Velocity Distribution Functions: Bulk Flow Velocity Changing Software

Lynn B. Wilson III

August 1, 2017

Contents

1 Introduction

This software package is intended to allow a user to take an array of velocity distribution functions (VDFs), as IDL structures, and interactively alter the bulk flow velocity (i.e., change the reference frame). All the routines in this package require either the ***UMN Modified* Wind/*3DP***[1] or ***SPEDAS***[2] IDL libraries. The routines each have a detailed Man. page that explains the usage, purpose, inputs, and keywords. The Man. page also contains a list of routines that are called in any given program/function, as well as the routine which called said program/function. **Make sure your IDL paths are set correctly so that the routines and their dependencies can be found and compiled when called**.

The routines make no assumptions about the type of distribution on input, rather it takes that information from the input structures. The routines will output a structure containing arrays of structures that define the parameters used for each element in the intput array of VDFs.

1.1 Placement/Location of IDL Routines

You should have a specific directory where you run IDL to avoid making IDL search your entire computer for called routines to compile. If not, you should at least have two special places, one for the ***SPEDAS*** library and one for the ***UMN Modified* Wind/*3DP*** library. Many of the routines in the ***UMN Modified* Wind/*3DP*** library overlap with those in the ***SPEDAS*** library, so be careful how you use these together. For instance, do not place the *∼/wind_3dp_pros/* folder in the *∼/spedas_?_??/idl/* directory. The two libraries should be in separate locations, but you can alter your IDL path specifications to allow IDL to find both sets of routines from one place[3].

[1] Found at: https://github.com/lynnbwilsoniii/wind_3dp_pros

[2] Found at: http://themis.ssl.berkeley.edu/software.shtml

[3] I discuss this in Section 2.1, but I apologize for limiting the discussion to Unix/Linux/Mac OSs. There are similar methods for forcing IDL to look at specific directories on Windows machines, but I am not familiar with them.

2 IDL Startup

Below, we will very briefly explain how to start IDL so that the software can be compiled and used correctly. All routines are found in the *~/wind_3dp_pros/LYNN_PRO/vbulk_change_routines/* directory of the ***UMN Modified* Wind/*3DP*** IDL package[4].

2.1 Starting/Initializing IDL

I am going to assume that you know how to obtain the ion velocity distribution functions (VDFs) as IDL structures for the instrument you are interested in examining. Appendix A lists many of the necessary structures tags for each individual IDL structure. If you are not familiar with these structures, there are detailed crib sheets found in *~/wind_3dp_pros/wind_3dp_cribs/* directory in the ***UMN Modified* Wind/*3DP*** IDL library explaining how to obtain either *Wind*/3DP [*Lin et al.*, 1995] or THEMIS ESA [*McFadden et al.*, 2008a,b] VDFs[5].

If you are looking at THEMIS data, then you want to use the SPEDAS start-up initialization software (i.e., type *themis* in a Unix terminal at the command line). If you are looking at *Wind* data, then use the ***UMN Modified* Wind/*3DP*** start-up initialization software[6].

Note that if you use the SPEDAS software, you need to include comp_lynn_pros.pro in the *~/tdas_?_??/idl/* directory. You will also need to adjust the IDL paths in setup_themis_bash to allow IDL to look at the following directories in the *~/wind_3dp_pros/* directory: *LYNN_PRO/*, *THEMIS_PRO/*, *Coyote_Lib/*, and *rh_pros/*. Follow the same format as the supplied version of setup_themis_bash for each directory and add them to the end of SPEDAS IDL path specification.

The following should be placed in your .bash_profile or .bashrc. Here is an example of how to initialize your IDL paths using IDL's bash setup routine:

```
########## IDL setup #############
##  Let IDL's bash script define default IDL paths
source /Applications/harris/idl/bin/idl_setup.bash
##  Add current working directory path and any subdirectory paths
##    onto IDL's default paths
unset IDL_PATH
if [ ${IDL_DIR:-0} != 0 ] ; then
  export IDL_PATH ; IDL_PATH=$IDL_DIR
  ##  Make sure to recursively search subdirectories
  IDL_PATH=${IDL_PATH}:$(find $IDL_DIR -type d | tr '\n' ':' | sed 's/:$//')
  ##  Add my directories
  IDL_PATH=$IDL_PATH:+.:+$HOME/Desktop/idllibs
else
  export IDL_PATH ; IDL_PATH=:+.:+/Applications/exelisS/idl/lib:+$HOME/Desktop/idllibs
fi
```

[4]Found at: https://github.com/lynnbwilsoniii/wind_3dp_pros

[5]e.g., see my_3DP_moments_save-files.txt for 3DP or themis_esa_*_crib.txt for ESA

[6]see bash profile examples for *uidl* and *uidl64* at: https://github.com/lynnbwilsoniii/wind_3dp_pros

The following is an example of how to use a bash script to initialize IDL specific to the SPEDAS software:

```
################################################################################
# Set up SPEDAS, THEMIS
function spedas {
   # We have to unset the IDL_PATH to avoid TPLOT conflicts... kludgy.
   # If you have personal IDL routines that you normally include (as long as
   # they don't include any TPLOT routines!), you may add them to the IDL_PATH
   # after we clobber it below.
   unset IDL_PATH
   #  Define the SPEDAS path
   export SPEDAS_LBW=$HOME/Desktop/Old_or_External_IDL/SPEDAS/spedas_1_00
   #  Define the ITT IDL path
   IDL_LOC='/Applications/harris/idl/'
   source ${IDL_LOC}/bin/idl_setup.bash
   source $SPEDAS_LBW/idl/projects/themis/setup_themis_bash
   unset DYLD_LIBRARY_PATH
   export DYLD_LIBRARY_PATH=/opt/X11/lib/flat_namespace
   idl
   ##  Reset bash on exit
   source $HOME/.bash_profile
}
```

It should be placed in your .bash_profile or .bashrc. Note I have added comments specifying where you need to alter specific directory paths that will be specific to your machine.

The following shows you how to alter the file ~/spedas_?_??/idl/themis/setup_themis_bash. Here are some examples of how to alter the default IDL paths in that file:

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;  Configure the directory paths below accordingly for your machine
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

# Location where the IDL code (including THEMIS code) is installed
# (i.e., the directory which
# contains (ssl_general, external, themis)
if [ ${IDL_BASE_DIR:-0} == 0 ] ; then
;;   export IDL_BASE_DIR ; IDL_BASE_DIR=/disks/socware/idl
   export IDL_BASE_DIR ; IDL_BASE_DIR=/Users/lbwilson/Desktop/Old_or_External_IDL/SPEDAS/spedas_1_00/idl
fi

#  Location of extra utility IDL code
if [ ${IDL_EXTRA_DIR:-0} == 0 ] ; then
   export IDL_EXTRA_DIR ; IDL_EXTRA_DIR=/Users/lbwilson/Desktop/idllibs/codemgr/libs/utility
fi

#  Location of my IDL code
if [ ${IDL_LYNN_PRO_DIR:-0} == 0 ] ; then
   export IDL_LYNN_PRO_DIR ; IDL_LYNN_PRO_DIR=/Users/lbwilson/Desktop/swidl-0.1/wind_3dp_pros/LYNN_PRO
   ;;  Make sure to recursively search subdirectories
   IDL_LYNN_PRO_DIR=${IDL_LYNN_PRO_DIR}:$(find ~/Desktop/swidl-0.1/wind_3dp_pros/LYNN_PRO -type d | tr '\n' ':' | sed 's/:$//')
fi

if [ ${IDL_THEMIS_PRO_DIR:-0} == 0 ] ; then
   export IDL_THEMIS_PRO_DIR ; IDL_THEMIS_PRO_DIR=/Users/lbwilson/Desktop/swidl-0.1/wind_3dp_pros/THEMIS_PRO
fi

#  Location of Rankine-Hugoniot Solver utility IDL code
if [ ${IDL_RHS_DIR:-0} == 0 ] ; then
   export IDL_RHS_DIR ; IDL_RHS_DIR=/Users/lbwilson/Desktop/swidl-0.1/wind_3dp_pros/rh_pros
fi

# make sure IDL_PATH is intialized before we add THEMIS paths to it
export IDL_PATH; IDL_PATH=${IDL_PATH:-'<IDL_DEFAULT>'}

# Set path for all IDL source code:
IDL_PATH=$IDL_PATH':'+$IDL_BASE_DIR':'+$IDL_EXTRA_DIR':'+$IDL_LYNN_PRO_DIR
IDL_PATH=$IDL_PATH':'+$IDL_RHS_DIR':'+$IDL_THEMIS_PRO_DIR
```

where you would need to change the following partial directory path:

```
/Users/lbwilson/Desktop/swidl-0.1/
```

to a path specific to your computer that points to the location of the *~/wind_3dp_pros/* directory. The corresponding partial path would also need to be changed in comp_lynn_pros.pro before calling. In comp_lynn_pros.pro, the routines with the following directory paths can be commented out as well:

```
/Users/lbwilson/Desktop/idllibs/*
/Users/lbwilson/Desktop/swidl-0.1/IDL_stuff/*
```

since they are idiosyncratic to my computer and not included in the ***UMN Modified* Wind/*3DP*** IDL library. Note that the * is use to represent a wild card flag here indicating all subsequent subdirectory path extensions beyond what is shown.

2.2 IDL Structures

The IDL structures may require minor modification[7] if you are using data from the THEMIS spacecraft [*Angelopoulos*, 2008]. Once you retrieve an array of burst data structures using thm_part_dist_array.pro, it is important to note that the THETA and PHI (see Appendix A for definition) angle bin tags are defined in the DSL coordinate system. It is also important to note that some of the structure tags differ from the 3DP structure tags (e.g., *VSW* is not found). The ***UMN Modified* Wind/*3DP*** IDL library uses the *VSW* tag for the bulk flow velocity while the SPEDAS software uses *VELOCITY*.

The routine, modify_themis_esa_struc.pro, is a vectorized routine that adds the appropriate structure tags necessary for using THEMIS ESA IDL structures with the ***UMN Modified* Wind/*3DP*** software. Then one can take that modified array of IDL structures and pass it to the vectorized routine, rotate_esa_thetaphi_to_gse.pro, which rotates the THETA and PHI angles to GSE coordinates. This routine will also add the corresponding GSE magnetic field and bulk flow velocities (using TPLOT handles with the **MAGF_NAME** and **VEL_NAME** keywords) to the input array of structures so that everything is in the same coordinate basis. **Be careful! Both of these routines modify the input structures so you may wish to make copies.**

[7]Technically, this software does not require these modifications so long as the *VELOCITY* and *MAGF* tag values are in the DSL coordinate basis. However, some of the routines will check for tags that are added by my alteration routines and will kick you out if they are not altered. So just make a copy of the IDL structure arrays, modify them, and then call this software.

3 Plot Outputs

3.1 Plot Examples

This section will explain/describe the basic anatomy of the plots produced by the routines described below. Figure 1 shows the basic anatomy of the ion velocity distribution functions (VDFs) that will be shown herein[8]. The basic setup of this example will reflect the all the VDFs presented. Note that the crosshairs (horizontal red line and vertical blue line) in the contour plot are commandable (i.e., the user can change them), which define where the color-coded cuts of the VDF, shown in the panel below, are calculated. Meaning, if the red line in the crosshair was shifted vertically 100 km/s, then the red line in the cuts plot below would change to indicate the cut along this line through the VDF.

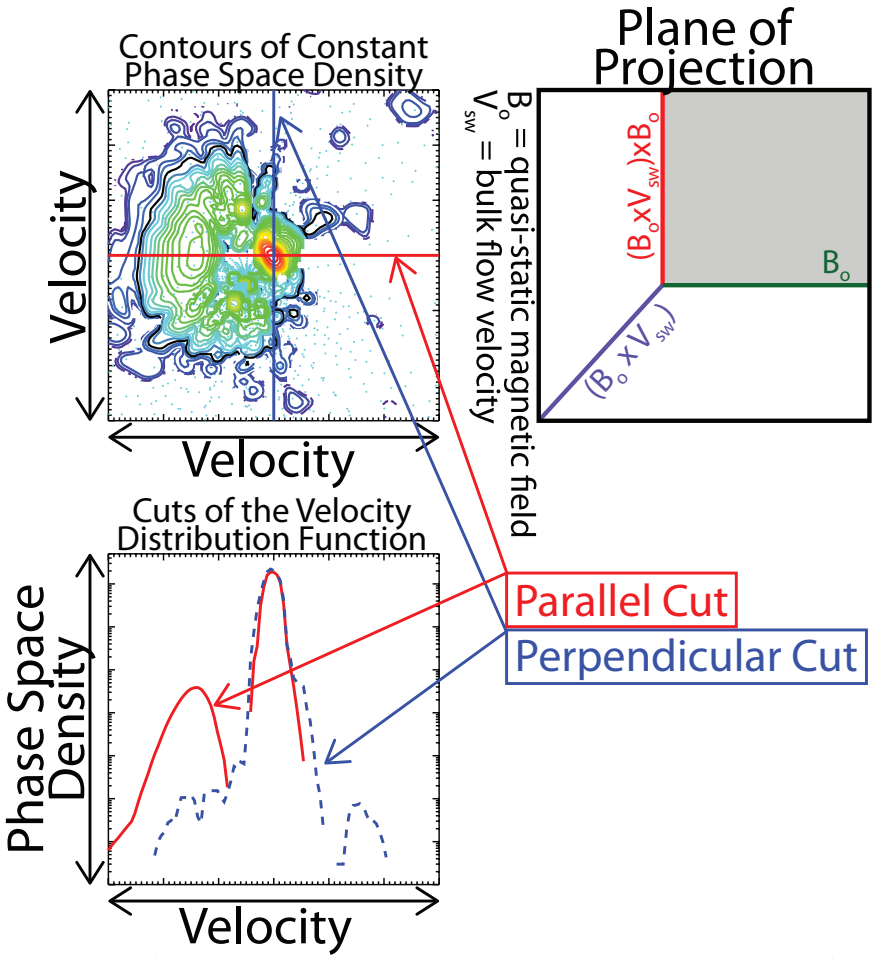

Figure 1: An example ion velocity distribution observed by the THEMIS IESA electrostatic analyzer. The figure shows the contour plot, cuts of the distribution, and plane of projection for the contour plot (indicated by the shaded region in upper-right-hand corner panel). In the contour plot, red(purple) contours correspond to the highest(lowest) phase space densities for this distribution. The color scale for the contours will be indicated by the range of phase space densities in the cuts plot directly below. The cut lines shown in the bottom panel are color-coded and correspond to the crosshairs in the contour panel. The velocity will be shown in 1000's km/s and the phase space densities in $s^3cm^{-3}km^{-3}$.

The example shows the contours projected onto the plane containing the average bulk flow velocity ($\mathbf{V}_{sw}$ or $\mathbf{V}_{bulk}$) and quasi-static magnetic field ($\mathbf{B}_o$), centered on the origin defined by the value of $\mathbf{V}_{bulk}$. All the VDFs shown herein will lie in the same plane, but the routines allow the user to use any of the three planes

[8]Figures 1 and 2 are from an old version of my software that made projections onto the plane defined by the vectors $\mathbf{V}_{bulk}$ and $\mathbf{B}_o$ (panel inset to right of contour) and the reference frame defined by the value of $\mathbf{V}_{bulk}$. The new/current version of the software produces nearly identical plots, but the contours are now true 2D slices through the plane defined by two commandable vectors, ***VEC1*** and ***VEC2***, centered on the origin defined by a commandable vector defining the reference frame, ***VFRAME***.

shown in the upper-right-hand panel of Figure 2. These distributions do not assume gyrotropy. For more examples of these types of plots, see *Wilson III et al.* [2009, 2010, 2012, 2013a,b, 2014a,b].

The purpose of this software is to allow the user to interactively change $\mathbf{V}_{bulk}$. Many studies plot VDFs in the spacecraft frame, however, I have found that no matter how contour plots like Figure 1 are created the reference frame matters. Figure 2 shows contours of constant phase space density in three different reference frames (columns), projected onto three different planes (rows) of the coordinate basis defined by the shadowed planes shown in the insets to the right of the contours. As you can see, the interpretation of the distributions might change as a consequence of not being in the bulk flow frame (right-hand column of contours)[9].

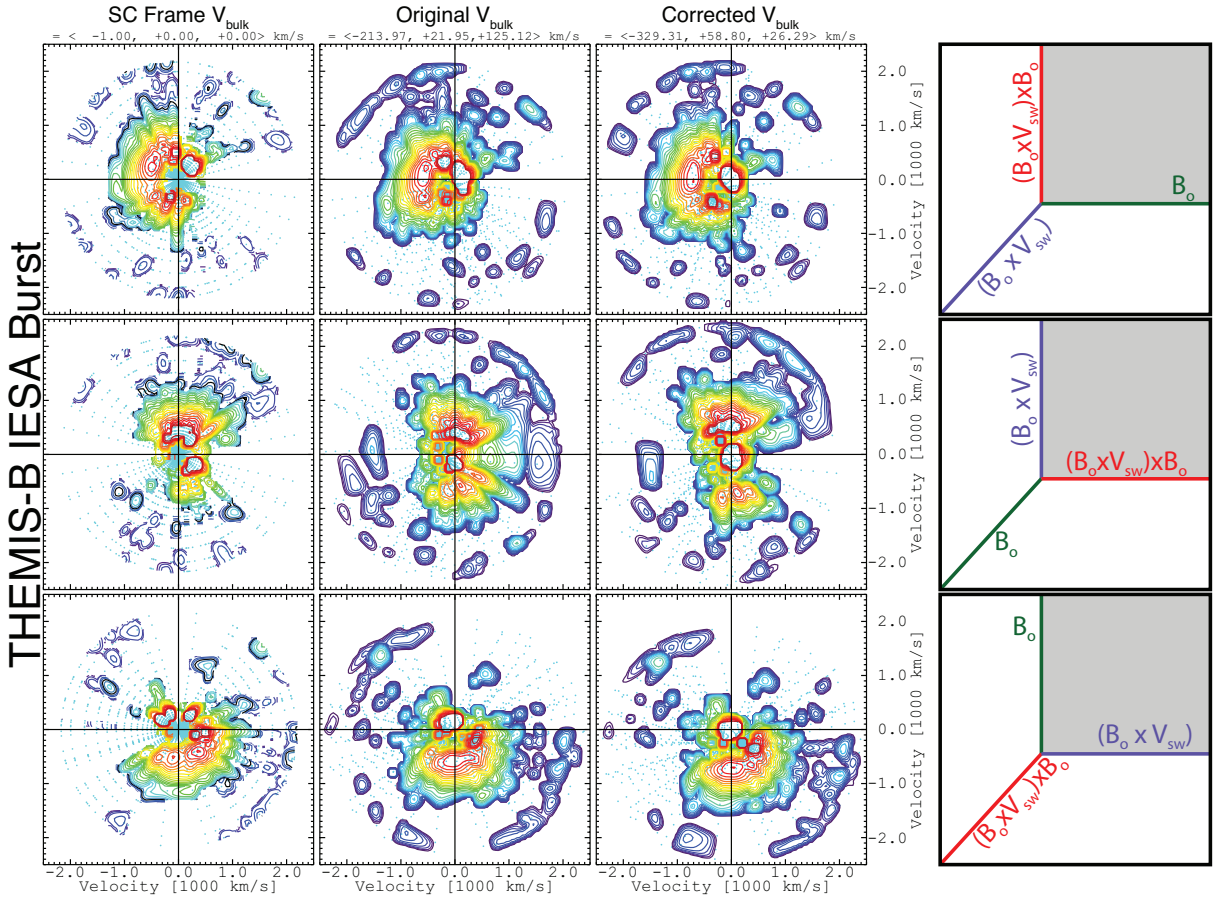

Figure 2: An example showing an ion particle velocity distribution function, observed by IESA in burst mode, in three different reference frames (columns) projected onto three different planes (rows). The shock normal (red arrow) and spacecraft–to–Earth (magenta arrow) vectors are projected onto each contour for reference. The three reference frames are defined by $\mathbf{V}_{bulk}$ at the top of each column. The three different planes defined by the shaded region in the coordinate axes shown in right-hand column. Each contour plot shows contours of constant phase space density (uniformly scaled from $1{\times}10^{-13}$ to $1{\times}10^{-7}$ $s^3cm^{-3}km^{-3}$, where red is high) versus velocity. The velocity axes range from ±1500 km/s and the crosshairs show the location of the origin. In the third column, a circle of constant energy defining the gyrospeed of specularly reflected ions is shown [e.g., *Gosling et al.*, 1982]. Adapted from Figure I:6 in *Wilson III et al.* [2014a].

[9]Note that in the old version of this software, i.e., the beam fitting routines, the plane is also dependent on your definition of $\mathbf{V}_{bulk}$, which if inaccurate, can cause the triangulation routines to project data onto a plane that does not contain a significant fraction of the core and/or beam components (e.g., compare top row of contour plots in Figure 2). The new/current software allows the user to define the coordinate basis with vectors independent of $\mathbf{V}_{bulk}$ if they wish.

3.2 Plot Descriptions

The plots are constructed in the reference frame defined by a commandable ***VFRAME*** input (i.e., defines the vector $\mathbf{V}_{bulk}$) and a coordinate basis defined by commandable ***VEC1*** and ***VEC2*** inputs. A new orthonormal coordinate basis (NCB) can be constructed from any two arbitrary, non-parallel unit vectors in an input coordinate basis (ICB)[10], $\hat{\mathbf{V}}_1{}^{ICB}$ and $\hat{\mathbf{V}}_2{}^{ICB}$, by defining a matrix that rotates data in the ICB to the NCB, $\mathbb{A}$. We can construct the rotation matrix by:

I. Define one of the basis unit vectors in the NCB to be parallel to $\hat{\mathbf{V}}_1{}^{ICB}$. For simplicity, let's use $\boldsymbol{\zeta} \equiv \hat{\mathbf{V}}_1{}^{ICB} = \hat{\mathbf{z}}^{NCB}$.

II. So long as $\hat{\mathbf{V}}_1{}^{ICB} \times \hat{\mathbf{V}}_2{}^{ICB} \neq 0$ is satisfied, we can define the second basis unit vector as $\boldsymbol{\beta} \equiv \hat{\mathbf{y}}^{NCB} = \hat{\mathbf{V}}_1{}^{ICB} \times \hat{\mathbf{V}}_2{}^{ICB}$ (**Note:** You need to renormalize these products at each step to avoid rounding errors etc. from making the determinant of $\mathbb{A}$ deviate from unity).

III. The final unit vector completes the right-handed set, thus is defined as $\boldsymbol{\eta} \equiv \hat{\mathbf{x}}^{NCB} = \left(\hat{\mathbf{V}}_1{}^{ICB} \times \hat{\mathbf{V}}_2{}^{ICB}\right) \times \hat{\mathbf{V}}_1{}^{ICB}$.

Therefore, the rotation matrix is given by:

$$\mathbb{A} = \begin{bmatrix} \eta_x & \eta_y & \eta_z \\ \beta_x & \beta_y & \beta_z \\ \zeta_x & \zeta_y & \zeta_z \end{bmatrix} \tag{1}$$

such that the following are satisfied:

$$\mathbb{A} \cdot \boldsymbol{\eta} = \langle 1, 0, 0 \rangle \tag{2a}$$

$$\mathbb{A} \cdot \boldsymbol{\beta} = \langle 0, 1, 0 \rangle \tag{2b}$$

$$\mathbb{A} \cdot \boldsymbol{\zeta} = \langle 0, 0, 1 \rangle \ . \tag{2c}$$

There is an important thing to note here. If the coordinate vectors used to create $\mathbb{A}$ are not orthonormal, then the correct rotation tensor is given by $\mathbb{R} = (\mathbb{A}^T)^{-1}$, or the inverse transpose of $\mathbb{A}$. The need to perform the inverse transpose of $\mathbb{A}$ arises from the non-orthogonal nature of the NCB basis. If the NCB were created from an orthogonal basis, then $\mathbb{A}$ would be an orthogonal matrix, which means $\mathbb{A}^T = \mathbb{A}^{-1}$. Further, so long as $\mathbb{A}$ is invertible and orthogonal, then the following is also true $(\mathbb{A}^T)^{-1} = (\mathbb{A}^{-1})^T$. Thus, an orthogonal NCB basis would imply $\mathbb{R} = (\mathbb{A}^T)^{-1} = (\mathbb{A}^T)^T = \mathbb{A}$. However, we were careful to construct $\mathbb{A}$ from an orthonormal set of basis vectors, so this is not an issue.

[10]Note that so long as the vector components are not parallel to within, say, floating point precision there is no need for the use of quaternions, i.e., we need not worry about gimbal lock.

4 IDL Routines

The routines explained herein are found in the *~/wind_3dp_pros/LYNN_PRO/vbulk_change_routines/* subdirectory of the *~/wind_3dp_pros/LYNN_PRO/* directory of the ***UMN Modified* Wind/3DP** IDL package[11]. Below, the manner in which the routines are called will be discussed and how to format the input IDL structure will be discussed as well.

4.1 IDL Routine Outline

Below I outline the routines and their functions by category:

I. Main Routines

- **a:** wrapper_vbulk_change_thm_wi.pro: This is the main wrapping routine and the **only** routine the user should call directly.
- **b:** vbulk_change_vdf_plot_wrapper.pro: This is the main wrapping routine for plotting and interactively changing parameters.

II. Prompting Routines

- **a:** vbulk_change_keywords_init.pro: This routine initializes the plotting, saving, and preference parameters before anything else is performed.
- **b:** vbulk_change_change_parameter.pro: This routine determines what, if any, parameters the user would like to change and interactively re-plots to verify the changes made. It serves as a wrapping routine for vbulk_change_options.pro.
- **c:** vbulk_change_options.pro: This routine controls the dynamic plotting options defined by user input. It also serves as a wrapping routine for the prompting routine vbulk_change_prompts.pro.
- **d:** vbulk_change_prompts.pro: This is the main prompting routine that tests/verifies the format and validity of user input and then returns the results to the calling routine.
- **e:** vbulk_change_list_options.pro: Prints to screen all the optional values allowed for user input and their purpose.

III. Testing/Error Handling Routines

- **a:** vbulk_change_test_windn.pro: This routine tests the user defined device window number, i.e., it makes sure the user did not define a bad input that would cause WINDOW.PRO to fail.
- **b:** vbulk_change_test_plot_str_form.pro: This routine tests the structure format of the plotting structure used by and returned by general_cursor_select.pro[12].
- **c:** vbulk_change_test_cont_str_form.pro: This routine tests the structure format of the main informational structure passed between nearly all routines that contains all the relevant information for producing contour plots with general_vdf_contour_plot.pro[13].
- **d:** vbulk_change_test_vdf_str_form.pro: This routine tests the structure format of the input velocity distribution functions (VDFs). The format should match the output from the routine conv_vdfidlstr_2_f_vs_vxyz_thm_wi.pro[14].
- **e:** vbulk_change_test_vdfinfo_str_form.pro: This routine tests the structure format of the informational structure for each input VDF. The structures passed to this routine contain time stamps, spacecraft, instrument, spacecraft potential, etc. informative tags used by several routines within the *Vbulk Change Software* library.

IV. Other Routines

- **a:** vbulk_change_get_default_struc.pro: This routine creates a structure filled with default values for the structure passed as the *CONT_STR* keyword throughout and tested by the vbulk_change_test_cont_str_form.pro routine.
- **b:** vbulk_change_get_fname_ptitle.pro: This routine returns the file name and plot title for output corresponding to the ith element of the VDF structure array defined by the *INDEX* keyword.
- **c:** vbulk_change_print_index_time.pro: This routine prints to screen the available particle VDF dates, times, and array indices.

[11] Found at: https://github.com/lynnbwilsoniii/wind_3dp_pros
[12] Routine found in *~/wind_3dp_pros/LYNN_PRO/plotting_routines/*.
[13] Routine found in *~/wind_3dp_pros/LYNN_PRO/esa_mcp_software/*.
[14] Routine found in *~/wind_3dp_pros/LYNN_PRO/esa_mcp_software/*.

4.2 Routine Output

Assume that on input, we provided an N–element array of IDL structures called ***DATA***. On output, the routine will define a user named keyword called ***STRUC_OUT*** with structure tags *CONT_STR* and *VDF_INFO*. Each tag will contain N–element array of IDL structures where the i^{th} element will contain information about plotting, reference frame, and coordinate basis definitions for the i^{th} in ***DATA***[15]. Below I define the tags contained within each structure.

In the descriptions, I will use the following definitions: **SCF** ≡ spacecraft frame (i.e., the measurement/instrument frame); **PRF** ≡ plasma rest frame (or bulk flow rest frame or solar wind frame); **NCB** ≡ new orthonormal coordinate basis constructed from ***VEC1*** and ***VEC2***; **ICB** ≡ input coordinate basis (i.e., the measurement/instrument coordinate basis); and **VDF** ≡ velocity distribution function.

[15]Note that the i^{th} element to which we refer here is defined the by the user commandable input ***INDEX***.

***CONT_STR* Tags:**

I. **VFRAME** ≡ [3]-Element [**double-precision**] array defining the 3-vector velocity [km/s] of the PRF relative to the SCF used to transform[16] the velocity distribution into the bulk flow reference frame
[Default = [0d0,0d0,0d0]]

II. **VEC1** ≡ [3]-Element [**double-precision**] vector to be used for "parallel" direction in a 3D rotation of the input data (see Section 3.2 for details)
[Default = [1d0,0d0,0d0]]

III. **VEC2** ≡ [3]-Element [**double-precision**] vector to be used with **VEC1** to define a 3D rotation matrix (see Section 3.2 for details)
[Default = [0d0,1d0,0d0]]

IV. **VLIM** ≡ Scalar [**double-precision**] defining the velocity [km/s] range for the plot axes
[Default = 1e3]

V. **NLEV** ≡ Scalar [**long integer**] defining the # of contour levels to plot
[Default = 30L]

VI. **XNAME** ≡ Scalar [**string**] defining the name of vector associated with the **VEC1** input
[Default = **'X'**]

VII. **YNAME** ≡ Scalar [**string**] defining the name of vector associated with the **VEC2** input
[Default = **'Y'**]

VIII. **SM_CUTS** ≡ If set, program smoothes the cuts of the VDF before plotting
[Default = FALSE]

IX. **SM_CONT** ≡ If set, program smoothes the contours of the VDF before plotting
[Default = FALSE]

X. **NSMCUT** ≡ Scalar [**long integer**] defining the # of points over which to smooth the 1D cuts of the VDF before plotting
[Default = 3L]

XI. **NSMCON** ≡ Scalar [**long integer**] defining the # of points over which to smooth the 2D contour of the VDF before plotting
[Default = 3L]

XII. **PLANE** ≡ Scalar [**string**] defining the plane projection to plot with corresponding cuts
[Let $\hat{\mathbf{V}}_1{}^{ICB}$ = **VEC1**, $\hat{\mathbf{V}}_2{}^{ICB}$ = **VEC2**]. Allowable inputs include:

- **a:** **'xy'** : $\hat{\mathbf{V}}_1{}^{ICB}$ and $\left(\hat{\mathbf{V}}_1{}^{ICB} \times \hat{\mathbf{V}}_2{}^{ICB}\right) \times \hat{\mathbf{V}}_1{}^{ICB}$ define horizontal and vertical axes
- **b:** **'xz'** : $\left(\hat{\mathbf{V}}_1{}^{ICB} \times \hat{\mathbf{V}}_2{}^{ICB}\right)$ and $\hat{\mathbf{V}}_1{}^{ICB}$ define horizontal and vertical axes
- **c:** **'yz'** : $\left(\hat{\mathbf{V}}_1{}^{ICB} \times \hat{\mathbf{V}}_2{}^{ICB}\right) \times \hat{\mathbf{V}}_1{}^{ICB}$ and $\left(\hat{\mathbf{V}}_1{}^{ICB} \times \hat{\mathbf{V}}_2{}^{ICB}\right)$ define horizontal and vertical axes

[Default = 'xy']

XIII. **DFMIN** ≡ Scalar [**double-precision**] defining the minimum allowable phase space density to plot, which is useful for ion distributions with large angular gaps in data (prevents lower bound from falling below **DFMIN**)
[Default = 1d-20]

XIV. **DFMAX** ≡ Scalar [**double-precision**] defining the maximum allowable phase space density to plot, which is useful for distributions with data spikes (prevents upper bound from exceeding **DFMAX**)
[Default = 1d-2]

XV. **DFRA** ≡ [2]-Element [**double-precision**] array specifying the VDF range in phase space density [e.g., # s^{+3} km^{-3} cm^{-3}] for the cuts and contour plots
[Default = [**DFMIN**,**DFMAX**]]

XVI. **V_0X** ≡ Scalar [**double-precision**] defining the velocity [km/s] along the X-Axis (horizontal) to shift the location where the perpendicular (vertical) cut of the VDF will be performed
[Default = 0d0]

XVII. **V_0Y** ≡ Scalar [**double-precision**] defining the velocity [km/s] along the Y-Axis (vertical) to shift the location where the parallel (horizontal) cut of the VDF will be performed
[Default = 0d0]

XVIII. **SAVE_DIR** ≡ Scalar [**string**] defining the directory where the plots will be stored

[16] A relativistically correct Lorentz transformation is used assuming an incompressible Liouville's theorem.

[Default = current working directory]

XIX. **FILE_PREF** ≡ Scalar [**string**] defining the prefix associated with each PostScript plot on output [Default = **'VDF_ions'**]

XX. **FILE_MIDF** ≡ Scalar [**string**] defining the file name middle which includes information about the plane of projection and number

***VDF_INFO* Tags:**

I. **SE_T** ≡ [2]-Element [**double-precision**] array defining to the start and end times [Unix] of the VDF

II. **SCFTN** ≡ Scalar [**string**] defining the spacecraft name [e.g., **'Wind'** or **'THEMIS-B'**]

III. **INSTN** ≡ Scalar [**string**] defining the instrument name [e.g., **'3DP'** or **'ESA'** or **'EESA'** or **'SST'**]

IV. **SCPOT** ≡ Scalar [**single-precision**] defining the spacecraft electrostatic potential [eV] at the time of the VDF

V. **VSW** ≡ [3]-Element [**single-precision**] array defining to the solar wind velocity [km/s] 3-vector at the time of the VDF[17]

VI. **MAGF** ≡ [3]-Element [**single-precision**] array defining to the quasi-static magnetic field [nT] 3-vector at the time of the VDF

[17]This may not be the same as the corresponding **VFRAME** tag for the same VDF. The **VFRAME** tag is the one that is altered by the routines, this tag will correspond to the value defined by either the **VELOCITY** or **VSW** tag within each **DATA** structure.

5 Prompt Information

This section will provide more details about most of the prompts the user will encounter. As a general rule, informational outputs will be surrounded by:

```
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|
blah blah blah... pontificate
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|
```

prompting outputs will be surrounded by:

```
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
blah blah blah... enter something
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
```

Often times the routines will inform the user what display window they are working with, which is important for cursor routines. These outputs will be surrounded by:

```
<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>
blah blah blah... Window number and title
<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>
```

These prompts are generated by vbulk_change_prompts.pro, which is called by other wrapping routines. In most cases, there will be information provided as to the type of expected input (i.e., string vs. float) and what should be the format. If the user enters an incorrect format, the general prompting routine will catch the error and prompt the user again. However, incorrect input format can break the code in some places so please pay attention to instructions.

When the user is prompted and the last part of the prompt contains *(y/n)*, then the only acceptable inputs are y, n[18]. Pay attention to both the prompt and the information provided before the prompt for input format information etc.

[18] Note that at any prompt the user may enter q to quit. An input of q will often result in quitting the current program but can also be used to exit a loop, exit one of the prompting routines, or to stop changing something.

5.1 Prompt Information: Plotting Parameter Initialization

The first set of prompts are initialized by vbulk_change_keywords_init.pro, and are as follows:

```
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|
[Type ‘q’ to quit at any time]

You can print the indices prior to choosing or enter the value
if you already know which distribution you want to examine.

The default (and current) index is 00000
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Do you wish to print the indices prior to choosing (y/n)?n
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Enter a value between 0000 and 1999:  200
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
```

This prompt is obvious as it asks the user to define which VDF in the array of VDFs to plot first. The next prompt asks whether the user wants to alter the first unit vector used to construct the orthonormal basis, **VEC1**, for plotting as follows:

```
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|
     You can estimate a new vec1 using the command line.
The end result will be used as the new 'parallel' direction in
the orthonormal coordinate basis constructed for plotting the
VDFs.  The VDF will be re-plotted after user is satisfied with
the new vec1 estimate.

[Type ‘q’ to quit at any time]
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Do you wish to keep the current value of vec1 = < +1.000, +0.000, +0.000 > [units]? (y/n):  n
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>


-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|
     You have chosen to enter a new estimate for vec1
on the command line.  You will be prompted to enter each
component separately.
Then you will be prompted to check whether you agree
with this result.

*** Remember to include the sign if the component is < 0. ***

[Type ‘q’ to quit at any time]
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
```

```
Enter a new value for vec1_x [units] (format = XXXX.xxx):  1d0
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Enter a new value for vec1_y [units] (format = XXXX.xxx):  0d0
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Enter a new value for vec1_z [units] (format = XXXX.xxx):  0d0
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>


-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|
The old/current vec1 is < +1.000, +0.000, +0.000 > units.

The new vec1 is < +1.000, +0.000, +0.000 > units.

[Type ‘q’ to quit at any time]
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Do you wish to use this new value of vec1 (y/n):  y
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
```

The next prompt asks the same question for the second unit vector, **VEC2**, used to construct the orthonormal basis for plotting. Note that entering the unit vectors $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ for **VEC1** and **VEC2** is equivalent to plotting the data in the input coordinate basis.

Immediately following that, the program prompts:

```
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|
You can enter a new value for VLIM (format = XXXX.xxx).
[Type ‘q’ to quit at any time]
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Do you wish to keep the current value of vlim = 1.00e+03 [km/s]? (y/n):  n
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
```

This prompt asks the user whether they would like to use the default value[19] for the velocity range limit, **VLIM**. Depending on the situation, I type n and enter some reasonable velocity range (e.g., 1500 km/s). If I want to examine something close to the center of the contour plot in the bulk flow frame, I use 1500 km/s, e.g., for *Wind*/3DP PESA High or THEMIS IESA. If I want to examine the entire distribution, then I use a larger range like 2500 km/s[20]. Let's say I type n then we get the following:

```
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Enter a value > 77.69 [km/s] and < c [~300,000 km/s]:  5d2
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
```

[19]derived from maximum energy of input data structures
[20]Note that these values only apply to ion detectors that have upper energy limits near ∼30 keV.

The routine then does the usual verification that I indeed want to keep/use this value (this always happens, so I will stop mentioning it for brevity). Following this, the routine asks the user if they wish to use a different number of contour levels (i.e. the number of lines shown on the contour plots), defined by the variable **NLEV**.

```
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Do you wish to keep the current value of nlev = 30 [ ]? (y/n):  y
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
```

I recommend using the default value of 30 because including more clutters the plots and less can reduce resolution. The input is an integer value.

The next prompt asks the user about the character name associated with **VEC1**. The user can enter most any string but keep it short as this is used to define the horizontal and vertical axis labels in contour plots. For now, I will just use the default.

```
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Do you wish to keep the current value of xname = X [ ]? (y/n):  y
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
```

Immediately following this prompt is one asking for the character name associated with **VEC2** and again let's just respond in the affirmative to use the default.

The next prompts ask the user about smoothing the contour lines and the cuts of the VDF.

```
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|
     You will now be asked if you wish to smooth the contour and
cuts of the particle velocity distribution functions (DFs).  I
recommend smoothing the cuts but not the contours any more than the
minimum amount designated by SMOOTH.  The SM_CUTS keyword determines
if the routines smooth the cuts and the SM_CONT keyword determines
if the routines smooth the contours.  The NSMOOTH defines the number
of points to use in SMOOTH [Width parameter in routine] for each.
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Do you wish to use the default value of sm_cuts = FALSE [ ]? (y/n):  y
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>

 {the same informational text is shown again but I dropped it for brevity}

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Do you wish to use the default value of sm_cont = FALSE [ ]? (y/n):  y
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
```

The next prompts ask for the width parameter used by SMOOTH.PRO for the cuts and contours independently. These are also preceeded the same informational text as **SM_CUTS** and **SM_CONT**, but I will drop it for brevity:

```
 {the same informational text is shown again but I dropped it for brevity}
```

```
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Do you wish to keep the current value of nsmcut = 3 [ ]? (y/n):  y
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>

 {the same informational text is shown again but I dropped it for brevity}

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Do you wish to keep the current value of nsmcon = 3 [ ]? (y/n):  y
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
```

The next prompt asks the user which plane[21], defined by the orthonormal basis constructed from **VEC1** and **VEC2**, they wish to project contours of constant phase space density onto:

```
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|
     You will now be asked which plane you wish to enter project
the particle velocity distribution functions (DFs) onto.  If we
assume we have two input vectors, V1 and V2, then the options are:
  'xy'  :  horizontal axis parallel to V1 and plane normal to vector
             defined by (V1 x V2) [DEFAULT]
  'xz'  :  horizontal axis parallel to (V1 x V2) and vertical axis
             parallel to V1
  'yz'  :  horizontal axis defined by (V1 x V2) x V1 and vertical
             axis parallel to (V1 x V2)
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Enter 'xy', 'xz', or 'yz' for the desired plane of projection:  xy
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
```

The next three prompts ask the user to define the range of phase space densities to shown in the contour and cut plots. The first two ask about allowed limits and the third about the actual plot range.

```
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|
     You will now be asked whether you wish to enter a default
value for the lower and upper bound to be used in the phase
(velocity) space density plots.  The input format should be
(format = x.xxxESee), where x is any integer 0-9, E = e in IDL
[= x 10^(See)], S = sign of exponent, and ee = exponent values.

An example input would be:  1.05e-12
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Do you wish to keep the current value of dfmin = 1.000e-18 [s^(+3) cm^(-3) km^(-3)]? (y/n):  n
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Enter a new value for dfmin (format = x.xxxESee):  1e-13
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
```

[21] In nearly all publications, the plane chosen will correspond to the xy option.

```
 {verification check dropped it for brevity}
```

which is followed by:

```
 {the same informational text is shown again but I dropped it for brevity}

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Do you wish to keep the current value of dfmax = 1.000e-02 [s^(+3) cm^(-3) km^(-3)]? (y/n):  n
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Enter a new value for dfmin (format = x.xxxESee):  1e-3
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>

 {verification check dropped it for brevity}

 {the same informational text is shown again but I dropped it for brevity}

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Do you wish to use the default value of dfra = 1.000e-18 -- 1.000e-02 [s^(+3) cm^(-3) km^(-3)]? (y/n):  n
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Enter a new lower bound for dfra (format = x.xxxESee):  1e-10
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Enter a new upper bound for dfra (format = x.xxxESee):  1e-5
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
```

The last two initialization prompts ask about the location of the crosshairs (i.e., where cuts are performed) in the contour plots.

```
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|
You can enter a new value for v_0x (format = XXXX.xxx).
This will change the location for the perpendicular cut.

[Type 'q' to quit at any time]
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Do you wish to use the default value of v_0x = 0.00 [km/s]? (y/n):  y
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>

 {the same informational text is shown again but I dropped it for brevity}

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
Do you wish to use the default value of v_0y = 0.00 [km/s]? (y/n):  y
=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
```

5.2 Prompt Information: Changing Plotting Parameters

The program will plot chosen VDF and then the following input/output will be shown:

```
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|
[Type 'q' to leave this loop at any time]

Do you wish to change any of the plot ranges or the VSW estimate?
     [e.g. VLIM, DFMIN, DFMAX, DFRA, VSW, etc.]
-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|

=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>=>
To change any of these type 'y', otherwise type 'n':  y
<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=<=
```

At this point, I have chosen to change something. The routine input/output goes as follows:

```
Enter command (type '?' for help or 'q' to leave) :?

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
------------------------ POSSIBLE VBULK CHANGE COMMANDS ------------------------
q          quit [Enter at any prompt to exit]

zrange     change the DF range of the contour and cut plots
zfloor     change the Min. DF-value to allow in plots
zceil      change the Max. DF-value to allow in plots
vrange     change the velocity range of contour plots

plane      change the plane of projection in contour plots

nsmcut     change the number of points to smooth in the contour plots
nsmcon     change the number of points to smooth in the 1D cut plots
sm_cut     change the smooth [NSMCUT pts] setting for 1D cut plots
sm_con     change the smooth [NSMCON pts] setting for contour plots

vbulk      change the bulk flow velocity estimate
vec1       change the 'parallel' or 'X' vector for orthonormal coordinate basis
vec2       change the 2nd vector for constructing orthonormal coordinate basis
v_0x       change the origin of vertical cut line in contour plots
v_0y       change the origin of horizontal cut line in contour plots

next       switch to next particle distribution
prev       switch to previous particle distribution
index      choose an index for the particle distribution

save1      save the currently shown particle distribution plot
save3      save all three planes of the current particle distribution
--------------------------------------------------------------------------------
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

The user can then enter any one of these commands and follow the instructions/prompts until they are satisfied with the results, then move on to the next VDF or quit.

A Particle Data Structures in IDL

Full 3-dimensional particle distributions from the THEMIS ESA (or *Wind*/3DP) instruments come as data structures in the SPEDAS (*Wind*/3DP) software [*McFadden et al.*, 2008a]. The list of structure tags includes (but is not limited to), in no particular order:

I. **PROJECT_NAME** ≡ scalar [string] *e.g.* 'THEMIS-B'
II. **SPACECRAFT** ≡ scalar [string] *e.g.* 'b'
III. **DATA_NAME** ≡ scalar [string] *e.g.* 'IESA 3D burst'[22]
IV. **UNITS_NAME** ≡ scalar [string] *e.g.* 'counts'[23]
V. **UNITS_PROCEDURE** ≡ scalar [string] (*e.g.* 'thm_convert_esa_units') that tells conv_units.pro which IDL routine to use to convert the data units
VI. **TIME** ≡ scalar [double] defining the Unix[24] time associated with start of data sample (might be a slight delay from the sun pulse timestamp)
VII. **END_TIME** ≡ scalar [double] defining the Unix time associated with end of data sample
VIII. **DELTA_T** ≡ scalar [double] defining total duration of IDL structure [= END_TIME - TIME]
IX. **INTEG_T** ≡ scalar [double] defining the average time needed for the 1024 counter readouts per spin (s) [= (END_TIME - TIME)/1024]
X. **NENERGY** ≡ scalar [integer] defining the number of energy bins
XI. **NBINS** ≡ scalar [integer] defining the number of solid angle bins
XII. **DT_ARR** ≡ [NENERGY,NBINS]-element array [float] of anode accumulation times [unitless] per bin ⇒ accumulation time [s] of any given bin = (INTEG_T * DT_ARR)
XIII. **DATA** ≡ [NENERGY,NBINS]-element array [float] defining the data point values for each energy/angle bin [units depend on the value of UNITS_NAME]
XIV. **ENERGY** ≡ [NENERGY,NBINS]-element array [float] of average energy bin values [eV]
XV. **DENERGY** ≡ [NENERGY,NBINS]-element array [float] defining the energy range [eV] of each value of ENERGY
XVI. **PHI** ≡ [NENERGY,NBINS]-element array [float] defining the average azimuthal angle[25] [deg] for each bin
XVII. **DPHI** ≡ [NENERGY,NBINS]-element array [float] defining the angular range(uncertainty) [deg] for each value of PHI
XVIII. **THETA** ≡ [NENERGY,NBINS]-element array [float] defining the average poloidal angle[26] [deg] for each bin
XIX. **DTHETA** ≡ [NENERGY,NBINS]-element array [float] defining the angular range(uncertainty)[27] [deg] for each value of DTHETA
XX. **EFF** ≡ [NENERGY,NBINS]-element array [double] defining the efficiency correction [unitless] to the geometry factor accounting for dead time corrections
XXI. **GEOM_FACTOR** ≡ scalar [float] defining the total geometry factor of the detector [cm^2sr]
XXII. **GF** ≡ [NENERGY,NBINS]-element array [float] defining the relative geometric factor per bin ⇒ the geometry factor of each bin is = (GEOM_FACTOR * GF * EFF)
XXIII. **DEAD** ≡ scalar [float] defining the detector dead time [$\sim 170 \pm 10$ ns] of the Amptek A121 preamplifier[28]
XXIV. **CHARGE** ≡ scalar [float] defining the sign of the particle charge being measured
XXV. **MASS** ≡ scalar [float] defining the mass [$(eV/c)^2$ with c in km/s] of the particles being measured
XXVI. **MAGF** ≡ [3]-element array [float] defining the average magnetic field vector [nT] for the duration of the distribution (coordinate system depends on user preference but should match the basis defining PHI and THETA to be meaningful and useful)
XXVII. **VELOCITY** ≡ [3]-element array [double] defining the average bulk flow velocity [km/s] for the duration of the distribution (coordinate system issue similar to MAGF)[29]

[22] see dat_themis_esa_str_names.pro for more possibilities
[23] see thm_convert_esa_units_lbwiii.pro for descriptions and more possibilities
[24] seconds since January 1, 1970
[25] $+95° \lesssim \theta \lesssim +450°$ in DSL coordinates for THEMIS ESA and GSE coordinates for *Wind*/3DP, where +180° is roughly in the sun direction
[26] $-90° \leq \theta \leq +90°$, where 0° is roughly in the spin plane
[27] this is limited primarily by the anodes being used in a particular mode
[28] see, for example, *Paschmann and Daly* [1998] for explanation of dead times
[29] For *Wind*/3DP distributions, this tag will be **VSW**.

XXVIII. SC_POT ≡ scalar [float] defining the estimate of the spacecraft potential (eV)

B Unit Conversions

To convert between different units[30], a few quantities must be calculated first. Let us assume we start with the units of counts. Let us assume we have a particle distribution IDL data structure called *dat* (see Section A for structure tag definitions), then we can define the following quanties:

I. $E \equiv$ particle kinetic energy (eV) [associated with dat.ENERGY]
II. $N_E \equiv$ number of energy bins [associated with dat.NENERGY]
III. $N_A \equiv$ number of solid angle bins [associated with dat.NBINS]
IV. $\delta t \equiv$ sample/accumulation time (s) [associated with dat[0].INTEG_T[0]*dat.DT_ARR]
V. $gf \equiv$ differential geometry factor for each data point [associated with dat.GF * dat.GEOM_FACTOR * dat.EFF]
VI. $M_s \equiv$ particle mass of species s ((eV/c)^2 with c in km/s) [associated with dat.MASS]
VII. $\tau \equiv$ dead time [associated with dat.DEAD]
VIII. *f(E,Ω)* $\equiv$ the data [associated with dat.DATA in counts], where Ω is the solid angle
IX. *g(E,Ω)* $\equiv$ the data in new user specified units
X. $\delta g \equiv$ estimated uncertainty in *g(E,Ω)*

To correct for the dead time, we define:

$$\delta t_c \equiv \frac{\tau f(E,\Omega)}{\delta t} \tag{3}$$

The scale factors used to convert from counts to any of the following are:

$$Counts: scale = 1.0 \tag{4a}$$
$$rate: scale = (\delta t)^{-1} \tag{4b}$$
$$crate: scale = (\delta t)^{-1} \tag{4c}$$
$$eflux: scale = (\delta t * gf)^{-1} \tag{4d}$$
$$flux: scale = (\delta t * gf * E)^{-1} \tag{4e}$$
$$df: scale = (\delta t * gf * E)^{-1} * \left(\frac{mass^2}{2.0\times10^5}\right) \tag{4f}$$

where the final result in new units is given by:

$$g(E,\Omega) = scale * \left(\frac{f(E,\Omega)}{\delta t_c}\right) \ . \tag{5}$$

The uncertainty in *g(E,Ω)* is given by:

$$\delta g = scale * \left(\frac{f(E,\Omega)}{\delta t_c}\right)^{1/2} \ . \tag{6}$$

C Bi-Maxwellian Distribution Functions

In general, for uncorrelated velocity variables, we can write:

$$f\left(V_x, V_y, V_z\right) = f\left(V_x\right) f\left(V_y\right) f\left(V_z\right) \ . \tag{7}$$

Note that a generalized Gaussian probability density function is given by:

$$f\left(x\right) = \frac{A_o}{\sqrt{2\pi\sigma^2}}\, e^{-\frac{(x-x_o)^2}{2\sigma^2}} \tag{8}$$

where x_o is the displacement of the peak from x = 0, A_o is a normalization amplitude, and σ^2 is the variance (defined in Equation 24). For this distribution, one can find the Full Width at Half Maximum (FWHM) =

[30] see, for example, thm_convert_esa_units_lbwiii.pro

$2\sqrt{2\ln 2}\sigma$, or the width of the distribution at half its peak value. In terms of physical parameters, FWHM $= 2\sqrt{\ln 2}\ \mathrm{V}_{T_s}$, where V_{T_s} is the *most probable speed*, which we will use as the thermal speed. The most probable speed is given by:

$$V_{T_s} = \sqrt{\frac{2k_B T_s}{m_s}} \tag{9}$$

where k_B is Boltzmann's constant, T_s is the temperature, m_s is the mass, and s is the particle species. Note that all thermal speeds discussed herein will be defined by Equation 9. Now if we change Equation 8 by letting $2\sigma^2 \rightarrow \mathrm{V}_{T_s}{}^2$, $\mathrm{x}_o \rightarrow \mathrm{v}_o$[31], and $\mathrm{A}_o \rightarrow \mathrm{n}_o$[32], then a one-dimensional Maxwell-Boltzmann velocity distribution, or Maxwellian, is given by:

$$f_s(v) = \frac{n_o}{\sqrt{\pi}\, V_{T_s}}\, e^{-\left(\frac{v - v_o}{V_{T_s}}\right)^2}. \tag{10}$$

Now we can take Equation 7 and let $\mathrm{V}_x \rightarrow \mathrm{V}_\perp \cos\phi$, $\mathrm{V}_y \rightarrow \mathrm{V}_\perp \sin\phi$, and $\mathrm{V}_z \rightarrow \mathrm{V}_\parallel$, where ϕ is the phase angle of the velocity. If we assume azimuthal symmetry[33], then the distribution is said to be *gyrotropic*. This leads to $\mathrm{V}_{T\perp,x} = \mathrm{V}_{T\perp,y} \equiv \mathrm{V}_{T\perp}$, which gives us the general form of a bi-Maxwellian give by:

$$f(V_\parallel, V_\perp) = \frac{n_o}{\pi^{3/2} V_{T\perp}{}^2 V_{T\parallel}}\, e^{-\left[\left(\frac{V_\parallel - V_{o\parallel}}{V_{T\parallel}}\right)^2 + \left(\frac{V_\perp - V_{o\perp}}{V_{T\perp}}\right)^2\right]} \tag{11}$$

where the subscripts $\perp(\parallel)$ are the perpendicular(parallel) directions with respect to a quasi-static background magnetic field.

D Fluid Moment Definitions

Let us assume we have a function, $\mathrm{f}_s(\mathbf{x},\mathbf{v},\mathrm{t})$, which defines the number of particles of species s in the following way:

$$dN = f_s(\mathbf{x},\mathbf{v},t)\ d^3x\, d^3v \tag{12}$$

which tells us that $\mathrm{f}_s(\mathbf{x},\mathbf{v},\mathrm{t})$ is the particle distribution function of species s that defines a probability density in phase space. We can define moments of the distribution function as expectation values of any dynamical function, $\mathrm{g}(\mathbf{x},\mathbf{v})$, as:

$$\langle g(\mathbf{x},\mathbf{v})\rangle = \frac{1}{N}\int d^3x\, d^3v\ g(\mathbf{x},\mathbf{v})\ f(\mathbf{x},\mathbf{v},t) \tag{13}$$

where $\langle\ \rangle$ is the average, which can mean ensemble average, arithmetic mean, etc.

If we define a set of fluid moments with similar format to that of Equation 30, which act as averages,

[31] drift speed
[32] particle number density
[33] $\partial \mathrm{f}/\partial\phi = 0$

then we have:

$$\text{number density: } n_s = \int d^3v\ f_s\left(\mathbf{x},\mathbf{v},t\right) \tag{14a}$$

$$\text{average velocity: } \mathbf{U}_s = \frac{1}{n_s}\int d^3v\ \mathbf{v}\ f_s\left(\mathbf{x},\mathbf{v},t\right) \tag{14b}$$

$$\text{kinetic energy density: } W_s = \frac{m_s}{2}\int d^3v\ v^2\ f_s\left(\mathbf{x},\mathbf{v},t\right) \tag{14c}$$

$$\text{pressure tensor (dyadic): } \overleftrightarrow{\mathbb{P}}_s = m_s\int d^3v\ \mathbf{ww}\ f_s\left(\mathbf{x},\mathbf{v},t\right) \tag{14d}$$

$$\text{heat flux tensor (triadic): } Q_{l,m,n} = m_s\int d^3v\ \mathbf{w}_l\mathbf{w}_m\mathbf{w}_n\ f_s\left(\mathbf{x},\mathbf{v},t\right) \tag{14e}$$

where $\mathbf{w} = (\mathbf{v}$ - $\mathbf{U}_s)$ and the pressure tensor can be written as:

$$\overleftrightarrow{\mathbb{P}}_s = \begin{bmatrix} P_{xx} & P_{xy} & P_{xz} \\ P_{yx} & P_{yy} & P_{yz} \\ P_{zx} & P_{zy} & P_{zz} \end{bmatrix} \tag{15}$$

which can be reduced to a symmetric tensor in most cases with the only off-diagonal elements being P_{xy}, P_{xz}, and P_{yz}. In a magnetized plasma, the magnetic field direction can often organize the collective particle motion so that the pressure tensor is reduced to a diagonal tensor[34]. In general, one can separate the pressure tensor into a diagonal part and the off-diagonal part, which is usually called the stress tensor. The general diagonal elements of the pressure tensor are:

$$\overleftrightarrow{\mathbb{P}}_s = \begin{bmatrix} P_{\perp,1} & 0 & 0 \\ 0 & P_{\perp,2} & 0 \\ 0 & 0 & P_{\parallel} \end{bmatrix} \tag{16}$$

where a gyrotropic assumption will result in $P_{\perp,1} = P_{\perp,2}$. Thus, a gyrotropic plasma will have:

$$P_{\perp,s} = n_s k_B T_{\perp,s} \tag{17a}$$

$$P_{\parallel,s} = n_s k_B T_{\parallel,s} \tag{17b}$$

and a non-gyrotropic plasma will have:

$$T_{\perp,s} = \frac{1}{2n_s k_B}\left(P_{\perp,1,s} + P_{\perp,2,s}\right) \tag{18a}$$

$$T_{\parallel,s} = \frac{1}{n_s k_B}P_{\parallel,s}\ . \tag{18b}$$

[34]Off-diagonal terms arise from non-gyrotropic features e.g. gyrating [*e.g. Meziane et al.*, 1997] or gyrophase-bunched [*e.g. Gurgiolo et al.*, 1981] ion distributions.

Therefore, if we have the following relationships:

$$\text{gyrotropic: } V_{T_s} = \sqrt{\frac{1}{2}\left(V_{T_s,\perp}{}^2 + V_{T_s,\parallel}{}^2\right)} \tag{19a}$$

$$\text{non-gyrotropic: } V_{T_s} = \sqrt{\frac{2}{3m_s} Tr\left[\frac{\overleftrightarrow{\mathbb{P}}_s}{n_s k_B}\right]} \tag{19b}$$

$$= \sqrt{\frac{2k_B \langle T_s \rangle}{m_s}} \tag{19c}$$

where we have used Tr[] as the trace and defined:

$$\langle T_s \rangle = \frac{1}{3} Tr\left[\frac{\overleftrightarrow{\mathbb{P}}_s}{n_s k_B}\right] \ . \tag{20}$$

The average temperature of particle species s shown in Equation 20 is the one most often used when calculating temperatures from electrostatic plasma analyzers [*e.g. Curtis et al.*, 1989]. The temperature is physically a measure of the average kinetic energy density of particle species s, and can be represented as:

$$T_{\perp,s} = \frac{1}{2}\left(T_{\perp,1,s} + T_{\perp,2,s}\right) \tag{21a}$$

$$\langle T_s \rangle = \frac{1}{3}\left(T_{\perp,1,s} + T_{\perp,2,s} + T_{\parallel,s}\right) \tag{21b}$$

therefore, if we already have $V_{T_s,\perp}$ and $V_{T_s,\parallel}$ and we assume $T_{\perp,1} \neq T_{\perp,2}$ (*i.e.* non-gyrotropic)[35], then we have:

$$V_{T_s} = \sqrt{\frac{1}{3}\left(V_{T_s,\perp,1}{}^2 + V_{T_s,\perp,2}{}^2 + V_{T_s,\parallel}{}^2\right)} \tag{22a}$$

$$= \sqrt{\frac{2V_{T_s,\perp}{}^2}{3} + \frac{V_{T_s,\parallel}{}^2}{3}} \tag{22b}$$

$$\neq \sqrt{\frac{1}{2}\left(V_{T_s,\perp}{}^2 + V_{T_s,\parallel}{}^2\right)} \tag{22c}$$

E Statistics Definitions

We define $\bar{x}$ as the arithmetic mean of a set of {x_1, x_2, ..., x_N}, given by:

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{23}$$

and this allows us to define the variance of {x_1, x_2, ..., x_N} as:

$$\sigma^2 = \frac{1}{N-1}\sum_{i=1}^{N}\left(x_i - \bar{x}\right)^2 \tag{24}$$

where N - 1 = number of degrees of freedom and σ is the standard deviation. Sometimes $\bar{x}$ is written as $\langle x \rangle$.

[35]In most cases, it is assumed that the electrons are gyrotropic and ions are non-gyrotropic. Physically, this is due to the relatively long sample period ($\geq$3 s) of current particle detectors compared to ${\Omega_{ce}}^{-1}$ for electrons, which causes the resulting measured distribution to appeared *smeared out* in phase space. Most non-gyrotropic features are lost due to the relatively long sample periods. For ions, however, ${\Omega_{cp}}^{-1}$ can be $\sim$1-10 s (for $B_o \sim$ 1-10 nT). Therefore, non-gyrotropic features (*e.g.* gyrophase bunching) can often be observed in ion distributions.

If there exists a probability density function, P(x), then for discrete (Equation 25a) or continuous (Equation 25b) values we have:

$$\langle x \rangle = \sum_{i=1}^{N} P(x_i) f(x_i) \tag{25a}$$

$$= \int dx\ P(x) f(x) \tag{25b}$$

$$\equiv \mu_x\ . \tag{25c}$$

Note that $\langle\ \rangle$ represents the arithmetic mean. Similarly, the variance can be expressed as:

$$\sigma^2 = \sum_{i=1}^{N} P(x_i) (x_i - \mu)^2 \tag{26a}$$

$$= \int dx\ P(x) (x - \mu)^2 \tag{26b}$$

where we can write a variance operator as follows:

$$var(x) \equiv \langle x^2 \rangle - \langle x \rangle^2 \tag{27}$$

and define the covariance as:

$$cov(x, y) \equiv \langle (x - \mu_x)(y - \mu_y) \rangle \tag{28}$$

which can also be written as:

$$\mathcal{V}_{xy} = \sigma_{xx}\sigma_{yx} + \sigma_{xy}\sigma_{yy} \tag{29}$$

where we have defined $\sigma_{jk} \equiv$ cov(x_j,x_k) and $\sigma_{jj} = \sigma_j{}^2$.

The moments of of the distribution can be defined as $\mu^n \equiv \langle$(x - $\langle$x$\rangle)^n\rangle$, which for a general Gaussian distribution takes the form:

$$\mu^n = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{-\infty}^{\infty} dx\ (x - \mu)^n\ e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{30}$$

References

Angelopoulos, V. (2008), The THEMIS Mission, *Space Sci. Rev.*, *141*, 5–34, doi:10.1007/s11214-008-9336-1.

Curtis, D. W., C. W. Carlson, R. P. Lin, G. Paschmann, and H. Reme (1989), On-board data analysis techniques for space plasma particle instruments, *Rev. Sci. Instr.*, *60*, 372–380, doi:10.1063/1.1140441.

Gosling, J. T., M. F. Thomsen, S. J. Bame, W. C. Feldman, G. Paschmann, and N. Sckopke (1982), Evidence for specularly reflected ions upstream from the quasi-parallel bow shock, *Geophys. Res. Lett.*, *9*, 1333–1336, doi:10.1029/GL009i012p01333.

Gurgiolo, C., G. K. Parks, B. H. Mauk, K. A. Anderson, R. P. Lin, H. Reme, and C. S. Lin (1981), Non-E x B ordered ion beams upstream of the earth's bow shock, *J. Geophys. Res.*, *86*, 4415–4424, doi:10.1029/JA086iA06p04415.

Lin, R. P., et al. (1995), A Three-Dimensional Plasma and Energetic Particle Investigation for the Wind Spacecraft, *Space Sci. Rev.*, *71*, 125–153, doi:10.1007/BF00751328.

McFadden, J. P., C. W. Carlson, D. Larson, M. Ludlam, R. Abiad, B. Elliott, P. Turin, M. Marckwordt, and V. Angelopoulos (2008a), The THEMIS ESA Plasma Instrument and In-flight Calibration, *Space Sci. Rev.*, *141*, 277–302, doi:10.1007/s11214-008-9440-2.

McFadden, J. P., C. W. Carlson, D. Larson, J. Bonnell, F. Mozer, V. Angelopoulos, K.-H. Glassmeier, and U. Auster (2008b), THEMIS ESA First Science Results and Performance Issues, *Space Sci. Rev.*, *141*, 477–508, doi:10.1007/s11214-008-9433-1.

Meziane, K., et al. (1997), Wind observation of gyrating-like ion distributions and low frequency waves upstream from the earth's bow shock, *Adv. Space Res.*, *20*, 703–706, doi:10.1016/S0273-1177(97)00459-6.

Paschmann, G., and P. W. Daly (1998), Analysis Methods for Multi-Spacecraft Data. ISSI Scientific Reports Series SR-001, ESA/ISSI, Vol. 1. ISBN 1608-280X, 1998, *ISSI Sci. Rep. Ser.*, *1*.

Wilson III, L. B., C. A. Cattell, P. J. Kellogg, K. Goetz, K. Kersten, J. C. Kasper, A. Szabo, and K. Meziane (2009), Low-frequency whistler waves and shocklets observed at quasi-perpendicular interplanetary shocks, *J. Geophys. Res.*, *114*, A10106, doi:10.1029/2009JA014376.

Wilson III, L. B., C. A. Cattell, P. J. Kellogg, K. Goetz, K. Kersten, J. C. Kasper, A. Szabo, and M. Wilber (2010), Large-amplitude electrostatic waves observed at a supercritical interplanetary shock, *J. Geophys. Res.*, *115*, A12104, doi:10.1029/2010JA015332.

Wilson III, L. B., et al. (2012), Observations of electromagnetic whistler precursors at supercritical interplanetary shocks, *Geophys. Res. Lett.*, *39*, L08109, doi:10.1029/2012GL051581.

Wilson III, L. B., et al. (2013a), Electromagnetic waves and electron anisotropies downstream of supercritical interplanetary shocks, *J. Geophys. Res.*, *118*(1), 5–16, doi:10.1029/2012JA018167.

Wilson III, L. B., et al. (2013b), Shocklets, SLAMS, and field-aligned ion beams in the terrestrial foreshock, *J. Geophys. Res.*, *118*(3), 957–966, doi:10.1029/2012JA018186.

Wilson III, L. B., D. G. Sibeck, A. W. Breneman, O. Le Contel, C. Cully, D. L. Turner, V. Angelopoulos, and D. M. Malaspina (2014a), Quantified Energy Dissipation Rates in the Terrestrial Bow Shock: 1. Analysis Techniques and Methodology, *J. Geophys. Res.*, *119*(8), 6455–6474, doi:10.1002/2014JA019929.

Wilson III, L. B., D. G. Sibeck, A. W. Breneman, O. Le Contel, C. Cully, D. L. Turner, V. Angelopoulos, and D. M. Malaspina (2014b), Quantified Energy Dissipation Rates in the Terrestrial Bow Shock: 2. Waves and Dissipation, *J. Geophys. Res.*, *119*(8), 6475–6495, doi:10.1002/2014JA019930.