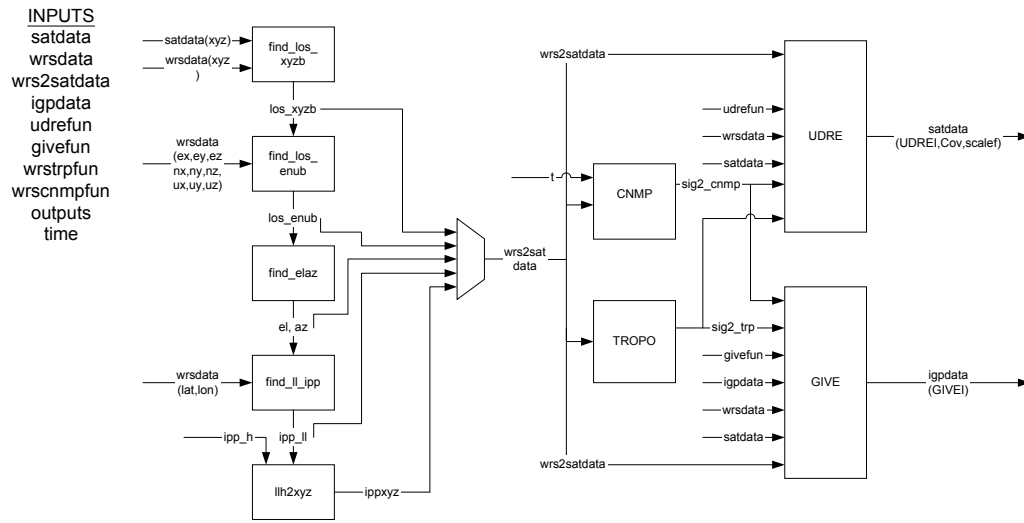


Figure #4: WMS Processing Flow (UDRE/GIVE)

Table #5: WMS Process Summary Table

wmsprocess.m		
Function Line		
<pre>function [satdata,igpdata,wrs2satdata] = wmsprocess(alm_param, satdata, wrsdata, igpdata, wrs2satdata, gpsudrefun, geoudrefun, givefun, wrstrpfun, wrsgpscncmpfun, wrsgocncmpfun, outputs, time, tstart, tstep, trise)</pre>		
Inputs		Outputs
alm_param	Wrstrpfun	satdata igpdata wrs2satdata
satdata	Wrsgpscncmpfun	
wrsdata	Wrsgeocncmpfun	
igpdata	Outputs	
wrs2satdata	Time	
gpsudrefun	Tstart	
geoudrefun	Tstep	
givefun	trise	

3.3 Custom WMS Processing

New WMS algorithms are easily incorporated into MAAST provided they include the same inputs and outputs used by MAAST. This is accomplished by referring to the feval function lines in the code to ensure the custom algorithm will match the same inputs called in MAAST. For convenience, those lines are provided in Table #6 below:

Table #6: Feval Lines in wmsprocess.m

	Input/Output matching for customizing WMS processing
TROPO	<code>wrs2satdata(:,COL_U2S_EL) = feval(wrstrpfun,el)</code>
CNMP	<code>wrs2satdata(gps,COL_U2S_SIG2L1MP) = feval(wrsgpscncmpfun,ttrack(gps),el(gps));</code> <code>wrs2satdata(geo,COL_U2S_SIG2L1MP) = feval(wrsgeocncmpfun,ttrack(geo),el(geo));</code>
UDRE	<code>Satdata(sgps,:) = feval(gpsudrefun, satdata(sgps,:), wrsdata,wrs2satdata(gps,:), 1);</code> <code>Satdata(sgeo,:) = feval(geoudrefun, satdata(sgeo,:), wrsdata,wrs2satdata(geo,:), 1);</code>

When making a custom algorithm ensure the new function is consistent with the inputs and outputs called for in MAAST. For example, if making a new GPS CNMP algorithm, refer to the table above to match the inputs and outputs of the feval line. The new function should use `ttrack(gps)` and `el(gps)` as inputs to the `wrsgpscncmpfun.m` algorithm.

3.3.1 Procedure for Custom WMS algorithms (CNMP, TROPO or UDRE)

1. Compose the custom WMS algorithm function as a .m file in Matlab saved in the same directory as all MAAST files. For consistency, all algorithm files in MAAST begin with 'af_'.
2. Make the following modification to the `init_gui.m` file:
 - Modify the 'af_XXXXcustom1' text of the associated `xxx_XXXX_ALGO` line of code to the name of the custom routine .m file.
 - Modify the 'Custom1' text of the associated `XXXXXXXXX_MENU` line of the code to the desired name for the button used in the GUI for the routine.
3. Save the `init_gui.m` file and run MAAST.

3.3.2 Custom WMS Modification Example

A user has generated a custom algorithm for WRS CNMP calculation named `af_cnmpspecial.m`. To integrate this into MAAST, the user would open the `init_gui.m` file and go to the following lines:

```
GUI_WRSCNMP_ALGO = {'af_cnmpadd','af_cnmpagg','af_wrscnmpcustom'};
GUI_WRSCNMP_MENU = {'ADD-DET','ADD-Agg','Custom'};
```

Substitute 'af_wrscnmpcustom' with 'af_cnmpspecial' and 'Custom' with 'CNMP Special' to change the code as follows:

```
GUI_UDREGPS_ALGO = {'af_udreadd','af_udregls','af_udreconst','af_cnmpspecial','af_udrecustom2'};
GUI_WRSCNMP_MENU = {'ADD-DET','ADD-Agg','CNMP Special'};
```

The button in the GUI is now renamed and linked to the custom function. Selecting this button on the GUI will run the custom function as the UDRE function in the SVM run algorithm.

3.4 User Processing

Following WMS processing by `svmrn`, "User Processing" simulates the WAAS user's computations of confidence bounds on clock/ephemeris and ionospheric corrections at the user site. See Figure #5 below. These confidence bounds are then used to derive the VPL and HPL values. Using the previously generated line-of-site data `usrprocess.m` implements two modules in accordance with WAAS Mops as follows:

1. `udre2flt.m`: Projects satellite UDRE's with MT28 covariance matrixes into fast and long-term correction variances, σ_{flt}^2 for each user line-of-site.
2. `grid2uive.m`: Derives user ionospheric correction variances, σ_{uive}^2 , for each user line-of-site.

USER PROCESSING

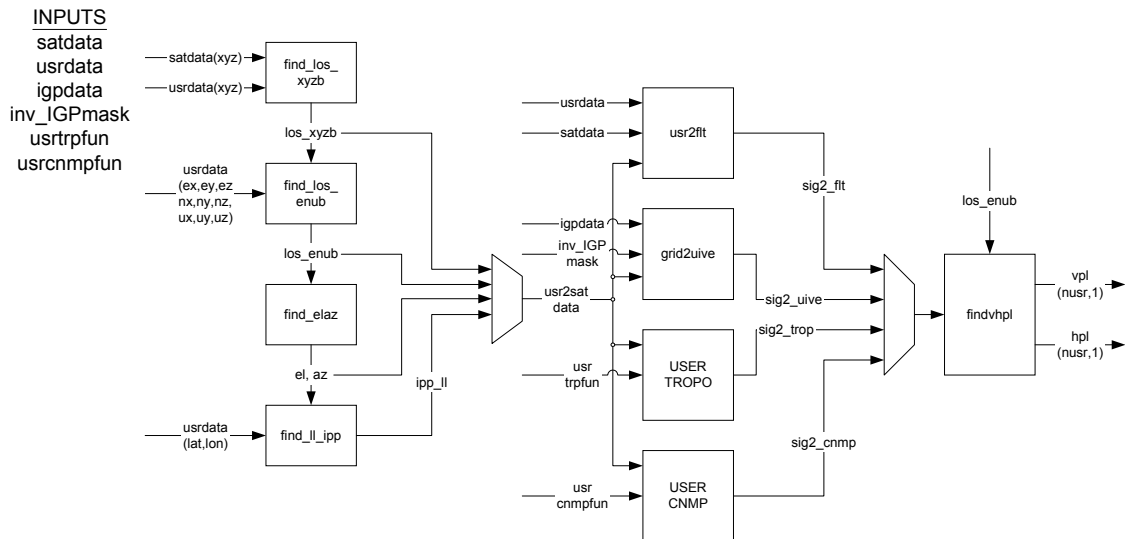


Figure #5: USER Processing Flow

Table #7: User Process Summary Table

usrprocess.m		
Function Line		
Function [vhpl, sig2_fit, sig2_uive, usr2satdata] = usrprocess (satdata, usrdata, igpdata, inv_igp_mask, usr2satdata, usrtrpfun, usrcnmpfun, alm_param, time);		
Inputs		Outputs
satdata	Usrtrpfun	vhpl
usrdata	Usrcnmpfun	sig2_fit
igpdata	alm_param	sig2_uive
inv_igp_mask	time	usr2satdata
usr2satdata		

3.5 User processing Modifications

Modifications to user processing can be accomplished in the same manner as changes to WMS processing above.

3.6 Output Processing

Once user processing completes the updates of VPL and HPL for the time period of analysis, *svmrn.m* calls *outputprocess.m* to prepare and display the data requested by the user in the GUI. This completes the MAAST processing.

Table #8: Output Process Summary Table

outputprocess.m		
Function Line		
Function outputprocess(satdata,usrdata,wrsdata,igpdata,inv_igp_mask, sat_xyz, udrei,givei,usrvpl, usrhpl,latgrid,longgrid,outputs,percent,udre_hist,give_hist,udrei_hist,givei_hist)		
Inputs		Outputs (if selected in GUI)
satdata	Usrhpl	Availability contour
usrdata	Latgrid	VPL contour
wrsdata	Longrid	HPL contour
igpdata	Outputs	UDRE map
inv_igp_mask	Percent	GIVE map
sat_xyz	udre_hist	UDRE / σ_{fit} histogram
udrei	give_hist	GIVE / UIVE histogram
givei	udrei_hist	
usrvpl	givei_hist	

4 References

4.1 Text/Publication References

1. Enge, P., Walter, T., Pullen, S., Kee, C., Chao, Y.-C., Tsai, Y.-J., “Wide Area Augmentation of The Global Positioning System,” *Proceedings of the IEEE*, Volume: 84: 8, August, 1996.
2. Jan, S., Chan W., Walter T., Enge, P., ”Matlab Simulation Toolset for SBAS Availability Analysis”, ION GPS Proceedings, September 2001.
3. Chan. W., “Matlab Algorithm Availability Simulation Tool User’s Guide: Version 0.1, 7 September, 2001.
4. The MathWorks, Inc., “Using MATLAB”, Version 5, 1998.

4.2 Website References

1. Stanford GPS WAAS Laboratory
<http://waas.stanford.edu/>
2. MAAST download site at Stanford University
<http://waas.stanford.edu/~wwu/maast/maast.html>
3. Matlab help desk
<http://www.mathtools.net/MATLAB/index.html>
4. U.S. Coast Guard web site for YUMA formatted ephemeris files
<http://www.navcen.uscg.gov/ftp/GPS/almanacs/yuma/>

5 List of Figures

Figure #1: SVM Analysis Block Diagram.....	5
Figure #2: MAAST Control Panel GUI.....	6
Figure #3: SVM Processing Flow.....	10
Figure #4: WMS Processing Flow (UDRE/GIVE).....	12
Figure #5: USER Processing Flow	14

6 List of Tables

Table #1: MAAST Control Panel Selections.....	6
Table #2: GUI Summary Table.....	7
Table #3: MAAST data files.....	8
Table #4: SVMRUN Summary Table.....	10
Table #5: WMS Process Summary Table.....	12
Table #6: Feval Lines in wmsprocess.m.....	12
Table #7: User Process Summary Table.....	14
Table #8: Output Process Summary Table	15

7 List of Appendixes

Appendix #1: MAAST File Listing.....	18
Appendix #2: MAAST Data File Formats.....	19
Appendix #3: Table of MAAST Constants	20
Appendix #4: MAAST Function Map	21
Appendix #5: Matrix Mapping	23
Appendix #6: Block Diagrams.....	27

Appendix #1: MAAST File Listing

The Following files are found in the Maast.zip file

Name	Type	Comments	Name	Type	Comments
1 af_cnmpaad.m	Matlab .m		41 init_const.m	Matlab .m	MAAST global constants
2 af_geoconst.m	Matlab .m		42 init_graph.m	Matlab .m	
3 af_giveconst.m	Matlab .m		43 init_gui.m	Matlab .m	
4 af_trpmops.m	Matlab .m		44 init_hist.m	Matlab .m	
5 af_udreconst.m	Matlab .m		45 init_igpdata.m	Matlab .m	
6 alm2satposvel.m	Matlab .m		46 init_igps.m	Matlab .m	
7 almmops.dat	Data File		47 init_mops.m	Matlab .m	
8 avail_contour.m	Matlab .m		48 init_satdata.m	Matlab .m	
9 calc_ll_ipp.m	Matlab .m		49 init_usr2satdata.m	Matlab .m	
10 calc_los_enub.m	Matlab .m		50 init_usrdata.m	Matlab .m	
11 check_igpsquare.m	Matlab .m		51 init_wrsdata.m	Matlab .m	
12 checkfor2.m	Matlab .m		52 intriangle.m	Matlab .m	
13 colorb.m	Matlab .m		53 llh2xyz.m	Matlab .m	
14 disccov.m	Matlab .m		54 maast.m	Matlab .m	Main Program
15 find_elaz.m	Matlab .m		55 maastgui.m	Matlab .m	Primary GUI file
16 find_inv_IGPmask.m	Matlab .m		56 maastgui.mat	Matlab .mat	
17 find_ll_ipp.m	Matlab .m		57 mapdata.mat	Matlab .mat	
18 find_los_enub.m	Matlab .m		58 mapudre.m	Matlab .m	
19 find_los_xyzb.m	Matlab .m		59 mynormcdf.m	Matlab .m	
20 find_trise.m	Matlab .m		60 mysetdiff.m	Matlab .m	
21 findcommon.m	Matlab .m		61 obliquity2.m	Matlab .m	
22 findxyz2enu.m	Matlab .m		62 outputprocess.m	Matlab .m	Primary Output processing
23 geo.dat	Data File		63 project2user.m	Matlab .m	
24 give_contour.m	Matlab .m		64 quadfit.m	Matlab .m	
25 give_histogram.m	Matlab .m		65 read_yuma.m	Matlab .m	
26 grid2uive.m	Matlab .m		66 svm_contour.m	Matlab .m	
27 gui_mexclude.m	Matlab .m		67 svm_hist.m	Matlab .m	
28 gui_readnum.m	Matlab .m		68 svm_histogram.m	Matlab .m	
29 gui_readnum.m	Matlab .m		69 svmrn.m	Matlab .m	
30 gui_readselect.m	Matlab .m		70 udre_histogram.m	Matlab .m	
31 guicbfun.m	Matlab .m		71 udre2fit.m	Matlab .m	
32 hpl_contour.m	Matlab .m		72 usr_vhpl.m	Matlab .m	
33 igp_plane.m	Matlab .m		73 usralaska.dat	Data File	
34 igpaor.dat	Data File		74 usrconus.dat	Data File	
35 igpjoint.dat	Data File		75 usrprocess.m	Matlab .m	Primary USR simulation
36 igppor.dat	Data File		76 vpl_contour.m	Matlab .m	
37 igps4ipps.m	Matlab .m		77 wmsprocess.m	Matlab .m	Primary WMS simulation
38 init_aada.m	Matlab .m		78 wrs25.dat	Data File	
39 init_aadb.m	Matlab .m		79 xyz2llh.m	Matlab .m	
40 init_col_labels.m	Matlab .m	Column labels			

Appendix #2: MAAST Data File Formats

File 'usrconus.dat'

	Latitude°(N)	Longitude°(E)
Row 1 of 101	4.9054152e+01	-1.2317659e+02

File 'usralaska.dat'

	Latitude°(N)	Longitude°(E)
Row 1 of 43	54.8000	-130.0000

File 'wrs25.dat'

	Wrs #	Latitude°(N)	Longitude°(E)	Height (meters)
Row #1 of 25	1	34.050	-118.250	0

Files 'igpaor.dat', 'igppor.dat' and 'igpjoint.dat'

Igpaor.dat Row 1 of 161	0	142	10	205	0
Igppor.dat Row 1 of 150	0	24	50	180	0
Igpjoint.dat Row 1 of 190	0.0000000 e+00	2.4000000 e+01	5.0000000 e+01	1.8000000 e+02	0.000000 e+00

File 'geo.dat' (Entire file)

GEO PRN #	Latitude°(N)	Default Flag 0-off 1-on	Name of Button
120	-15.5	0	AOR-E
122	-54.0	1	AOR-W
131	64.0	0	IOR
134	178.0	1	POR

Notes: 1) PRNs limited to 120-138 by WAAS MOPS

2) GUI of MAAST can place up to six (6) GEO's

File 'almmops.dat'

***** Week 703 almanac for PRN-01 ***** (PRN 01 of 24 listed below)	
ID:	01
Health:	000
Eccentricity:	0.0
Time of Applicability(s):	344063.0000
Orbital Inclination(rad):	0.9599310886
Rate of Right Ascen(r/s):	0.0
SQRT(A) (m ^{1/2}):	5153.620087
Right Ascen at TOA(rad):	0.4762078504E+001
Argument of Perigee(rad):	0.0
Mean Anom(rad):	0.4679681510E+001
Af0(s):	0.0000000000E+000
Af1(s/s):	0.0000000000E+000
week:	703

Appendix #3: Table of MAAST Constants

Parameter	MAAST Name	Value
Velocity of light, m/sec	CONST_C	299792458.0
L1 frequency, Hz	CONST_F1	1575.42e6
L2 frequency, Hz	CONST_F2	1227.60e6
L1 wavelength, m	CONST_LAMBDA1	CONST_C/CONST_F1
L2 wavelength, m	CONST_LAMBDA2	CONST_C/CONST_F2
SV orbit semimajor axis, m	CONST_R_SV	26561750
Earth's grav. parameter (m^3/s^2)	CONST_MU_E	3.986005e14
Earth's angular velocity (rad/s)	CONST_OMEGA_E	7292115.1467e-11
Earth's semimajor axis, m	CONST_R_E	6378137
Earth's semiminor axis, m	CONST_B_E	6356752.314
Earth flattening constant	CONST_FLAT_E	1.0/298.257223563
altitude of the ionosphere, m	CONST_H_IONO	350000
Iono's approximate radius, m	CONST_R_IONO	CONST_R_E + CONST_H_IONO
Ionospheric constant for L1/L2	CONST_GAMMA	$(\text{CONST\_F1}/\text{CONST\_F2})^2$;
CONST_K_TEC	CONST_F1 ² *CONST_F2 ² /(CONST_F1 ² +CONST_F2 ²)/40.3;	
Seconds per day	CONST_SEC_PER_DAY	24*3600;
Seconds per week	CONST_SEC_PER_WEEK	7*CONST_SEC_PER_DAY;
	CONST_TEC2L1M	0.163;
	CONST_R_GEO	42241095.8;
	CONST_H_GEO	CONST_R_GEO - CONST_R_E;

Appendix #4: MAAST Function Map

Listing and location of MAAST Functions			
File	Line	Code text	Comments
svmrun.m	1	svmrun(gpsudrefun, geoudrefun, givefun, usrtrpfun, usrcnmpfun, wrstrpfun, wrsgpscnmpfun, wrsggeocnmpfun, wrsfile, usrfile, igpfile, svfile, geodata, tstart, tend, tstep, usrlatstep, usrlonstep, outputs, percent);	
wmsprocess.m	1	function [satdata, igpdata, wrs2satdata] = wmsprocess(alm_param, satdata, wrsdata, igpdata, wrs2satdata, gpsudrefun, geoudrefun, givefun, wrstrpfun, wrsgpscnmpfun, wrsggeocnmpfun, outputs, time, tstart, tstep, trise)	WMS Algorithm
usrprocess.m	1	[vhl, sig2_fit, sig2_uive, usr2satdata] = usrprocess(satdata, usrdata, igpdata, inv_igp_mask, usr2satdata, usrtrpfun, usrcnmpfun, alm_param, tcurr);	User Algorithm
outputprocess.m	1	outputprocess(satdata, usrdata, wrsdata, igpdata, inv_igp_mask, sat_xyz, udrei, givei, vpl, hpl, usrlatgrid, usrlonggrid, outputs, percent, udre_hist, give_hist, udrei_hist, givei_hist);	Output processing
igp_plane.m	1	[cov, r, rcm, idx] = igp_plane(xyz_igp, igp_en_hat, xyz_ipp, sig2_ipp)	
disccov.m	1	[dcov, incr] = disccov(cov)	
mt28cov.m	1	[dCov, scalef, maxrat] = mt28cov(satxyz, satxyzdot, G, sig2_mon)	
find_elaz.m	1	[el, az] = find_elaz(los_enub)	Given line of sight vectors in the East, North, Up frame this function
init_igpdata.m	1	[igpdata, inv_igp_mask] = init_igpdata(igpfile)	
check_igpsquare.m	1	[IGPs, xyIPP, nBadIGPs] = check_igpsquare(ll_ipp, mask_idx, inv_IGPmask, lat_spacing, lon_spacing, lat_base, lon_base)	base latitudes and longitudes for this square
igps4ipps.m	1	[IGPs, xyIPP, nBadIGPs] = igps4ipps(ll_ipp, IGPmask, inv_IGPmask)	this function will determine the four IGP numbers for each IPP and put it
xyz2llh.m	1	[llh] = xyz2llh(xyz)	
char_poly.m	1	[p, a0, a1, a2, a3] = char_poly(l, A)	
alm2satposvel.m	1	[prn, sv_xyz, sv_xyz_dot] = alm2xyz(time, alm_param)	
init_usrdata.m	1	[usrdata, grid_lat_grid_lon] = init_usrdata(polyfile, latstep, lonstep)	
llh2xyz.m	1	[XYZ] = llh2xyz(LLH)	
init_igps.m	1	[xyz_igp, igp_en_hat, igp_corner_den, igp_mag_lat, inv_igp_mask] = ...	
findxyz2enu.m	1	[xyz2enu] = findxyz2enu(lat, lon)	
read_yuma.m	1	alm_param = read_yuma(filename)	
avail_contour.m	1	avail = avail_contour(lats, lons, vpl, hpl, isinbnd, percent)	this function will plot the contour lines according to intervals specified
findcommon.m	1	c = findcommon(a, b)	
mysetdiff.m	1	c = mysetdiff(a, b)	
maastgui.m	1	fig = maastgui()	
checkfor2.m	1	flag = checkfor2(wrs_id)	
give_contour.m	1	give_contour(igp_mask, inv_igp_mask, givei, percent, ax)	
give_histogram.m	1	give_histogram(give_hist, givei_hist)	
gui_mexclude.m	1	gui_mexclude(obj_list, sel_obj)	
guicbfun.m	1	guicbfun(hndl)	
svm_histogram.m	1	h = svm_histogram(x, n, xi, ni, xsplit, xratio, xticks, xticklabels)	
colorb.m	1	handle = colorb(levels, ticklabels, loc)	
hpl_contour.m	1	hpl_contour(lats, lons, vpl, percent)	HPL(n_lats*n_lons, 1) in meters, this function will plot the contour lines
phmi_eq.m	1	i = phmi_eq(sig2_cp, sig2_mon, alpha, lambda)	
af_giveadd.m	1	igpdata = af_giveadd(t, igpdata, wrsdata, satdata, wrs2satdata)	assemble GIVE and apply the ceiling function
af_giveconst.m	1	igpdata = af_giveconst(t, igpdata, wrsdata, satdata, wrs2satdata)	
init_aada.m	1	init_aada()	
init_aadb.m	1	init_aadb()	
init_cnmp.m	1	init_cnmp()	
init_graph.m	1	init_graph()	
init_hist.m	1	init_hist()	
find_inv_IGPmask.m	1	inv_IGPmask = find_inv_IGPmask(IGPmask)	function creates a 35x72 matrix that points back to the IGP number.
gui_readselect.m	1	iselect = gui_readselect(objlist)	

calc_ll_ipp.m	1	ll_ipp=calc_ll_ipp(ll_usr, el, az)	% satellites, this function returns the latitudes and longitudes of the
find_ll_ipp.m	1	ll_ipp=find_ll_ipp(ll_usr, el, az, idx)	(n_los, 1) and AZ (n_los,1) to the
			different satellites, this function
calc_los_enub.m	1	los_enub=find_los_enub(los_xyzb, e_hat, n_hat, u_hat)	in E_HAT, N_HAT, and U_HAT
			respectively this function returns the
find_los_enub.m	1	los_enub=find_los_enub(los_xyzb, usr_ehat, usr_nhat, usr_uhat, losmask)	n_los
			in E_HAT, N_HAT, and U_HAT
find_los_xyzb.m	1	los_xyzb=find_los_xyzb(xyz_usr, xyz_sat, losmask)	respectively this function returns the
mapudre.m	1	mapudre(udrei, sat_llh, wrsLL)	n_los
project2user.m	1	min_rat=project2user(sat_xyz, Cov)	XYZ_USR and XYZ_SAT respectively,
svm_hist.m	1	n=svm_hist(y,x)	this function returns the n_usr*n_sat
gui_readnum.m	1	num=gui_readnum(hndl,llim,ulim,errmsg)	
obliquity2.m	1	ob2=obliquity2(el);	
mynormcdf.m	1	p = mynormcdf(x)	OBLIQUITY2 returns the square of the
			ionospheric obliquity function
intriangle.m	1	result=intriangle(x,y,corner)	Express normal CDF in terms of the
find_ja_maxratio.m	1	rootm=find_maxratio(C,X)	error function.
			this function determines if an IPP is in
			the triangle (returns 1) or not
af_geoconst.m	1	satdata = af_geoconst(satdata,wrsdata,wrs2satdata,do_mt28)	Generates from columns X, y of
			Satdata
af_udreadd.m	1	satdata = af_udreadd(satdata,wrsdata,wrs2satdata,do_mt28)	
af_udreconst.m	1	satdata = af_udreconst(satdata,wrsdata,wrs2satdata,do_mt28)	
init_satdata.m	1	satdata = init_satdata(geodata, alm_param, satdata, t)	
af_cnmpadd.m	1	sig2 = af_wrsnmpadd(del_t,el)	
af_cnmpagg.m	1	sig2_cnmp = af_wrsnmpagg(del_t,el)	
udre2fit.m	1	sig2_fit=UDRE2fit(los_xyzb, pm, sig2_udre, mt28_cov, mt28_sf)	
sig2_max_vtec.m	1	sig2_mvtec = sig2_max_vtec(t, delay, mag_lat)	SIG2_MAX_VTEC returns the variance
af_trpadd.m	1	sig2_trop = af_wrstrpadd(EI)	for the iono ceiling function
af_trpmops.m	1	sig2_trop = af_wrstrpmops(EI)	
grid2uive.m	1	sig2_uive=grid2uive(ll_ipp, IGPmask, inv_IGPmask, givei)	GIVE indices for each of the IGP's in
undersamp_threat.m	1	sig2_undersamp=undersamp_threat(radius, rel_centroid)	GIVEI(nIGPs,1), this function will
af_cnmpaad.m	1	sig2=af_usrcnmpaad(del_t,el)	
sig2_cnmp.m	1	sig2=sig2_cnmp(del_t,el)	
svm_contour.m	1	svm_contour(lons, lats, data, cnt_int, tick_text, colors, ...)	
find_trise.m	1	trise = find_trise(tmin,tmax,sinmask,alm_param,usr_xyz,...)	
udre_histogram.m	1	udre_histogram(udre_hist, udrei_hist)	
init_usr2satdata.m	1	usr2satdata = init_usr2satdata(usrdata,satdata);	
usr_vhpl.m	1	vhpl=usr_vhpl(los_xyzb, usr_idx, sig2_i)	variances for each los in SIG2_I
			(n_los,1), this function will determine
vpl_contour.m	1	vpl_contour(lats, lons, vpl,percent)	VPL(n_lats*n_lons,1) in meters, this
			function will plot the contour lines
init_wrsdata.m	1	wrsdata = init_wrsdata(wrsfile)	
quadfit.m	1	x0 = quadfit(y0,x,y)	
min_eig.m	1	y=min_eig(A)	

Appendix #5: Matrix Mapping

"alm param" matrix

n x 12 Matrix containing
almanac satellite data

PRN	ECCEN	TOA (time of applicability, absolute time since 1980 seconds)	INCLIN	RORA (rate of right ascension)	SQRT_A Square root of semi-major axis	R_ACEN Right Ascension	ARG_PERIG Argument of Perigee	MEAN_ANOMALY	AFO (seconds)	AF1	Week
-----	-------	--	--------	-----------------------------------	--	---------------------------	----------------------------------	--------------	---------------	-----	------

										sec/		
Units			sec.	rad.	rad/sec	m^1/2	rad.	rad.	rad.	sec.	sec	
Columns	1	2	3	4	5	6	7	8	9	10	11	12

Rows

satellite ₁
satellite ₂
satellite ₃
satellite ₄
satellite ₅
satellite ₆
satellite ...
satellite _n

Description

Where modified

Contains the raw almanac satellite data at time t
read_yuma.m

USR2SATDATA and WRS2SATDATA Matrixes		COL U2S INITNAN																				
		COL U2S GXYZB			COL U2S GENUB																	
n X 20 Matrix containing the WMS and User to satellite data		COL_U2S_UID (User ID)	COL_U2S_PRN	COL_U2S_GXYZB COL_U2S_LOSXYZ			COL_U2S_LOSENU			COL_U2S_GXYZB	COL_U2S_EL (User Elevation)	COL_U2S_AZ (User Azimuth)	COL_U2S_SIG2TRP (Tropo signal correction)	COL_U2S_SIG2L1MP	COL_U2S_SIG2L2MP	COL_U2S_IPPLL		COL_U2S_IPPXYZ			COL_U2S_TTRACK0 COL_U2S_MAX	
		Units		m	m	m	m	m	m	m	rad.	rad.		° Lat	° Long	m	m	m				
		Column	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Rows																						
USR2SATDATA Matrix																						
LOS from User 1 to satellite ₁																						
LOS from User 1 to satellite ...																						
LOS from User 1 to satellite _n																						
LOS from User m to satellite ₁																						
LOS from User m to satellite ₃																						
LOS from User m to satellite _n																						
Description		Contains User processing data																				
Where modified		usrprocess.m																				
Where used or called		svmrun.m called for calculations																				
WRS2SATDATA																						
LOS from WRS 1 to satellite ₁																						
LOS from WRS1 to satellite ...																						
LOS from WRS1 to satellite _n																						
LOS from WRS m to satellite ₁																						
LOS from WRS m to satellite ...																						
LOS from WRS m to satellite _n																						
Description		Contains WAAS master station processing data																				
Where modified		wmsprocess.m																				
Where used or called		af_geoconst.m called in function satdata af_giveadd.m, called in function igpdata af_udreadd.m called in function satdata svmrun.m called for calculations																				

IGPDATA Matrix

n X 29 Matrix containing IGP data		COL_IGP_BAND	COL_IGP_ID	COL_IGP_LL		COL_IGP_XYZ			COL_IGP_ISCONUS	COL_IGP_EHAT (unit vectors)			COL_IGP_NHAT (unit vectors)			COL_IGP_MAGLAT	COL_IGP_CORNERDEN											COL_IGP_GIVEI	COL_IGP_MINMON	
				° Lat	° Long	m	m	m								° Lat														
Units →	Column →	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29

Rows

IGPDATA Matrix																												
IGP #1																												
IGP #2																												
IGP #3																												
IGP #4																												
IGP #...																												
IGP #n																												

Description

Where modified

Where used or called

Contains ionispheric grid point data
init igpdata.m
svmrun.
wmsprocess.m: data from IGPMATRIX used for processing
usrprocess.m data from IGPMATRIX used for processing
outprocess.m
quicbfun.m: called in function outprocess

USRDATA and WRSDATA Matrix			COL_USR_LLH														COL_USR_MAX	
nX 17 Matrix containing WSR and User data	COL_USR_UID (user id)	COL_USR_LL			COL_USR_XYZ				COL_USR_EHAT (unit vectors)			COL_USR_NHAT (unit vectors)			COL_USR_UHAT (unit vectors)			COL_USR_INBND
		° Lat	° Long	m	m	m	m											
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	

Rows

USRDATA Matrix																		
User ₁																		
User...																		
User _n																		

Description

Where modified

Where used or called

Matrix containing the user data																
init_usrdata.m initializes the matrix and inputs the data prior to svm analysis																
init_usr2satdata.m called in function usr2satdata																
project2user.m																
guicbfun.m called in function outprocess																
outputprocess.m called in function outprocess and for plots																
svmrn.m calles for intialization of matrix, establishes sizes																

WRSDATA Matrix																		
WRS ₁																		
WRS																		
WRS _n																		

Description

Where modified

Where used or called

Matrix containing the WRS location and position data from file																
init_wrsdata.m - initializes and adds data																
af_geoconst.m called in function satdata																
af_giveadd.m called in function igpdata																
af_udread.m called in function satdata and other calculations																
guicbfun.m called in function outprocess																
outputprocess.m called in function outprocess and for plots																
svmrn.m calles for intialization of matrix, establishes sizes																
wmsprocess.m called in functions and for calculations																

Appendix #6: Block Diagrams

Rev A
11/2/01

SVM Analysis Block Diagram (Top Level View)

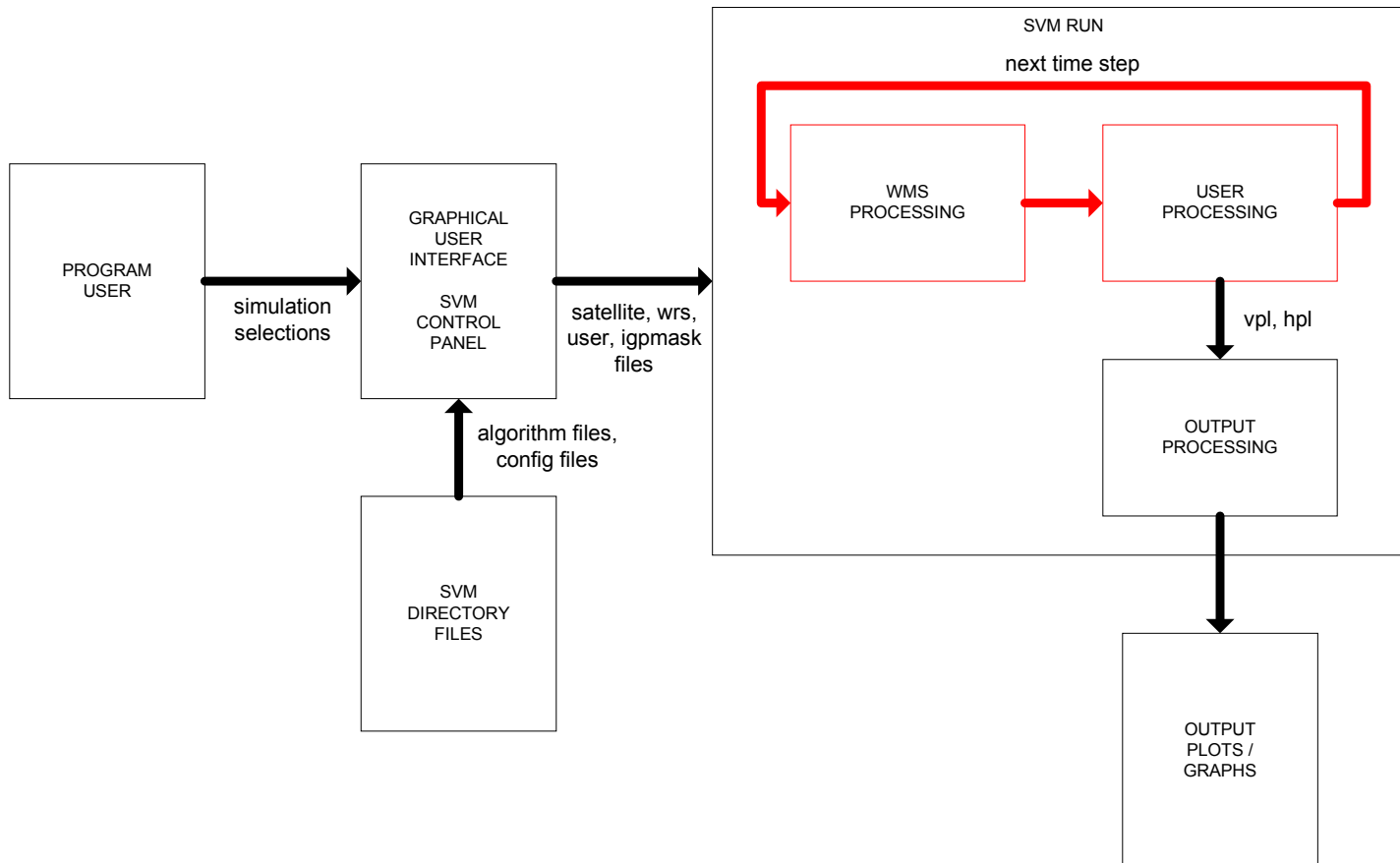


Diagram #1: SVM Analysis Simulation Top Level

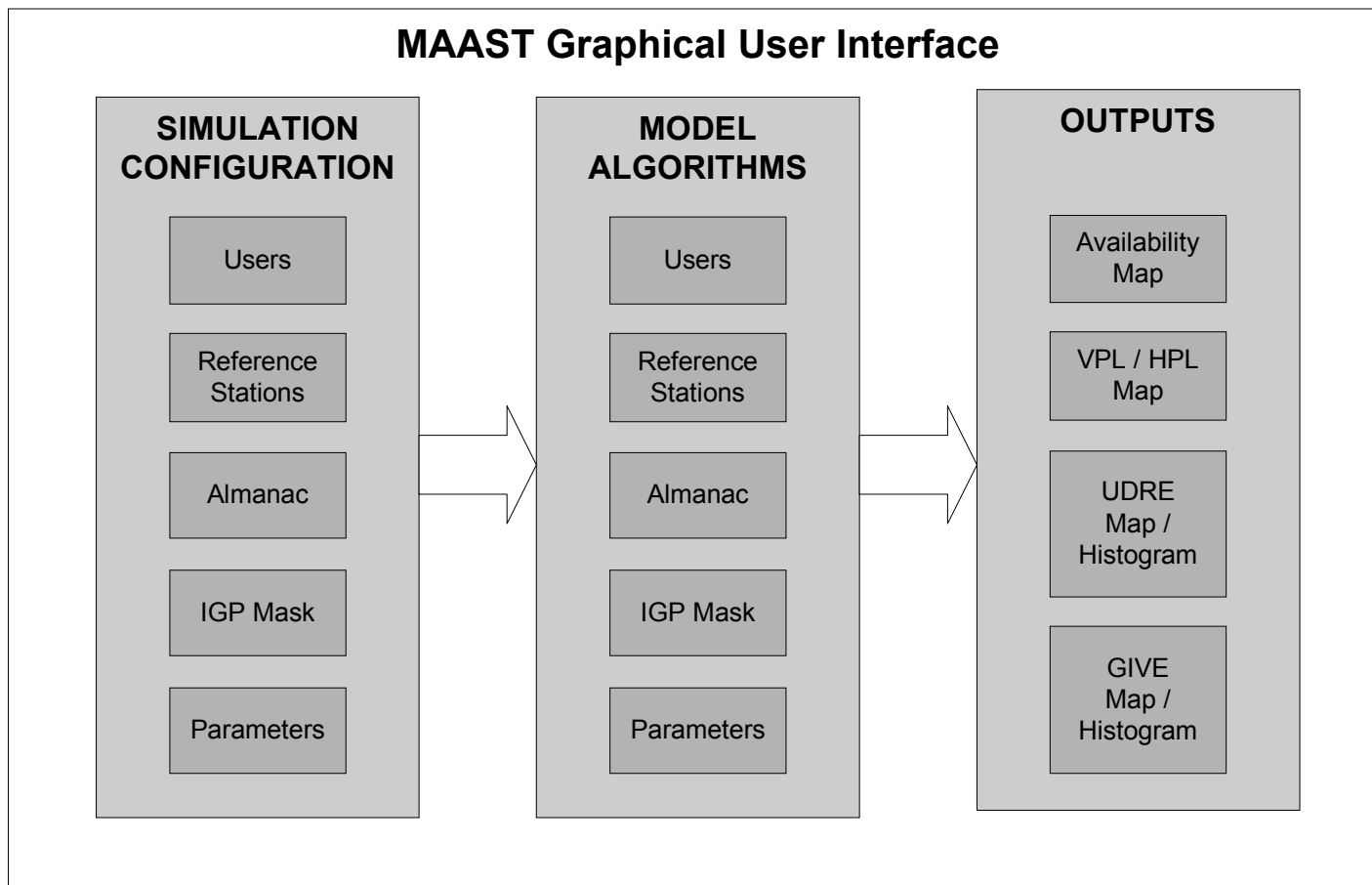
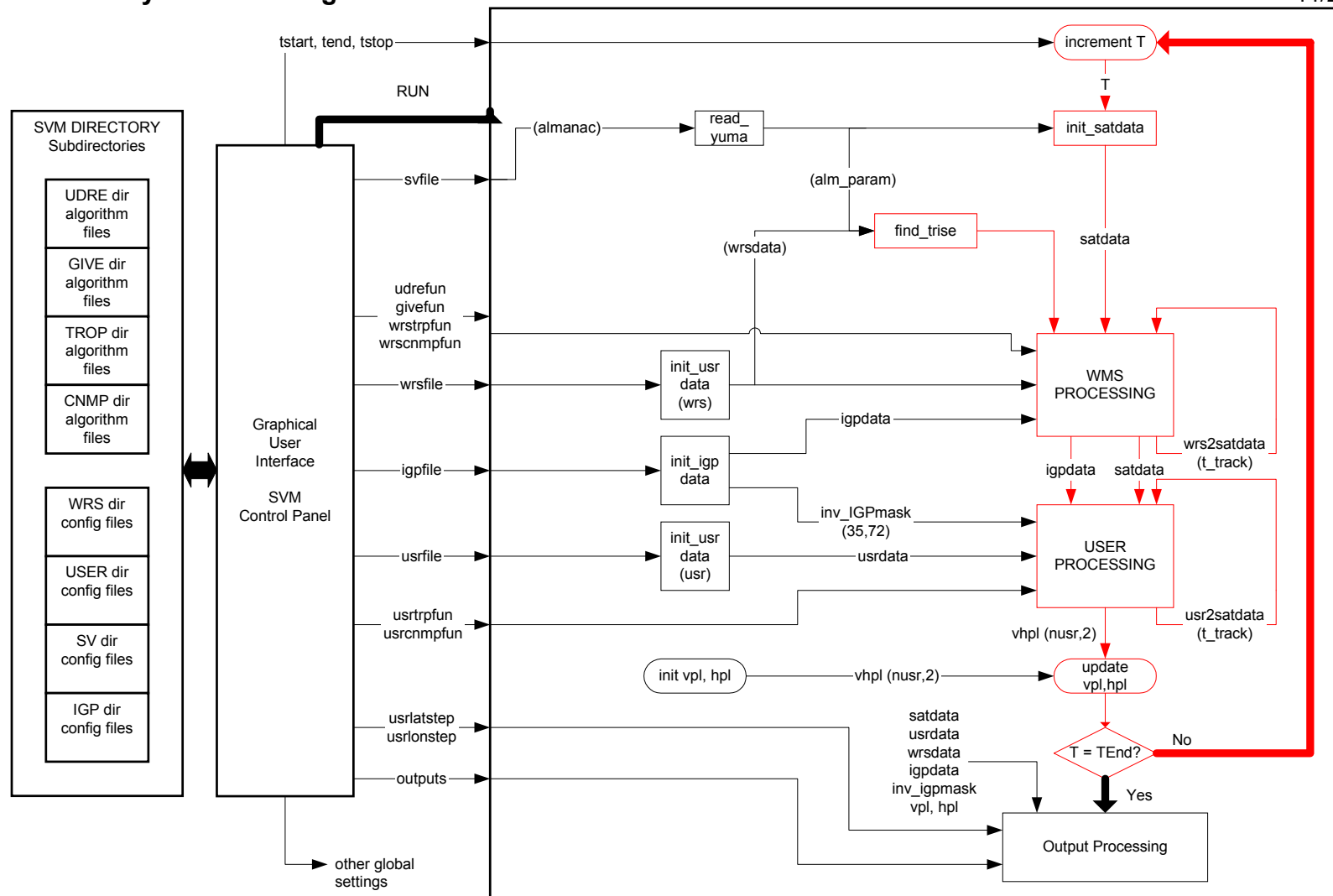


Diagram #2: MAAST Graphical User Interface

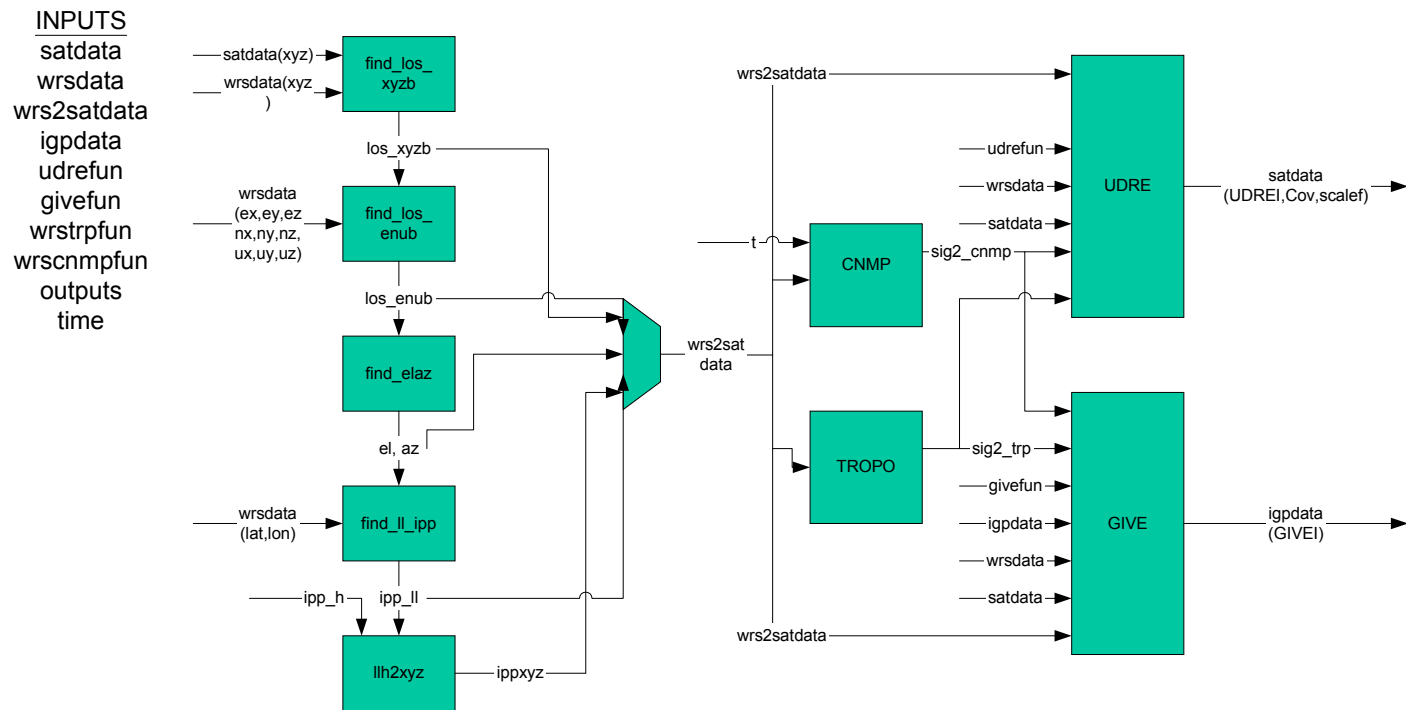


satdata (nsat,26): [id,x,y,z(m),xdot,ydot,zdot(m/s),sig2_cp,udrei,cov(16),scalef]
wrsdata (nwrs,17): [id,lat,lon(deg),ht(m),x,y,z(m),ex,ey,ez,nx,ny,nz,ux,uy,uz,isconus]
usrdata (nusr,17): [id,lat,lon(deg),ht(m),x,y,z(m),ex,ey,ez,nx,ny,nz,ux,uy,uz,isconus]

usr2data(usr2,20): [id,prn,x,y,z(m),enux,enuy,enuz,el,az,sig2trp,sig2l1mp,sig2l2mp,ippll,ippxyz,ttrack0]
wsr2data(wsr2,20): [id,prn,x,y,z(m),enux,enuy,enuz,el,az,sig2trp,sig2l1mp,sig2l2mp,ippll,ippxyz,ttrack0]
igpdata (nigp,28): [band, number, lat,lon(deg), x,y,z(m),isconus,ehat(3),nhathat(3),mag_lat,cornerden(12),igpgivei]

Diagram #3: SVM Analysis Simulation

WMS PROCESSING (UDRE/GIVE)

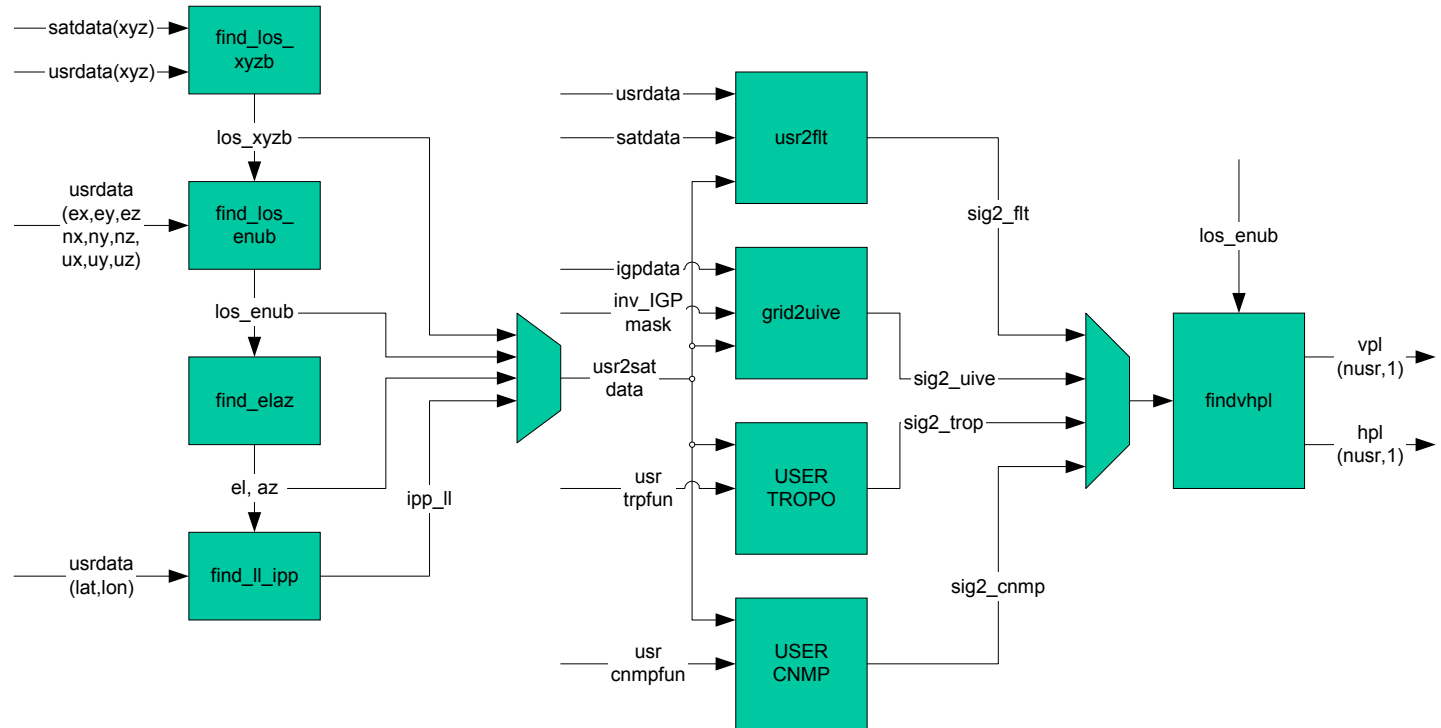


wrs2satdata (nwrs*nsat,19): [wrsid,satprn,-losx,-losy,-losz,-lose,-losn,-losu,1,el(rad),az(rad),sig2_mon,sig2_ipp,ipplat(deg),ipplon(deg),ippx,ippy,ippz,t_track0]

Diagram #4: WAAS Master Station Processing Simulation (UDRE/GIVE)

USER PROCESSING

INPUTS
satdata
usrdata
igpdata
inv_IGPmask
usrtrpfun
usrcnmpfun



usr2satdata (nusr*nsat,19): [usrid,satprn,-losx,-losy,-losz,-lose,-losn,-losu,1,el(rad),az(rad),sig2_mon,sig2_ipp,ipplat(deg),ipplon(deg),ippxyz(3),t_track0]

Diagram #5: User Processing Simulation