A 3-COMPONENT LASER-DOPPLER VELOCIMETER 
DATA ACQUISITION AND REDUCTION SYSTEM

BY

L. C. Rodman, J. H. Bell and R. D. Mehta

Stanford University 
Department of Aeronautics and Astronautics 
Stanford, CA 94305 

AUGUST 1985
A 3-COMPONENT LASER-DOPPLER VELOCIMETER DATA ACQUISITION AND REDUCTION SYSTEM

BY

L. C. Rodman, J. H. Bell and R. D. Mehta

The work here presented has been supported by NASA Grant NCC 2-294.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>iii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>iv</td>
</tr>
<tr>
<td>NOMENCLATURE</td>
<td>v</td>
</tr>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2. OPTICAL SYSTEM AND SIGNAL PROCESSING HARDWARE</td>
<td>3</td>
</tr>
<tr>
<td>2.1 Optics Table</td>
<td>4</td>
</tr>
<tr>
<td>2.2 Transmitting Optics</td>
<td>5</td>
</tr>
<tr>
<td>2.3 Receiving Optics</td>
<td>6</td>
</tr>
<tr>
<td>2.4 Signal Processing Hardware</td>
<td>6</td>
</tr>
<tr>
<td>2.5 Computer Interface</td>
<td>7</td>
</tr>
<tr>
<td>2.6 HP 9845A Desk-Top Computer</td>
<td>8</td>
</tr>
<tr>
<td>3. DATA ACQUISITION AND REDUCTION SOFTWARE</td>
<td>8</td>
</tr>
<tr>
<td>3.1 Data Acquisition Software</td>
<td>9</td>
</tr>
<tr>
<td>3.2 Data Reduction Software</td>
<td>10</td>
</tr>
<tr>
<td>4. OPERATING PROCEDURES</td>
<td>15</td>
</tr>
<tr>
<td>4.1 Alignment Procedures</td>
<td>15</td>
</tr>
<tr>
<td>4.2 Signal Processing</td>
<td>17</td>
</tr>
<tr>
<td>5. INHERENT PROBLEMS</td>
<td>19</td>
</tr>
<tr>
<td>6. SAMPLE RESULTS</td>
<td>21</td>
</tr>
<tr>
<td>7. CONCLUDING REMARKS</td>
<td>22</td>
</tr>
<tr>
<td>APPENDIX -- SOFTWARE FOR THE HP 9845B COMPUTER</td>
<td>23</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>67</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>68</td>
</tr>
<tr>
<td>TABLE 1</td>
<td>69</td>
</tr>
<tr>
<td>FIGURES</td>
<td>70</td>
</tr>
</tbody>
</table>
ABSTRACT

This report describes a Laser Doppler Velocimeter capable of measuring all three components of velocity simultaneously in low-speed flows. All the mean velocities, Reynolds stresses, and higher-order products can then be evaluated. The approach followed is to split one of the two colors used in a 2-D system, thus creating a third set of beams which is then focused in the flow from an off-axis direction. The third velocity component is computed from the known geometry of the system. In this report, the laser optical hardware and the data acquisition electronics are described in detail. In addition, full operating procedures and listings of the software (written in BASIC and ASSEMBLY languages) are also included. Some typical measurements obtained with this system in a vortex/mixing layer interaction are presented and compared directly to those obtained with a cross-wire system. A brief description of the present system together with a review of existing 3-D Laser Doppler Velocimeters is given in Ref. 1.
LIST OF FIGURES

Fig. 1 Schematic of the optics table layout.
Fig. 2 Schematic of the transmitting optics.
Fig. 3 Details of the probe volumes in the 3-D system.
Fig. 4 Schematic of the receiving optics.
Fig. 5 Evaluation of the effective probe length.
Fig. 6 Schematic of the signal processing electronics.
Fig. 7 Computer interface simplified block diagram.
Fig. 8 Block diagram for data acquisition program.
Fig. 9 Block diagram for data reduction program.
Fig. 10 NASA LDV-A/D computer interface connections and settings.
Fig. 11 Experimental rig.
   (a) Overall schematic
   (b) Details of boundary layer trips and coordinate system
Fig. 12 Secondary velocity plots.
   (a) Cross-wire measurements
   (b) LDV measurements
Fig. 13 Comparison of X-wire and LDV measurements in a vortex/mixing layer interaction.
NOMENCLATURE

a(i)        ith sample value
\(d_f\)     fringe spacing
f           frequency
i           sample number
Ns          number of samples
R           range setting on Macrodyne processor
\(S\langle a(i)\rangle\) sum of \(a(i)\) from \(i = 1\) to \(i = Ns\)
S\(a\)       average of quantity \(a\), defined by equations (3)
\(\bar{u}, \bar{v}, \bar{w}\) mean velocity components in \(x,y,z\) directions, respectively
\(u,v,w\)   instantaneous velocity components in \(x,y,z\) directions, respectively
\(U_o\)     \(U_1 - U_2\), velocity difference between the two streams in the mixing layer
\(x,y,z\)   Streamwise, normal, and spanwise coordinate directions, respectively

Subscripts:

b\(\text{ragg}\)  Bragg
m\(\text{ix}\)     mixing
d\(\text{opp}\)    doppler
r\(\text{es}\)     resultant
1            quantity measured in high-speed side of mixing layer
2            quantity measured in low-speed side of mixing layer

Superscripts:

'         fluctuating quantity, e.g., \(u = \bar{u} + u'\)
-         time average
1. INTRODUCTION

Our ability to understand and model turbulent flows still relies heavily on the availability of accurate measurements of mean and fluctuating quantities within the flow. Until recently, the hot wire was the only reliable tool available for the measurement of fluctuating velocities. In fact, almost all of our present knowledge about turbulent flows is based on measurements made with hot wires. In relatively simple flows (moderately two-dimensional with small cross-flows), reliable and accurate hot wire measurements are now possible with fully automated data acquisition and reduction systems which minimize errors due to drifts in calibrations. An example of such a system is given in Ref. 2.

However, as we turn our attention towards more complex turbulent flows, a need for more sophisticated measurement techniques has become apparent. These complex flows include those with compressibility effects, strong three-dimensionality (with steep mean gradients), flow reversals, and time-dependent behavior. Since about the mid-sixties, the most popular alternative tool for measuring mean and fluctuating velocities in turbulent flows has been the Laser Doppler Velocimeter (LDV).

The most popular LDV arrangement used for wind tunnel measurements is the dual beam or fringe method. In this method, one of the laser lines is split into two lines of equal intensity which are then focused through a lens so that they cross over at the focal point. The flow is seeded with small particles (typically less than 3-4 μm in diameter) which follow the fluid motion. As these particles pass through interferometric fringes created by the crossed laser beams, light is scattered off them which is received by a photodetector. The frequency of this scattered light, along with a knowledge of the fringe pattern formed by the laser beams, provide the means to calculate the velocity of the particle. The fringe method, especially in
the forward scatter, off-axis mode, generally offers the best signal-to-noise ratios and spatial resolution.

Although LDV systems are somewhat complex and tiresome to set up, they have certain advantages over hot wires for turbulence measurements. The fact that Laser Doppler Velocimetry is non-intrusive is especially beneficial in the measurement of unstable flow phenomena which are very sensitive to the presence of measurement probes. In certain situations, LDV systems can also provide greater spatial resolution and better directional discrimination than hot wires. This makes it possible to use LDV systems for the measurement of separated flows. Since an LDV measures the velocity directly, independent of the thermodynamic properties of the flow, it is particularly attractive for velocity measurements in compressible flows. Furthermore, the calibration converting the frequency to velocity is linear and easy to implement in software. This feature also allows for uniform sensitivity in measuring both moderate and high turbulence intensities.

Two-color LDV systems capable of measuring two components of velocity simultaneously are now being widely used. However, the main interest in the present investigation was to study three-dimensional interactions where it is desirable to obtain measurements of all three velocities. Hence, the first objective was to develop a laser velocimeter system capable of measuring all three components of velocity simultaneously so that all six components of Reynolds stress may be computed. Another objective was to compare these LDV measurements directly with those obtained with hot wires in flow fields where both techniques are expected to perform satisfactorily. The purpose of this is to evaluate the performance of the system quantitatively and objectively.

The approach followed in the present investigation was to convert an existing two-component LDV system into one capable of measuring all three components of velocity simultaneously. The system utilizes two wavelengths (488.0 and 514.5 nm) from a 4-watt Argon-Ion laser. The main four-beam matrix measures
u and v directly. The green line in the four-beam matrix is split (in half) using a dichroic filter and directed over the top of the traverse mechanism with mirrors, giving the third beam pair for the measurement of the w-component. This pair of beams (with rotated polarization) measures \( w \sin 45^\circ + v \cos 45^\circ \).

Scattered light is collected in the off-axis forward scatter mode using two collection lenses.

Signal processing is accomplished with single-particle burst counters, and the validated data are multiplexed through a "home-built" interface to an HP 9845B desk-top computer. Some selected first and second order products are reduced on-line, and the raw data is dumped onto floppy disk. An off-line program reduces the data, giving up to third order quantities and also plots histograms of the raw data for each channel. The software includes the capability to filter out noise by examining the histograms.

The optical and signal processing hardware is described in Section 2. The data acquisition and reduction software is described in Section 3 and detailed operating procedures are given in Section 4. Some problems, inherent to 3-component LDV systems, are presented in Section 5. Sample results from an experiment measuring mean and turbulence quantities in a vortex/mixing layer interaction are compared directly to results obtained using crossed hot-wire anemometry in Section 6, and concluding remarks are presented in the final section. Complete software listings written in BASIC and ASSEMBLY languages to run on the HP 9845B desk-top computer are included in the appendix.

2. OPTICAL SYSTEM AND SIGNAL PROCESSING HARDWARE

The hardware for the 3-component LDV system can be divided into three categories: the optical system, the signal processing instrumentation, and the computer. The LDV optics consist of the optics table, where the laser beam is split into green and blue beam pairs, the transmitting optics, where the beams are
directed into the flow field, and the receiving optics, where the scattered light is picked up by photodetectors. The signal processing instrumentation consists of amplifiers, filters, burst counters and a computer interface. The sampling procedures are all computer controlled.

2.1 Optics Table

The optics table consists of the laser and all the optical elements needed to provide the necessary four-beam matrix. Fig. 1 shows a schematic of the arrangement with component numbers as referred to in this section. A 4-watt Argon-Ion laser (Lexel Model 95) is used to produce the main beam. The beam then passes through a collimator (1), which ensures that the beam waist occurs at the focal point of the transmitting lens.

The collimated beam is passed through a color separator box, which consists of a polarization rotator (2), an attenuator (3), and a pair of high dispersion Brewster angle prisms (4) which are used to separate the multi-line beam into two colors, blue (488 nm) and green (514.5 nm). These two beams are then reflected across the box by mirror (5), and out of the box by mirrors (6, 7).

Following the color separation, the green beam's polarity is rotated to horizontal (8), and the beam is split into two beams in the vertical plane (9). Most beam splitters prefer this type of perpendicular polarization for maximum efficiency. Using the beam displacer (10), the blue beam is then moved to the center of the optics, and its polarity is rotated to vertical (11). The blue beams are split in the horizontal plane (12). At this point, the four beams are each displaced 25 mm from the optical axis.

Two Bragg cells (13, 14) are used to shift the frequency of one beam from each pair. The unshifted beam passes through an optical rod so that the path lengths are matched. The frequency is shifted by a fixed amount of 40 Mhz. This shift creates
a moving system of fringes at the beam intersection point, allowing for directional discrimination of the velocity. Frequency shifting also helps to reduce the percentage of frequency change in highly turbulent flow, to reduce fringe bias, and to optimize frequency, thus enabling easy removal of the pedestal by high pass filtering.

The four beams are then passed through a beam steering module (15). The module consists of a set of wedge prisms that can be independently rotated about the beam axis to steer the shifted beam in any direction. This allows a more precise alignment of the beams. The beams are finally passed through a beam displacer (16) to reduce the beam spacing to 13 mm and a rotating prism (17) before leaving the optics table. The rotating prism enables the four-beam matrix to be rotated independently so that the beams may be aligned relative to the tunnel axes.

2.2 Transmitting Optics

The transmitting optics (Fig. 2) are mounted on a traversing mechanism with three degrees of freedom. The traverses are driven by individual stepper motors. The four beams from the optics table are directed by a set of five mirrors through a dichroic filter before being focused by a 380 mm (15 inch) focal length lens. This main four-beam matrix measures u and v directly. The dichroic filter, set at an angle of about 30 degrees to the incoming beams, splits the green beams in half, which provides the third beam pair for the third velocity component, w. This third beam pair is directed over the top by mirrors, passed through a polarization rotator (giving it a different polarity than the main-axis green beam pair) and then focused at the focal point of the main beam set. Since this third beam pair intersects the main measuring volume at a 45° angle to the main axis, it measures v and w with equal sensitivity, with the measured component being \( w_v = (v \cos 45^\circ + w \sin 45^\circ) \). Since v is measured directly, w can be evaluated using the equation:
Typical probe volume dimensions for each beam pair in the present configuration are 10 mm in length and 0.2 mm in diameter (Fig. 3). However, the actual "viewed" dimensions are reduced considerably, as discussed below, in Section 2.3.

2.3 Receiving Optics

The detector system (Fig. 4) is in the off-axis forward scatter mode. The receiving optics are mounted on a traversing gear, also run by stepper motors, which moves synchronously with the transmitting optics. Scattered light is collected by two 380 mm (15 inch) focal length lenses. The collimated light is passed through filters to separate the colors and the off-axis line is additionally passed through a polarization filter to avoid collecting light scattered by the main axis green pair. The collected light is then focused by 250 mm (10 inch) focal length lenses onto pin-hole apertures mounted in front of the three photomultiplier tubes. The collection angle and diameter can be adjusted to select the effective (viewed) probe length (Fig. 5). In the present set-up, a collection angle of 30 degrees and an aperture diameter of 0.5 mm were used to give an effective length of about 1.5 mm.

2.4 Signal Processing Hardware

The signals from the photomultiplier tubes are amplified and relayed to the signal processors via high-pass filters and mixers (Fig. 6). The amplifiers used are EIN model 403LA with a fixed gain of 37 dB and the filters are Allen Avionics F2440 with a fixed high-pass cutoff of 10 MHz. The filtered signals are mixed electronically with sine waves from three Tektronix SG503 Levelled Sine Wave Generators. The mixer is commercially available from Hewlett-Packard, model 10534A. The mixing procedure
is necessary for low-speed flows, where the actual Doppler frequencies are small compared to the Bragg frequency of 40 MHz. So in order to reduce the effective measured frequency and hence improve the counter resolution, the incoming signals are mixed with sine waves of known frequency.

The mixed signal is fed into single particle burst counters (Macodyne model 2096-2 and 3003) via high-pass/low-pass filters and an amplifier (x10). The counters measure the zero crossings of the Doppler signal, which is related to the Doppler frequency by the range set on the Macodyne. The range is set manually based on expected flow velocities, since it limits the frequencies that the processor can see for the given 10 bits of resolution.

The processors use two checks to validate a Doppler signal. The first check is the usual $5/8$ comparison, where the processor checks the frequency for 5 zero crossings against that for 8 crossings. The second check is the multi-sequence check. Positive and negative thresholds are set on the signal, and a validated output is permitted only if, for all eight fringe crossings, the signal passes through a positive threshold, a zero level, and a negative threshold in the proper sequence. The digital data (consisting of a 10 bit data word with 3 bits giving the range) and a sync pulse (produced every time the front end of a valid burst is detected) are passed to the computer interface.

2.5 Computer Interface

A NASA LDV-A/D computer interface (CI) is used to transfer data from the LDV signal processor to the computer (Fig. 7). The CI can interface either digital or mixed analog and digital data to an HP 9845B desk-top computer. The CI consists of an eight-channel multiplexer, a four-channel A/D converter, and an event synchronizer with time interval counter.

For use with the LDV, the inputs to the CI are all digital. Six of the eight words come from the processor, and two are time and status words from the synchronizer. The inputs are
multiplexed to a single digital data channel output.

The CI can accept data from the three processors in either random mode or sync mode. In random mode, the CI will accept data inputs when an event occurs on any of the three channels. A dead time (between 5 and 50 μs) is set in this mode, which controls the minimum time between samples to ensure that a given particle is sampled only once. In sync mode, the CI will accept data inputs when all three processors sample simultaneous events. In this mode, a coincidence time (between 5 and 50 μs) has to be set, which determines the time window within which all three events must occur. A detailed description of the CI can be found in Ref. 5.

2.6 HP 9845B Desk-top Computer

Multiplexed data are passed to the HP computer from the NASA LDV-A/D using the HP 98032A high-speed 16-bit parallel interface. Jumper labeled "9,B,D" are connected inside the 98032A for proper operation with the LDV-A/D. A select code of 10 (screw setting on the 98032A) is set for use with the software described in the appendix. A data buffer of 24 kbytes is provided in the memory of the HP 9845B desk-top computer for storage of up to 2000 samples obtained from one measurement location. Since three of the six data words passed for each sample are merely monitored and discarded as described above, 12 kbytes of raw data remain to be stored for each point. An HP 9895A floppy-disk drive is used for archival storage of raw data. The buffered data are written in real time to a sequential-access floppy-disk file. Enough header information is written to each file to identify the run, and to reproduce calibration tables.

3. DATA ACQUISITION AND REDUCTION SOFTWARE

Data acquisition and reduction on the HP 9845B is done via
two programs. The data acquisition program controls the processor sampling. The program accepts heading and initialization parameters provided by the operator, performs a fast I/O handshake to acquire the raw data from the LDV-A/D computer interface, and writes the raw data and initialization parameters onto floppy-disk. In addition, the data acquisition program has a limited capability for on-line data reduction so that key results may be monitored during a run. The off-line data reduction program converts the raw data into instantaneous velocities and computes all the statistical quantities. The off-line program also has options to plot histograms of the raw data, to plot profiles of the reduced data, and to filter out noise.

3.1 Data Acquisition Program

The data acquisition program structure can be divided into four areas: (1) initialization of variables, (2) data acquisition, (3) limited data reduction, and (4) raw data storage (see the block diagram in Fig. 8). During the initialization stage of the program, relevant test parameters are input to be saved along with the data. These parameters include the spatial coordinates, Bragg and mixing frequencies, and angle of the third beam pair. Fringe spacings and probe dimensions are also calculated at this time.

Data is acquired from the LDV computer interface by means of a fast I/O handshake. The operator requests that a certain number of data samples be taken and stored in the data buffer as described in Section 2.6. The 16-bit data words are put into an integer array, three words wide, containing values for the three velocity components. The number acquired from the counter occupies only bits 0-9 of the data word. Bits 10-13 specify the range, and bits 14 and 15 are unused. The data array, along with the initialization parameters given by the operator, are stored in a floppy-disk file. As an option, some of the data can be reduced on-line, before the raw data is written
to floppy-disk. This on-line reduction feature uses the same algorithms as the off-line data reduction program, but is faster since it computes fewer quantities. Data acquisition times depend on many factors other than the software, but on-line data reduction takes about 30 seconds per point on the HP 9845B, and each file write to the floppy-disk requires about 20 seconds, assuming 2000 samples are taken per point.

3.2 Data Reduction Program

A separate off-line program is used to reduce the complete data. The data reduction program has four major sections: (1) data read from floppy-disk, (2) conversion of raw data to instantaneous velocities, (3) calculation of sums of instantaneous velocity components, and (4) calculation of moments from the sums (see the block diagram in Fig. 9). Since the HP 9845B is a relatively slow micro-computer, the bulk of the calculation routines are written in ASSEMBLY language to reduce computation time.

The program reads the initialization parameters and raw data which were written on floppy-disk by the data acquisition program. The raw data is used together with the mixing frequencies, Bragg shift frequency, third beam crossing angle, and fringe spacings to calculate the three velocity components for each sample. The velocities for each sample are calculated in turn, and a running sum is maintained. After the samples have been summed, the average values of the various moments are computed, and the mean velocities and turbulence quantities are calculated from them. Reading from the floppy-disk takes about 20 seconds, while the data reduction including histogram plots requires about 90 seconds per point on the HP 9845B.

The procedure for data reduction is fairly straightforward. The raw data is converted to frequencies using the range set on the processor. The measured frequency is given by the relation
\[ f_{\text{res}} = 3.2 \times 10^4 / 2^R \times D \]  \hspace{1cm} (2)

where \( D = \) raw data.

The velocities are then calculated from the resultant frequencies, by the relation:

\[ \text{velocity} = d_f \times ((f_{\text{bragg}} - f_{\text{mix}}) - f_{\text{res}}) \]  \hspace{1cm} (3)

Once the velocities have been evaluated, average values of the various moments are computed as defined below:

\[
\begin{align*}
Su &= S<u(i)> / Ns \\
Sv &= S<v(i)> / Ns \\
Sw &= S<w(i)> / Ns \\
Suu &= S<u(i)u(i)> / Ns \\
Suv &= S<u(i)v(i)> / Ns \\
Suw &= S<u(i)w(i)> / Ns \\
Svv &= S<v(i)v(i)> / Ns \\
Swv &= S<v(i)w(i)> / Ns \\
Sww &= S<w(i)w(i)> / Ns \\
Suuv &= S<u(i)u(i)v(i)> / Ns \\
Suuw &= S<u(i)u(i)w(i)> / Ns \\
Suvv &= S<u(i)v(i)v(i)> / Ns \\
Suww &= S<u(i)w(i)w(i)> / Ns \\
Suvw &= S<u(i)v(i)w(i)> / Ns
\end{align*}
\]  \hspace{1cm} (4)

Using these definitions, the signal statistics are then calculated assuming nearly infinite sample size:

\[
\begin{align*}
\bar{u} &= Su \\
\bar{v} &= Sv \\
\bar{w} &= Sw \\
\bar{u}^2 &= Suu - SuSu \\
\bar{v}^2 &= SvV - SvSv
\end{align*}
\]  \hspace{1cm} (5)
\[ w'^2 = Sww - SwSw \]
\[ u'v' = Suv - SuSv \]
\[ u'w' = Suw - SuSw \]
\[ v'w' = Svw - SvSw \]
\[ u'^2v' = Suuv - 2SuSuv - SvSuu + 2SvSuSu \]
\[ u'^2w' = Suuw - 2SuSuw - SwSuu + 2SwSuSu \]
\[ u'v'^2 = Suvv - 2SvSuv - SuSvv + 2SuSvSv \]
\[ u'w'^2 = Suww - 2SwSuw - SuSww + 2SuSwSw \]
\[ u'v'w' = Suvw - SuSvw - SvSuw - SwSuv + 2SuSvSw \]

The implementation of the data reduction software is somewhat more complex. In order to achieve reasonable running times, the ASSEMBLY code is optimized for speed rather than clarity of operation. Operations which do not change between samples are performed only once. Thus,

\[
\text{velocity} = df * ((f_{\text{bragg}} - f_{\text{mix}}) - f_{\text{res}}) * 10^6
\]

becomes

\[
\text{fint} = f_{\text{bragg}} - f_{\text{mix}}; \quad \text{dint} = df * 10^6
\]

hence, \[ \text{velocity} = \text{dint} * (\text{fint} - f_{\text{res}}) \]

Additional calculations are required to obtain the \( w \) component velocity. Since the third set of beams measures \( w_v = w \sin 45^\circ + v \cos 45^\circ \), to find \( w \), we must also perform the calculation:

\[
w = (w_v - v \cos 45^\circ) / \sin 45^\circ.
\]

The raw data output of the digitizer is in the form of a 10 bit data word, giving a range of possible values (counts) from 0 to 1023. Since 2000 samples are normally taken for each point, a given value may be encountered many times. Accordingly, each time a new value of the raw data is encountered, the corresponding velocity is calculated and stored in a look-up table. The next time that value is encountered by the program, the
proper velocity can be easily looked up, eliminating the need for another time-consuming real variable calculation. The exact running procedure of the data reduction program therefore, is as follows:

1. Reads header and all raw data for a particular point from floppy disk.

2. Strips bits 10-13 from three raw data words corresponding to the three different velocity components, and uses them to calculate the three ranges set on the A/D.

3. Calculates various intermediate values which remain the same throughout the point.

4. Reads a data word, strips off bits 0-9, and uses this raw datum as an index to look up its corresponding velocity.

5. If it is a new value, the program calculates the velocity using the raw datum, and stores it in the appropriate place in the look-up table.

6. For the w velocity component, it finds the actual velocity from \( w = (w_v - v \cos 45°) / \sin 45° \).

7. Performs steps 4, 5, and 6 three times--once for each velocity component of the sample.

8. Updates the running sums of the velocity components and products of the velocity components.

9. Performs steps 4 through 8 for all samples.

10. Uses the sums to obtain the average velocities, Reynolds stresses, and third order products, and then prints these quantities.
11. Plots histograms of the raw data for all three channels.

12. Performs steps 1 through 11 for each profile point.

13. Plots profiles of the reduced data.

14. Tabulates normalized data profiles.

15. Writes a summary file containing the reduced data to disk.

The data reduction program also has a routine to filter noise and spurious data. Each of the 2000 data samples has 3 counts corresponding to the 3 velocities associated with it. The filtering routine causes the data reduction program to ignore samples associated with counts which are excessively far from the mean.

The filtering routine first sorts the data into three frequency tables, one for each channel. In a table, each count, i, has associated with it a number, $S_i$, which is the number of samples with that particular count.

The filtering routine finds the average count for each channel by going through the frequency tables and using the formula

$$C_j = \frac{\sum_{i=1}^{1024} i \times S_{ij}}{N_s}$$

(10)

where $j = 1,2,3 = \text{channel number}$

$C_j = \text{average count}$

$N_s = \text{number of samples}$.

Next, the filtering routine finds the standard deviation by performing the summation

$$\sigma_j^2 = \frac{\sum_{i=1}^{1024} S_{ij}(i - C_j)^2}{N_s}$$

(11)
where $\sigma_j = \text{standard deviation.}$

An input variable called $S_{\text{dev}}$ is read by the filtering routine. All counts further than $S_{\text{dev}}$ standard deviations away from the mean will be filtered out. This is done by multiplying $S_{\text{dev}}$ by $\sigma_j$ for each channel, and going through the frequency tables one more time. If the magnitude of $(i - C_j) > S_{\text{dev}} \cdot \sigma_j$, then $S_{ij}$ is set equal to zero.

The routine which calculates the statistical quantities takes each sample one at a time. The three counts associated with each sample are found and looked up in the frequency table. If for any count the table entry ($S_{ij}$) is zero, then the sample is discarded. Thus any sample which is excessively far from the mean in any one of its three counts is not used.

4. OPERATING PROCEDURE

4.1 Alignment Procedure

To achieve the best possible beam crossing and the most effective measuring volume, each module in the optics system must be carefully aligned. Detailed alignment instructions for individual optical components are given in Ref. 3. The overall alignment procedure is described in this report. Component numbers in this section refer to those shown in Fig. 1.

The first step is to check that the laser output beam is parallel to the optics table at the specified height, using the system alignment blocks. The collimator (1) should be positioned so that the laser beam goes through the center of the lens, and focused so that the beam waist occurs at the cross-over point. The aligned beam then passes through the polarization rotator (2) and into the color separator box. The attenuator (3) ensures that the beam has horizontal polarity at this point.

Each component of the color separator must be aligned separately. The dispersion prism (4) should be aligned so that the path length is the same in both prisms and is parallel to the
base of each prism. Several beams emerge from the prism, with the two brightest ones being green (514.5 nm) and blue (488 nm). The two beams are reflected out of the box using mirrors. The beams should be centered on mirrors (6) and (7). As the beams come out of the box, alignment blocks are used to check the beam positions. If both beams are off-axis in the same direction, mirror (5) is used to align them. If only one beam is off, the appropriate mirror, (6) or (7), is used for the adjustment. Beam splitter efficiency is maximized when the beam polarization is perpendicular to the plane of the split beams. This is achieved through components (8) and (11). The beam splitters (9) and (12) cannot split the beam intensity exactly in half, so one beam of each pair is always slightly brighter. Since the Bragg cells (13) and (14) normally attenuate the shifted beam, the brightest beam is frequency shifted so that the output beam pairs have nearly equal intensities. Detailed instructions for aligning the Bragg cells are given in Ref. 4. After passing through the Bragg cells, directional wedges are used to project the beams onto a distant surface (∼3 m). Any beam misalignments are more easily seen this way, and with the use of a marked mask, the beams can be adjusted to the correct orientation. Mirrors (6, 7) are used to adjust the unshifted beams, and the beam steering modules (15) are used for rotating the shifted beams along two circular arcs.

The next check is to ensure that the beams are parallel to each leg of the traverse mechanism on the transmitting optics table, so that beam alignment is maintained while traversing. The dichroic filter (Fig. 2) is adjusted so that the green beam pairs are split equally. Once the beams pass through the transmitting optics, the four beams must be arranged so that they all cross at the same position. A microscope objective is used to view the beam crossing. If the four beams are not symmetric about the optical axis, mirrors (6) or (7) can be adjusted to correct this. The beam steering modules (15) are used to ensure that the beams cross at the same point along the axis. The
third pair of beams are now aligned (by eye) so that they also
cross at the same point as the main line beams.

To align the receiving optics, a piece of translucent tape
is placed at the beam intersection point to scatter the laser
light. By tracking the scattered light, the receiving optics
are aligned to give a sharp image at the pin-hole aperture
in front of the photomultiplier tubes.

4.2 Signal Processing

The sensitivity of the photomultiplier (PM) tubes used
in the receiving optics can be varied by varying the applied
voltage. Typically, a voltage of about 1000 volts is applied.
This voltage can be increased to make the PM tubes more sensitive,
as long as the threshold levels on the processors are increased
accordingly, since the amount of noise picked up is also in­
creased.

The measured signal is mixed electronically with the signal
from a sine wave generator. The frequency of the sine wave
(the mixing frequency) is chosen based on the expected flow
velocities. The mixing frequency is chosen such that the difference
between it and the Bragg shift is about twice the maximum expected
Doppler frequency. This allows enough margin for fluctuations
about the expected Doppler frequency and still have a remaining
nonzero resultant frequency \( f_{\text{res}} = f_{\text{bragg}} - f_{\text{mix}} - f_{\text{dopp}} \). If \( f_{\text{mix}} \) is too high, a biasing results, similar to the fringe
biasing caused by stationary fringes. The number of fringes
crossed by a particle per second (as seen by the processor)
is proportional to \( \Delta t (f_{\text{bragg}} - f_{\text{mix}}) \), where \( \Delta t \) is the time taken
by a particle to cross one fringe. Noting that \( \Delta t \) is only determined
by the fringe spacing and the flow speed, if \( f_{\text{mix}} \) is increased,
the number of fringes crossed by a particle is effectively reduced.
This means that signals from particles which cross the fringes
at an angle may not have enough fringe crossings (8) to be validated,
and hence a bias towards particles moving perpendicularly to
the fringes (higher velocity) results. This gives a higher mean velocity but a lower fluctuation level.

The amplitude of the sine wave must be chosen so that an adequate signal-to-noise ratio is maintained. Typically, a peak-to-peak amplitude of 1 volt is required.

The mixed signal is then fed into single particle burst (Macrodyne) processors. The high pass and low pass filter frequencies are set so that the processor frequency is centered between the two. Usually the high pass filter is set at 0.5 MHz, which is the lowest non-zero setting, and the low pass filter is set anywhere from 2 to 16 MHz, depending on the magnitude of the velocity component being measured. The filter bandwidth should be broad enough so that no parts of the fluctuating signal are attenuated.

The gain on the processor is normally set to 10. The output of the processors is displayed on an oscilloscope, and the Doppler signals should read about 1 volt peak-to-peak. The PM tube voltage can be adjusted so that the signal is at the desired level. Signal levels of more than about 1 volt end up being clipped and will therefore not be validated by the processor.

The comparator accuracy for the 5/8 signal validation test can be set between 0 and 10 count variation. The 0 setting is the most accurate, and 11 is off. The processor manufacturer recommends that this level be usually set to 9.

The range on the processor sets the bandwidth of frequencies that the processor can see, according to Table 1. For each range, a particular frequency corresponds to a count, from 0 to 1023. The processor frequency should be matched to a number in the central column in Table 1, and the corresponding range should be set on the Macrodyne processor. An additional check can be made on the range setting by monitoring the analog output from the Macrodyne on a DC voltmeter. The output ranges from 0 to 10 volts, which corresponds to the counts from 0 to 1023. The correct range setting is that which gives about 5 volts on the voltmeter at the operating velocity.
The threshold is then set for the multi-sequence check. The threshold should be set so that the data rate is about half of the data rate at zero threshold. Another check is to block one beam in each color pair. The data rate on the corresponding channel should be zero when one beam is blocked.

To obtain good data rates in air, the flow must be seeded with uniformly sized particles. Smoke, obtained from burning mineral oil or incense, provides particles of approximately 2 μm.

The computer interface should be set up as shown in Fig. 10. First choose the coincidence mode. If shear stresses are to be evaluated, coincidence on all 3 channels must be selected. The coincidence time (5 to 50 μs) sets the window width during which coincidence is defined. This should be set at 5 μs. If only the individual velocities are desired, then the random mode can be selected. The dead time (5 to 50 μs) should then be set so that data from one particle is not recorded twice. A setting of about 25 μs is recommended for low-speed flows.

The number of words that must be multiplexed can be calculated as follows: #words = #inputs + 2. This number should be rounded to the nearest even number. In the present case a setting of six is used. The event mode is set to LDV (digital data only). The LDV-A/D switch enables both digital and analog data to be interfaced simultaneously. The counter clock frequency is set to equal the approximate data rate, and the computer select is set to HP.

5. INHERENT PROBLEMS

Some design problems inherent to 3-component LDV systems are discussed in this section. One main problem with some of the earlier designs which called for splitting a color to create the third beam pair had to do with cross-talk. This is where signals from the two channels bearing the same color could not be adequately separated. In the original design of the present
set-up, the polarization of one of the green beam pairs was rotated relative to the other, so that the receiving optics could distinguish between the two signals. However, using a relatively large angle for the off-axis beams (45°) and two separate collection lenses, cross-talk between the two green channels has been almost eliminated, thus making the polarization rotation dispensable.

Some earlier designs of 3-component LDV systems measured the \( u + w \) velocity component with the off-axis third beam pair. It is shown in Ref. 1 that measuring the \( v + w \) component instead, as done in the present system, reduces the uncertainty in the \( w \) component relative to the uncertainty in \( w \) from these earlier systems.

Another problem has to do with signal coincidence. Details of the probe volumes for the present system are shown in Fig. 3. It is clearly illustrated how the cross-over region between the three sets of beams forms a very small fraction of the overall probe volume. Thus, with heavy seeding (necessary for three-channel work), the electronics may validate data received from different particles which are not necessarily in the cross-over region but are within the coincidence time window set on the interface. This results in a lack of correlation between the measured velocities, and causes the evaluated shear stresses to be inaccurate. Two schemes have been used in an attempt to minimize this problem. First, the coincidence time was made so short that measurements from different particles may be considered instantaneous. The minimum setting of 5 \( \mu \)s available in the present hardware was used; this is equivalent to about half the flight time of a particle passing through the probe volume. The second procedure involved reducing the effective "viewed" probe length and thereby reducing the probability of this "apparent coincidence" (Fig. 5). A collection angle \( \theta \) of 30° and an aperture diameter of 0.5 mm were used to give an effective length of about 1.5 mm.
6. SAMPLE RESULTS

As an initial check on the accuracy of the present system, some preliminary measurements have been made in a vortex/mixing layer interaction, previously investigated using the cross-wire technique (Ref. 6). Since the induced cross-flow angles in this interaction are only $5^\circ - 10^\circ$, cross-wire measurements are expected to be accurate to within about 5%. A schematic of the experimental set-up is shown in Figs. 11a and b. LDV measurements of the secondary flow velocities at one streamwise station ($x = 229$ mm) are presented and compared to the cross-wire measurements in Figs. 12a and b. LDV and cross-wire measurements of the turbulence quantities at one spanwise position ($z = 13$ mm) are compared in Fig. 13.

The secondary flow velocities are qualitatively similar, although the LDV measurements indicate a somewhat higher $\bar{w}$. The normal intensity $\bar{w}^2$ also seems slightly high. The higher $\bar{w}$ measurements are more likely caused by a slight misalignment of the beams relative to the tunnel axis rather than by remnants of the apparent coincidence problem (discussed above in Section 5), since the latter would not affect the $\bar{w}$ measurements. $\bar{v}^2$ seems to agree very well whereas $\bar{u}^2$ is a bit low, and since $\bar{u}$ was a bit high, this was probably a result of fringe biasing caused by too high a mixing frequency (as discussed above in Section 4.2). (The mixing frequencies used for these measurements were 37.5 MHz for the $u$ channel, and 38 MHz for the $v$ and $w_v$ channels.) However, the normal stress measurements agree to within 10%, and the shear stresses are consequently affected. $\bar{u}'v'$ is somewhat low (about 20%) whereas $\bar{u}'w'$ is low by almost a factor of two. The measurement of $\bar{u}'w'$ with the present system seems to be very sensitive. This is due to the fact the $w_v \ast u$ is of the same order as $u \ast v \cos 45^\circ$, so any small error in the measurement of these velocities or the off-axis angle can result in large errors in $\bar{u}'w'$. $\bar{v}'w'$ in vortex affected flows is generally of the same order as $\bar{u}'w'$ and this seems to be the case with
the present measurements. With a cross-wire, $v'w'$ has to be evaluated from measurements made in four different planes about the probe axis, and hence was not measured here. The comparisons clearly demonstrate the potential of the system in measuring detailed mean flow and turbulence quantities in three-dimensional flows. Work is in progress on optimizing the problems discussed above so that the measurement accuracies may be improved.

6. CONCLUDING REMARKS

A 3-component LDV system, capable of measuring all three components of velocity simultaneously has been developed for use in low-speed three-dimensional flows. All the six components of Reynolds shear stress and higher order products of interest can hence be evaluated. The approach followed was to convert an existing 2-component system by splitting one of the colors to produce the third beam pair. The additional optical hardware required for this process is relatively minor.

For the first time, three-component measurements made with an LDV system have been compared directly with those obtained with the cross-wire technique, in a three-dimensional flow field where both techniques are expected to perform satisfactorily. The preliminary measurements are encouraging and work is in progress on improving the system accuracy.
APPENDIX

SOFTWARE FOR THE HP 9845B DESK-TOP COMPUTER

Complete listings of two programs written in BASIC and ASSEMBLY languages are included in this appendix: "LDV" for data acquisition, some on-line data reduction, and storage of data on floppy disks; and "STAT" for complete off-line data reduction from files written to disk.
REM PROGRAM LDV

! PROGRAM TO ACQUIRE DATA FROM THREE-COMPONENT LDV SYSTEM

! The program asks for initialization data and calculates the
! calibration constants from them. It reads 3 channels of raw
! LV data from the LDV A/D CI and writes them to a disk file
! together with the calibration constants. The program can reduce
! the data and display real-time histograms of the raw LV data
! if desired.

OPTION BASE 1

ICOM 16600

IDELETE ALL

IASSEMBLE Find_vel

IASSEMBLE Data_trans

IASSEMBLE Draw_hist

COM INTEGER Data(3,2000),D1(2000,6),Ns,Nn ! D1 is data buffer

COM REAL Df1,Df2,Df3

COM REAL Theta,Hub,Numix1,Numix2,Numix3

COM REAL Su,Sv,Sww,Suu,Svv,Suv,Suw,Svw

DIM Date$[80],File$[6],Name$[4],Titl$[80]

REAL Xpos,Ypos,Zpos

INTEGER A,B,Rangel,Range2

INTEGER Run,Dn,Nss,N

REAL Ph1,D,F,Ph2

REAL Lam1,Lam2,Lam3,Db,Prwid1,Pr1en1

REAL Re,Ue

REAL Prwid2,Prwid3,Pr1en2,Pr1en3

REAL Nfr1,Nfr2,Nfr3

REAL U,V,W

NS=2000

DEG

PRINT

PRINT " ** PROGRAM LDV : 3-COMPONENT VELOCITY DATA >> ** "

PRINT

PRINT "PROGRAM STRUCTURE"

PRINT " 1. INITIALIZE VARIABLES AND CALCULATE PARAMETERS"

PRINT " 2. ACQUIRE DATA FROM A/D"

PRINT " 3. WRITE TO FLOPPY DISC"

PRINT

PRINT

! ** CHECK HISTOGRAMS **

Ans$="N"

INPUT "DO YOU WISH TO LOOK AT HISTOGRAMS ? (Y/N, DEFAULT N)";,Ans$

IF Ans$="Y" THEN GOSUB Hist

! ** INITIALIZE RUN **

Run=1

PRINT " ** INITIALIZATION ** "

PRINT " ENTER RUN PARAMETERS:"

PRINT

INPUT "Enter date and time:";,Date$

INPUT "Enter 1-Line Name For Profile :";,Titl$

INPUT "No. of data samples per point (2000 maxm.) :";,Ns

! BEAM SPACING D IS FIXED

D=.013

! FOCAL LENGTH F IS FIXED
F=.381

WAVELENGTHS OF 3 BEAMS ARE FIXED

Lam1=4.88E-7
Lam2=5.145E-7
Lam3=5.145E-7

REFERENCE VELOCITY SET TO ZERO

Ue=0

CALCULATE HALF-ANGLES FROM BEAM SPACINGS AND FOCAL LENGTH

Ph1=ATN(D/2/F)
Ph2=Ph1*2

CALCULATE FRINGE SPACINGS FROM WAVELENGTHS AND HALF-ANGLES

Df1=Lam1/2/SINC(Ph1)
Df2=Lam2/2/SINC(Ph1)
Df3=Lam3/2/SINC(Ph1)

BEAM DIAMETER Db IS FIXED

Db=1.20E-3

CALCULATE PRO3E VOLUME WIDTH AND LENGTH

Prw1=4*Lam1*F/PI/Db/COSC(Ph1)
Prw1=4*Lam1*F/PI/Db/SINC(Ph1)
Prw2=4*Lam2*F/PI/Db/COSC(Ph1)
Prw2=4*Lam2*F/PI/Db/SINC(Ph1)
Prw3=Prw2
Prw1=Prw1/2
Prw2=Prw2/2

GET MORE RUN PARAMETERS

INPUT "Enter Bragg shift frequency (MHz) ":,Nub
If Kill w=3 THEN GOTO 1110
INPUT "Is this a two-channel or three-channel run (2/3, default 3) ",Kill

INPUT "Enter mixing frequency (MHz, 2 nos.)",Numix1,Numix2
Numix1=0
Theta=0
GOTO 1130
INPUT "Enter mixing frequency (MHz, 3 nos.)",Numix1,Numix2,Numix3
INPUT "Enter 3rd beam angle (degrs) ":,Theta
INPUT "Enter tunnel reference voltage (volts)",Vref
PRINT IS 0

PRINT HEADER

PRINT Titl$
PRINT "TEST DATE AND TIME ":,Date$
PRINT "BEAM SPACINGS (m) ":,D
PRINT "TOTAL ANGLE BETWEEN BEAMS (Degs) =",Ph2
PRINT "FRINGE SPACINGS (m) =",Df1,Df2,Df3
PRINT "PROBE WIDTHS (m) =",Prwid1,Prwid2,Prwid3
PRINT "PROBE LENGTHS (m) =",Prlen1,Prlen2,Prlen3
PRINT "NO. OF FRINGES =",Nfr1,Nfr2,Nfr3
PRINT "BRAgg SHIFT FREQUENCY (MHz) =",Nub
PRINT "MIXING FREQUENCY (MHz) =",Numix1,Numix2,Numix3
PRINT "THIRD BEAM SET ANGLE (Degs) =",Theta.
PRINT "RUN NUMBER = ",Run
PRINT "TUNNEL REFERENCE VOLTAGE (volts) =",Vref
PRINT "DATA ACQUISITION ** 
REDIM Data(3,Ns)
Ans$="N"
INPUT "DO YOU WISH TO CHANGE FILE NAME? (Y/N, DEFAULT N) ",Ans$
IF Ans$="N" THEN GOTO 1490
INPUT "Enter 4-digit filename: ",Name$
PRINTER IS 0
PRINT "4-digit filename for profile: ",Name$
INPUT "Enter Disk Number: ",Dn
PRINT "Disk Number = ",Dn
PRINTER IS 16
INPUT "Enter X, Y, AND Z locations: ",Xpos,Ypos,Zpos
! TAKE DATA
PRINTER IS 0
GOSUB Atod
CALCULATE SAMPLE VELOCITIES
Ans$="N"
INPUT "DO YOU WISH TO OBTAIN ESTIMATES OF U AND V? (Y/N, DEFAULT N) ",Ans$
IF Ans$="N" THEN GOTO 1720
RAD
ICALL Find_vel
GOTO 2420
WRITE CALIBRATION CONSTANTS AND DATA TO DISK
GOSUB Dfile
Ans$="Y"
INPUT "DO YOU WISH TO TAKE ANOTHER POINT? (Y/N, DEFAULT Y) ",Ans$
IF Ans$="N" THEN GOTO 1820
Run=Run+1
Ans$="N"
INPUT "DO YOU WISH TO CHANGE ANY PARAMETERS? (Y/N, DEFAULT N) ",Ans$
IF Ans$="Y" THEN GOTO 540
GOTO 1410
END

** END OF MAIN PROGRAM LDV **

Atod: Subroutine for input from the LDV-A/D CI

! Enter routine with Ns = no. samples

DISP "Press CONT to initiate data acquisition"
PAUSE
DISP "Acquiring Data"
RESET 10
CONTROL MASK 10;1
WRITE IO 10,5;0
WRITE IO 10,5;1

Ie=6*Ns

FOR I=1 TO 5
Dummy=READBIN(10)
NEXT I
REDIM D1(Ns,6)

ENTER Ie WFHS Nt NOFORMAT;D1(*) ! fast data acquisition
WRITE IO 10,5;0
PRINT
DISP "Data acquisition complete"
PRINT
ICALL Data_tran;
RETURN

Dfile: ! write data file to floppy disc
PRINTER IS 16
PRINT "DATA FILE WRITE TO FLOPPY DISK **"
PRINT "At this point be sure there is a floppy in drive 0 of"
PRINT "the 9895A with space for a file of 101, 256-byte records."
PRINT
PRINT
Ans$="Y".
INPUT "DO YOU WISH TO WRITE THESE DATA TO DISK ? (Y/N, DEFAULT Y)",Ans$
IF Ans$="N" THEN GOTO 23813
File$=Name$&VAL$(Run)
PRINT 
ASSIGN File$ TO #1
PRINT 
MAT PRINT
ASSIGN * TO #1
PRINT 

Run=Run-1
RETURN

! 

27
2410 Uguess: ! Estimate U-component of velocity
2420 DISP "CALCULATING ESTIMATES OF U AND V VELOCITIES"
2430 !
2440 !
2450 IF Kill w=3 THEN GOTO 2490
2460 DATA 0,0,0
2470 RESTORE 2460
2480 READ Sw,Sww,Suw
2490 Ubar=Su/Ns
2500 Vbar=Sw/Ns
2510 Wbar=Sww/Ns
2520 Upri2=Suw/Ns-Ubar*Ubar
2530 Vpri2=Sww/Ns-Vbar*Vbar
2540 Wpri2=Sww/Ns-Wbar*Wbar
2550 Uvbar=Suv/Ns-Ubar*Vbar
2560 Uwbar=Suw/Ns-Ubar*Wbar
2570 Vwbar=Svw/Ns-Vbar*Wbar
2580 PRINTER IS 0
2590 PRINT "ESTIMATE OF QUANTITIES FROM SAMPLES ":"Ns
2600 PRINT "Ubar =",Ubar
2610 PRINT "Vbar =",Vbar
2620 PRINT "Wbar =",Wbar
2630 PRINT "Upri2 =",Upri2
2640 PRINT "Vpri2 =",Vpri2
2650 PRINT "Wpri2 =",Wpri2
2660 PRINT "Uvbar =",Uvbar
2670 PRINT "Uwbar =",Uwbar
2680 PRINT "Vwbar =",Vwbar
2690 GOTO 1700
2700 ISOURCE ! NAM Find_vel
2710 ISOURCE !
2720 ISOURCE ! This subroutine converts raw data counts into instantaneous
2730 ISOURCE ! velocities, then sums several different products of the
2740 ISOURCE ! velocity components. All input and output data is passed
2750 ISOURCE ! through the COMMON storage area. The inputs are the raw
2760 ISOURCE ! data array (Arrayd), the Bragg shift frequency (Nub), the
2770 ISOURCE ! mixing frequencies (Numix1, Numix2, Numix3), the fringe
2780 ISOURCE ! spacings (DF1, DF2, DF3), the crossing angle of the third
2790 ISOURCE ! beam (Theta), and the number of samples in a data point (Ns).
2800 ISOURCE ! The outputs are the summations of various products of the
2810 ISOURCE ! velocity components, including U, V, W, U*U, V*V, W*W, U*V,
2820 ISOURCE ! U*W, and V*W.
2830 ISOURCE !
2840 ISOURCE ! EXT Get_value ! Declare subroutines stored
2850 ISOURCE ! EXT Get_info ! outside of the main program.
2860 ISOURCE ! EXT Get_element
2870 ISOURCE ! EXT Int_to_rel
2880 ISOURCE ! EXT Rel_math
2890 ISOURCE ! EXT Put_value
2900 ISOURCE !
2910 ISOURCE ! COM
2920 ISOURCE Data_par: INT (*) ! Declare common variables.
2930 ISOURCE D1_par: INT (*)
2940 ISOURCE Ns_par: INT
2950 ISOURCE Mn_par: INT
2960 ISOURCE DF1_par: REL
2970 ISOURCE DF2_par: REL
2980 ISOURCE DF3_par: REL
2990 ISOURCE Theta_par: REL
3000 ISOURCE Nub_par: REL
3010 ISOURCE Hmix1_par: REL
3020 ISOURCE Hmix2_par: REL
3030 ISOURCE Hmix3_par: REL
3040 ISOURCE Su_par: REL
3050 ISOURCE Sv_par: REL
3060 ISOURCE Sw_par: REL
3070 ISOURCE Suu_par: REL
3080 ISOURCE Svv_par: REL
3090 ISOURCE Sww_par: REL
3100 ISOURCE Suw_par: REL
3110 ISOURCE Svw_par: REL
3120 ISOURCE Suv_par: REL
3130 ISOURCE !
3140 ISOURCE Arrayd: BSS 39 ! Reserve space for data array
3150 ISOURCE Elementd: EQU Arrayd+16 ! descriptor.
3160 ISOURCE Array1: BSS 4096 ! Reserve space for lookup tables used
3170 ISOURCE Array2: BSS 4096 ! for count to velocity conversion.
3180 ISOURCE Array3: BSS 4096
3190 ISOURCE Ns: BSS 1 ! Reserve space for various input
3200 ISOURCE Df1: BSS 4 ! and output variables.
3210 ISOURCE Df2: BSS 4
3220 ISOURCE Df3: BSS 4
3230 ISOURCE Theta: BSS 4
3240 ISOURCE Nub: BSS 4
3250 ISOURCE Numix1: BSS 4
3260 ISOURCE Numix2: BSS 4
3270 ISOURCE Numix3: BSS 4
3280 ISOURCE Su: BSS 4
3290 ISOURCE Sv: BSS 4
3300 ISOURCE Sw: BSS 4
3310 ISOURCE Suu: BSS 4
3320 ISOURCE Svv: BSS 4
3330 ISOURCE Sww: BSS 4
3340 ISOURCE Suw: BSS 4
3350 ISOURCE Svw: BSS 4
3360 ISOURCE Swu: BSS 4
3370 ISOURCE Count: BSS 1 ! Count and I are general purpose index vari-
3380 ISOURCE I: BSS 1 ! ales. Count is usually 0, 1, or 2, to denote
3390 ISOURCE Check: BSS 1 ! whether U, V, or W is being calculated.
3400 ISOURCE Int: BSS 1 ! Int, Address, and Offset are all general
3410 ISOURCE Address: BSS 1 ! purpose storage areas.
3420 ISOURCE Offset: BSS 1
3430 ISOURCE R1: BSS 4 ! R1, R2, and R3 are the count-to-
3440 ISOURCE R2: BSS 4 ! frequency conversion factors.
3450 ISOURCE R3: BSS 4
3460 ISOURCE Xvar: BSS 4 ! Xvar and Yvar are general purpose real
3470 ISOURCE Yvar: BSS 4 ! number storage areas.
3480 ISOURCE U: BSS 4 ! reserve space for instantaneous velocity
3490 ISOURCE V: BSS 4 ! components.
3500 ISOURCE W: BSS 4
3510 ISOURCE Uu: BSS 4
3520 ISOURCE Uv: BSS 4
3530 ISOURCE Cost: BSS 4 ! Cos and Sin are the cosine and sine of Theta.
3540 ISOURCE Sin: BSS 4
3550 ISOURCE Rad: DAT 5.729578E1
3560 ISOURCE Mill: DAT 1.E6
3570 ISOURCE One: DAT 1.
3580 ISOURCE Zero: DAT 0.
3590 ISOURCE !
3600 ISOURCE SUB
Find vel:  LDA =Ns  ! Get number of samples.
LDB =Ns_par
JSM Get_value
LDA =Arrayd  ! Get parameters of data array.
LDB =Data_par
JSM Get_info
LDA =Df1  ! Get input parameters.
LDB =Df1_par
JSM Get_value
LDA =Df2
LDB =Df2_par
JSM Get_value
LDA =Df3
LDB =Df3_par
JSM Get_value
LDA =Theta
LDB =Theta_par
JSM Get_value
LDA =Nub
LDB =Nub_par
JSM Get_value
LDA =Numix1
LDB =Numix1_par
JSM Get_value
LDA =Numix2
LDB =Numix2_par
JSM Get_value
LDA =Numix3
LDB =Numix3_par
JSM Get_value
LDA =0
STA Count
The loop headed by Get_freq is repeated three times to get
the count-to-frequency conversion factors (which depend on
the range) for U, V, and W. Whenever a loop is controlled by
the variable "Count", the loop contains operations which are
the same for U, V, and W.
LDA =8
STA Count
Get_freq:  LDA =Count  ! Get the first word of the column of
LDB Ns  ! the data array which contains the
JSM Get_element  ! the velocity component for which we want
to get the range.
LDA =Int
LDB =Arrayd
JSM Get_element
LDA =Int  ! Mask and rotate to get the four
LDB =15360  ! bits containing the range.
AND B
SAR 18
TCA  ! Subtract from 15 to get the
LDB =15  ! actual range.
AND A
LDB =1
SZA Loopend
Loop:  SBL 1  ! Use the range to find the power
DSZ A  ! of two needed for the divisor.
JMP Loop
Loopend:  STB Int  ! Convert the power of two into a
4210 ISOURCE STA Oper_1 ! real number.
4220 ISOURCE LDA =Yvar
4230 ISOURCE STA Result
4240 ISOURCE JSM Int_to_rej
4250 ISOURCE LDA =3.2E4 ! Divide 3.2E4 by the appropriate
4260 ISOURCE LDB =Xvar ! power of two, using BCD math.
4270 ISOURCE XFR 4
4280 ISOURCE STB Oper_1
4290 ISOURCE LDA =Yvar
4300 ISOURCE STA Oper_2
4310 ISOURCE LDA Count ! Decide whether to put the result
4320 ISOURCE LDB =4 ! in R1, R2, or R3, depending on
4330 ISOURCE MPY ! Count.
4340 ISOURCE ADA =R1
4350 ISOURCE STA Result
4360 ISOURCE LDA =2
4370 ISOURCE LDB =147155B ! Now, finally, call the utility to
4380 ISOURCE JSM Re1_math ! perform the division.
4390 ISOURCE ISZ Count
4400 ISOURCE LDA =3 ! Increment and check Count so as
4410 ISOURCE CPA Count ! to follow the loop three times.
4420 ISOURCE JMP +2
4430 ISOURCE JMP Get_freq
4440 ISOURCE !
4450 ISOURCE LDA =Array1 ! Zero out the entire count-to-
4460 ISOURCE LDB =768 ! velocity conversion table so that
4470 ISOURCE Continue: CLR 16 ! it must be recalculated for each
4480 ISOURCE ADA =16 ! point. (This must be done if the
4490 ISOURCE DSZ B ! mixing frequencies or ranges are
4500 ISOURCE JMP Continue ! changed between counts.)
4510 ISOURCE !
4520 ISOURCE LDA =Su ! Set initial values of Su, Sv, 
4530 ISOURCE LDB =9 ! Sw, Suu, etc. to zero.
4540 ISOURCE Clear: CLR 4
4550 ISOURCE ADA =4.
4560 ISOURCE DSZ B
4570 ISOURCE JMP Clear
4580 ISOURCE !
4590 ISOURCE LDA =Theta ! Convert Theta from degrees to 
4600 ISOURCE STA Oper_1 ! radians using the Rel_math 
4610 ISOURCE LDA =Rad ! utility.
4620 ISOURCE STA Oper_2
4630 ISOURCE LDA =Xvar
4640 ISOURCE STA Result
4650 ISOURCE LDA =2
4660 ISOURCE LDB =147155B
4670 ISOURCE JSM Re1_math
4680 ISOURCE !
4690 ISOURCE LDA =Xvar ! Find the sine and cosine of
4700 ISOURCE STA Oper_1 ! Theta, and store them in the
4710 ISOURCE LDA =Sin ! locations Sin and Cos, respectively.
4720 ISOURCE STA Result
4730 ISOURCE LDA =1
4740 ISOURCE LDB =34213B
4750 ISOURCE JSM Rel_math
4760 ISOURCE LDA =Cos
4770 ISOURCE STA Result
4780 ISOURCE LDA =1
4790 ISOURCE LDB =34224B
4800 ISOURCE JSM Rel_math

31
The loop defined by Get_int calculates intermediate values used in converting counts to velocities. These values are the same for all samples in a point, so they can be calculated separately. The loop calculates \( \langle H_u - N_{\text{mix}} \rangle \) and \( \langle M_{\text{ill}} \times D_f \rangle \), where \( N \) is 1, 2, and 3.

LDA = 3
STA Count

Get_int: LDA Count
A DA = -1
SAL 2
STA Offset

LDA = H_u ! Find \( N_{\text{mix}} = H_u - N_{\text{mix}} \).

LDA = Num_x_1

A DA Offset

LDA = N_{\text{ill}}

LDA = Oper_2

LDA = 2
LDB = 146717B
JSM Rel math
LDA = D_f ! Find \( D_f = M_{\text{ill}} \times D_f \).

A DA Offset

LDA N_{\text{s}}

STB Offset

LDB N_{\text{s}}

MPY ADA I
ADA = -1

Figure out which element of the data array we want to pick up.
Get a raw datum from the data array.

Strip off the first six bits of the raw data word.

See if Count - Check.

If true, use the stripped data word as an index. If not, use 1024 minus the data word.

Store the count we have gotten.

Now, use Count to find out which lookup table array we want to use, and use the count we got from the data array to find exactly where in the table we want to go.

If that table entry is zero, calculate a velocity for it.

Convert the count to a real number.

Divide the range we found earlier by the count to get a frequency.

Find (Nub-NumixN)-FrequencyN.

If we are calculating U, reverse the sign of ((Nub-NumixN)-FrequencyN) so as to reverse the sign of U.

Leave V and W alone.
Find Velocity = ((Hub - NumixN) - FrequencyH) * (Mill * DfH) and store in a place in the lookup table corresponding to the data count.

Now transfer the velocity from the lookup table to U, V, or W, as appropriate.

Have U, V, and W all been calculated?

If not, go back again.

If so, set Count = a.

Now we convert the w we have obtained (which is measured at an angle Theta to the V-axis) to the W we want (which should be measured at an angle of 90 degrees to the V-axis).

Thus find W = (Wv - V * Cos(Theta)) / Sin(Theta).
Now take running sums of U, V, W, and several products of these velocities. The sums are taken using the utility "Add". The sums are calculated in an unusual sequence in order to reduce the number of steps needed to calculate them. Program steps which produce sums not considered necessary have been included, but these steps are preceded by the "!" comment marker.
LDA =Sw  ! Find Sw=Sw+W
LDB =W
JSM Add

LDA =U  ! Find U*W
STA Oper_1
LDA =Xvar
STA Result
LDA =2
LDB =1470378
JSM Rel_math

LDA =Suw  ! Find Suw=Suw+(U*W)
LDB Result
JSM Add

LDA =W  ! Find V*W
STA Oper_1
LDA =V
STA Oper_2
LDA =Xvar
STA Result
LDA =2
LDB =1470378
JSM Rel_math

LDA =Svw  ! Find Svw=Svw+(V*W)
LDB Result
LDA =2
LDB =1470378
JSM Rel_math

LDA =Sww  ! Find Sww=Sww+(W*W)
LDB Result
LDA =2
LDB =1470378
JSM Rel_math

LDA =Su
LDB =Su_par
JSM Put_value
LDA =Sv
LDB =Sv_par
JSM Put_value
LDA =Sw
LDB =Sw_par
JSM Put_value
LDA =Suu

Place the finished sums in the COMMON region so that
the BASIC program has access to them, and then return to
the BASIC program.

LDA =Su
LDB =Su_par
JSM Put_value
LDA =Sv
LDB =Sv_par
JSM Put_value
LDA =Sw
LDB =Sw_par
JSM Put_value
LDA =Suu
The utility "Add" is used to add up the running sums.

Subroutine to transfer raw data from input array to storage array.

Declare subroutines stored outside of the main program.

Declare common variables.
SUB

Data_trans: LDA =Arrayd ! Get parameters of data storage array.
LDB =Data_par
JSM Get_info
LDA =Array1 ! Get parameters of data input array.
LDB =D1_par
JSM Get_info
LDA =Ns ! Get number of samples.
LDB =Ns_par
JSM Get_value
!
LDA =0
STA I
LDA =0
STA Count1
LDA =2
STA Count2
!
LDA I
LDB =6
MPY
ADA Count2
STA Element1
LDA =Int
LDB =Array1
JSM Get_element
!
LDA Count1
LDB Ns
MPY
ADA I
STA Elementd
LDA =Int
LDB =Arrayd
JSM Put_element
ISZ Count1
ISZ Count2
LDA =-3
ADA Count1
SZA ++2
JMP Start2
!
ISZ I
LDA I
TCA
ADA Ns
SZA ++2
JMP Start1
RET 1
END Data_trans
!

******************************************************************************
** HISTOGRAM SUBPROGRAM **
******************************************************************************

Hist: ! Subroutine to produce online histograms of raw LDY data.
ON KEY #0 GOTO 9270
Hn=0
Ns_temp=Ns
! fast data acquisition

The subroutine works by first going to the samples-per-count tables and using them to draw the old histograms in black (Bit=0) to erase them. Then it calculates new samples-per-count tables from the data acquired from the LDV, and uses the new tables to draw histograms in white (Bit=1). Then the subroutine returns to the main program. The very first time (determined by Ntimes(=Nn)) the subroutine is called it doesn't erase the old histograms because there aren't any.

Declare subroutines stored outside of the main program.
Declare common variables.
Reserve space the number of for data array for tables which hold the number of samples per count.

Reserve space for data array
! Reserve space for tables which hold
9610 ISOURCE Ns: BSS 1 ! Count and I are general purpose index vari-
9620 ISOURCE Count: BSS 1 ! ables. Count is usually 0, 1, or 2, to denote
9630 ISOURCE I: BSS 1 ! whether U, V, or W is being calculated.
9640 ISOURCE Check: BSS 1 ! Int, Address, and Offset are all general
9650 ISOURCE Int: BSS 1 ! purpose storage areas.
9660 ISOURCE Address: BSS 1
9670 ISOURCE X_coord: BSS 1
9680 ISOURCE Y_coord: BSS 1
9690 ISOURCE Bit: BSS 1
9700 ISOURCE Ntimes: BSS 1
9710 ISOURCE !
9720 ISOURCE LIT 30
9730 ISOURCE !
9740 ISOURCE SUB
9750 ISOURCE Draw_hist: LDA =Ns ! Get number of samples.
9760 ISOURCE LDB =Ns_par
9770 ISOURCE JSM Get_value
9780 ISOURCE LDA =Arrayd ! Get parameters of data array.
9790 ISOURCE LDB =Data_par
9800 ISOURCE JSM Get_info
9810 ISOURCE LDA =Ntimes ! Get Ntimes, which tells if this is
9820 ISOURCE LDB =Ntime_par ! the first time Draw_hist is being
9830 ISOURCE JSM Get_value ! called.
9840 ISOURCE !
9850 ISOURCE LDA =0 ! If this is the first time Draw_hist
9860 ISOURCE STR Bit ! is being called, then jump to the
9870 ISOURCE LDA Ntimes ! data acquisition section. If not then
9880 ISOURCE RZA +2 ! write over the old histogram with
9890 ISOURCE JMP Acquire ! black to erase it
9900 ISOURCE !
9910 ISOURCE Do_graph: LDA =0 ! Produce histograms for all three
9920 ISOURCE STR Count ! channels.
9930 ISOURCE LDA Count ! Go to the end of the appropriate
9940 ISOURCE ADA =1 ! samples-per-count table.
9950 ISOURCE LDB =1024
9960 ISOURCE STB 1
9970 ISOURCE MPY
9980 ISOURCE ADA =Array1
9990 ISOURCE ADA =-1
10000 ISOURCE STR Address
10010 ISOURCE Make_rod: LDB Address,I ! Add together each adjacent pair of
10020 ISOURCE DSZ Address ! entries in the table.
10030 ISOURCE ADD Address,I
10040 ISOURCE DSZ Address
10050 ISOURCE DSZ 1
10060 ISOURCE RZB +2 ! If the result is zero, go to the next
10070 ISOURCE JMP Skip_bits ! pair of entries. If not, make sure
10080 ISOURCE STB Int ! the sum is <150 and then draw a
10090 ISOURCE ADA =-150 ! column with height equal to the sum.
10100 ISOURCE SBM +3
10110 ISOURCE LDB =150
10120 ISOURCE STB Int
10130 ISOURCE !
10140 ISOURCE LDA I ! Calculate the X-coordinate of the
10150 ISOURCE SAR 1 ! column.
10160 ISOURCE ADA =20
10170 ISOURCE STR X_coord
10180 ISOURCE !
10190 ISOURCE Make_bit: LDA =150 ! Calculate the Y-coordinate of the
10200 ISOURCE LDB Count ! top of the column.

40
10210 ISOURCE  ADB =1
10220 ISOURCE  MPY
10230 ISOURCE  LDB Int
10240 ISOURCE  TCB
10250 ISOURCE  ADA B
10260 ISOURCE  STA Y_coord
10270 ISOURCE  !
10280 ISOURCE  LDA =13  ! Prepare the graphics screen for
10290 ISOURCE  STA Pa  ! input.
10300 ISOURCE  LDA =51B
10310 ISOURCE  SFC *
10320 ISOURCE  STA R5
10330 ISOURCE  LDA Y_coord  ! Calculate word location.
10340 ISOURCE  LDB =36
10350 ISOURCE  MPY
10360 ISOURCE  LDB X_coord
10370 ISOURCE  SBR 4
10380 ISOURCE  ADA B
10390 ISOURCE  CMA
10400 ISOURCE  SFC *
10410 ISOURCE  STA R4  ! Input word location.
10420 ISOURCE  STA R7
10430 ISOURCE  !
10440 ISOURCE  LDA X_coord  ! Calculate bit location.
10450 ISOURCE  AND =1?B
10460 ISOURCE  LDB Bit
10470 ISOURCE  SBL 15
10480 ISOURCE  IOR B
10490 ISOURCE  SFC *
10500 ISOURCE  STA R4  ! Input bit location.
10510 ISOURCE  STA R7
10520 ISOURCE  DSZ Int
10530 ISOURCE  JMP Make_bit
10540 ISOURCE  !
10550 ISOURCE  Skip_bits: DSZ I  ! Make histograms for all three
10560 ISOURCE  JMP Make_rod  ! channels.
10570 ISOURCE  !
10580 ISOURCE  ISZ Count  ! Make histograms for all three
10590 ISOURCE  LDA =-3  ! channels.
10600 ISOURCE  ADA Count
10610 ISOURCE  SZA ++2
10620 ISOURCE  JMP Make_hist
10630 ISOURCE  !
10640 ISOURCE Acquire: LDA Bit  ! If Bit=1 then the new histograms
10650 ISOURCE  SZA ++2  ! have been drawn. Go back to the
10660 ISOURCE  JMP Stop  ! main program. If not, continue.
10670 ISOURCE  !
10680 ISOURCE Clear: LDA =Array1  ! Zero out the tables which hold the
10690 ISOURCE  CLR 16  ! number of samples per count.
10700 ISOURCE  ADA =16
10710 ISOURCE  DSZ B
10720 ISOURCE  JMP Clear
10730 ISOURCE  !
10740 ISOURCE  LDA =0  ! Prepare to write new samples-per-  
10750 ISOURCE  STA Count  ! count tables.
10760 ISOURCE  LDA =5
10770 ISOURCE  STA Check
10780 ISOURCE  LDA Ns
10790 ISOURCE  STA I
10800 ISOURCE  
41
Figure out which element of the data array we want to pick up. Get a raw datum from the data array. Strip off the first six bits of the raw data word. See if Count = Check. If true, use the modified data word as an index. If not, use 11324 minus the data word. Store the count we have gotten. Increment the appropriate place in the samples-per-count table by one. Have U, V, and W all been done? If not, go back again. If so, set Count=0. Continue to fill samples-per-count tables until out of samples. Now set Bit=1 and go back and draw the new histograms in white.
REM PROGRAM STAT
! PROGRAM TO REDUCE RAW DATA FROM THREE-COMPONENT LDV SYSTEM
! Input is 3 channels of raw LV data and calibration constants
! Output is the three components of mean velocity, all the
! components of Reynolds stress and some selected third-order
! products. Can filter out counts which are excessively far
! from the mean. Can display histograms of raw and
! filtered data for all 3 channels. Plots data at end and displays
! reduced data for entire run. Can write a summary file containing
! all reduced data and calibration constants.
OPTION BASE 1
ICOM 10000
DELETE ALL
I$ASSEMBLE Find vel
COM INTEGER Data(3,2000),Hs,Ngs,Countbin(3,1024)
COM INTEGER Range1,Range2,Range3
COM REAL Sdev,Df1,Df2,Df3
COM REAL Theta,Nub,Numix1,Numix2,Numix3
COM REAL Su,Su,Su,Su,Su,Su,Su,Su,Su,Su
COM REAL Su,Su,Su,Su,Su,Su,Su,Su,Su,Su
DIM L1(3),R1(3),L1(3),U1(3),Scale(3)
DIM Date#[80],File#[6],Name#[4],Titl#[80]
DIM L1(3),R1(3),L1(3),U1(3),L lim(1),U lim(1),L lim(2),U lim(2),L lim(3),U lim(3)
DATA .4,.13135,.131325
MAT READ Scale
PRINT "** << PROGRAM STAT : 3-COMPONENT VELOCITY DATA >> **"
PRINT "PROGRAM STRUCTURE"
PRINT "1. Read raw data from floppy disc"
PRINT "2. Convert to velocity"
PRINT "3. Calculate statistics"
PRINT "4. Print results"
PRINT "5. Write to disc file"
PRINT
PRINT "** Read raw data from floppy disc **
INPUT "Enter parent filename (or E to exit program) :",Name$
IF Name$="E" THEN GOTO 1780
INPUT "Enter no. of first and last data files ",Filel,Nf
Kill w=3
INPUT "Is this a two-channel or three-channel run (2/3, Default 3) ?",Kill
Hist$="Y"
INPUT "Do you wish to see histograms of the data points (Y/N, Default Y)?", Hist$

INPUT "Do you wish to filter the output (Y/N, Default N)?", Filter$

IF Filter$="N" THEN GOTO 660

INPUT "Enter standard deviation to throw out", Sdev

DISP "READING RAW DATA FROM DISK"

MASS STORAGE IS ":HS,0,0"

PRINTER IS 0

Filno=File1

IF Filno>Nf THEN GOTO 1740

Files=Name$&VAL$(Filno)

ASSIGN Files TO 1

READ 1; Date$

READ 1; Tit1$

READ 1; Name$

READ 1; Dn

READ 1; Nub, Numix1, Numix2, Numix3, Theta, Run

READ 1; Vref, Ue, Df1, Df2, Df3

READ 1; Xpos, Ypos, Zpos, Ns

MAT READ 1; Data

PRINT Tit1$

PRINT "DATE AND TIME OF TEST ":, Date$

PRINT "FILE NAME ON DISK ":, Name$

PRINT "DISK NUMBER ":, Dn

PRINT "RUN NO. (POINT NO. IN PROFILE) ":, Run

PRINT "X,Y,Z = ", Xpos, Ypos, Zpos

CALCULATE VELOCITIES

DISP "CALCULATING VELOCITIES AND TAKING RUNNING SUMS"

ICALL Find_vel

CALCULATE STATISTICS

DISP "CALCULATING STATISTICS"

IF Kill w=3 THEN GOTO 1133

DATA 0,0,0,0,0,0

RESTORE 990

READ Su, Sw, Suw, Suv, Suu, Suvw, Suvw, Suw

Ubar=Su/Ngs

Vbar=Sw/Ngs

Wbar=Suw/Ngs

Upri2=Suw/Ngs-Ubar*Ubar

Ypri2=Suv/Ngs-Vbar*Vbar

Wpri2=Suw/Ngs-Wbar*Wbar

Uvbar=Suw/Ngs-Ubar

Wvbar=Suvw/Ngs-Wbar

Uvbar=Suw/Ngs-Ubar*Wbar

Wvbar=Suvw/Ngs-Wbar*Ubar

Uuvbar=Suuv/Ngs-2*Ubar*Vbar*Ubar

Uvuvbar=Suuv/Ngs-2*Vbar*Ubar

Wuvbar=Suuv/Ngs-2*Ubar*Wbar

Wvuvbar=Suuv/Ngs-2*Vbar*Wbar

Uuwbar=Suww/Ngs-2*Ubar

Wuwbar=Suww/Ngs-2*Wbar

Uvwbar=Suww/Ngs-2*Vbar

Wuvwbar=Suww/Ngs-2*Wbar

STORE RESULTS IN ARRAY FOR PLOTTING

F1=Filno=File1+1
1210 Point(F1,1)=Ypos
1220 Point(F1,2)=Ubar
1230 Point(F1,3)=Vbar
1240 Point(F1,4)=Wbar
1250 Point(F1,5)=Upri2
1260 Point(F1,6)=Ypri2
1270 Point(F1,7)=Wpri2
1280 Point(F1,8)=Uvbar
1290 Point(F1,9)=Wvbar
1300 Point(F1,10)=Wvbar
1310 Point(F1,11)=Uvbar
1320 Point(F1,12)=Uvbar
1330 Point(F1,13)=Uvbar
1340 Point(F1,14)=Uvbar
1350 Point(F1,15)=Uvbar
1360 IF Ypos>Max_y THEN GOTO 1390
1370 Max_y=Ypos
1380 Min_u=Ubar
1390 IF Ypos<Min_y THEN GOTO 1450
1400 Min_y=Ypos
1410 Max_u=Ubar
1420 ! PRINT RESULTS
1430 !
1450 PRINT 
1460 PRINT "Bragg shift frequency = ",Nub
1470 PRINT "Mixing frequencies=",Numix1,Numix2,Numix3
1480 PRINT "Third beam angle = ",Theta
1490 PRINT "Run number = ",Run
1500 PRINT "Tunnel reference voltage (volts) = ",Vref
1510 PRINT "Fringe spacings = ",Df1,Df2,Df3
1520 PRINT "No. of samples = ",Ngs
1530 PRINT "Ranges = ",Range1,Range2,Range3
1540 PRINT "Filter [no. of std. deviations] = ",Sdev
1550 PRINT "Adjusted no. of samples = ",Ngs
1560 PRINT "Ubar = ",PROUNDCUbar,-4)
1570 PRINT "Vbar = ",PROUNDCVbar,-4)
1580 PRINT "Wbar = ",PROUNDCWbar,-4)
1590 PRINT "Upri2 = ",PROUNDCUpri2,-4)
1600 PRINT "Ypri2 = ",PROUNDCYpri2,-4)
1610 PRINT "Wpri2 = ",PROUNDCWpri2,-4)
1620 PRINT "Uvbar = ",PROUNDCUvbar,-4)
1630 PRINT "Wvbar = ",PROUNDCWvbar,-4)
1640 PRINT "UVbar = ",PROUNDCUVbar,-4)
1650 PRINT "UVVbar = ",PROUNDCUVVbar,-4)
1660 PRINT "UWbar = ",PROUNDCUWbar,-5)
1670 PRINT "UWbar = ",PROUNDCUWbar,-5)
1680 PRINT "UnWbar = ",PROUNDCUWbar,-5)
1690 PRINT "UVWbar = ",PROUNDCUVWbar,-5)
1700 PRINT
1710 IF Hist$="Y" THEN GOSUB Histogram
1720 F1no=F1no+1
1730 GOTO 700
1740 Ue=Max_u-Min_u
1750 Plotr$="Y"
1760 INPUT " Do you wish to see the turbulence quantities (Y/N, Default Y) ?",Plot$ 
1770 IF Plotr$="Y" THEN GOSUB Plot
1780 MASS STORAGE IS ":H8,0,1"
1790 PRINTER IS 16
1800 END
1810 !
1820 ! **************PLOT DATA*****************************
1830 !
1840 Plot: PRINTER IS 16
1850 PRINT "Estimate of normalizing velocity is :";Ue
1860 INPUT "Enter correct normalizing velocity if different.",Ue
1870 Graph$="Y"
1880 INPUT "Do you wish to graph the turbulence quantities (Y/N, Default Y) ?", Graph$
1890 PRINTER IS 0
1900 PRINT "Normalizing velocity is :";Ue
1910 ! 1920 FOR I=1 TO 4
1930 DATA 1,2,2,3
1940 READ Exp
1950 Kn=3
1960 IF I=4 THEN Kn=5
1970 FOR J=1 TO Nf-File1+1
1980 FOR K=1 TO Kn
1990 Index=(I-1)*3+K+1
2000 Point(J,Index)=Point(J,Index)/Ue^Exp
2010 IF Index>10 THEN Point(J,Index)=Point(J,Index)*1000
2020 NEXT K
2030 NEXT J
2040 NEXT I
2050 IF Graph$="N" THEN GOTO 2860
2060 ! 2070 PLOTTER IS 13,"GRAPHICS"
2080 GRAPHICS
2090 DEG
2100 ! 2110 Ymax=Point(1,1)
2120 Ymin=Ymax
2130 FOR J=1 TO Nf-File1+1
2140 IF Point(J,1)>Ymax THEN Ymax=Point(J,1)
2150 IF Point(J,1)<Ymin THEN Ymin=Point(J,1)
2160 NEXT J
2170 ! 2180 FOR I=1 TO 3
2190 LIMIT 0,184.47,0,149.8
2200 LOCATE Ll(I),Rl(I),39,99
2210 CLIP Ll(I),Rl(I),39,99
2220 ! 2230 Xmax=Point(1,(I-1)*3+2)
2240 Xmin=Xmax
2250 FOR J=1 TO Nf-File1+1
2260 FOR K=1 TO 3
2270 IF Point(J,(I-1)*3+K+1)>Xmax THEN Xmax=Point(J,(I-1)*3+K+1)
2280 IF Point(J,(I-1)*3+K+1)<Xmin THEN Xmin=Point(J,(I-1)*3+K+1)
2290 NEXT K
2300 NEXT J
2310 ! 2320 SCALE Llim(I),Ulim(I),Ymin,Ymax
2330 FRAME
2340 AXES Scale(I),1,0,0
2350 UNCLIP
2360 ! 2370 LROR &
2380 CSIZE 2.5
2390 Ypos=Ymin
2400 MOVE Llim(I)+Scale(I)/5,Ypos
2410 DRAW Llim(I),Ypos
2420 SETGU
2430 RPL0T 2,0,-2
2440 SETUU
2450 LABEL DROUND(Ypos,2)
2460 Ypos=Ypos+.1
2470 IF Ypos<=Ymax THEN GOTO 2400
2480 !
2490 LORG 6
2500 Xpos=Llim(I)
2510 MOVE Xpos,Ymin-.1
2520 LABEL DROUNDCXpos,2)
2530 MOVE Xpos,Ymin
2540 DRAW Xpos,Ymin+.1
2550 Xpos=Xpos+Scale(I)
2560 IF Xpos<ULim(I) THEN GOTO 2510
2570 !
2580 FOR J=1 TO Nf-File1+1.
2590 Ypos=Point(J,1)
2600 FOR K=1 TO 3
2610 MOVE Point(J,(I-1)*3+K+1),Ypos
2620 SETGU
2630 ON K GOTO 2641,271313,27613
2641 FOR Arc=1 TO 36B STEP 2
2651 PDIR Arc
2661 RPLOT .5,13
2671 NEXT Arc
2681 PDIR 0
2691 GOTO 2810
2701 RPLOT .5,.5,-2
2711 RPLOT .5,-.5,-1
2721 RPLOT -.5,-.5,-1
2731 RPLOT -.5,.5,-1
2741 RPLOT .5,.5,-1
2751 GOTO 2810
2761 RPLOT 0,.5,-2
2771 RPLOT 0,-.5,-1
2781 RPLOT .5,0,-2
2791 RPLOT -.5,0,-1
2801 GOTO 2810
2811 SETUU
2820 NEXT K
2830 NEXT J
2840 NEXT I
2850 DUMP GRAPHICS
,"uw/Uo3",/
2870 PRINT USING 2860
2880 FOR I=1 TO NF-File1+1
2890 IMAGE MDD.DD,6X,MD.DDDD,2X,MD.DDDD,2X,MD.DDDD,4X,MD.DDDD,2X,MD.DDDD,2X,MD.
DDDD
2900 PRINT USING 2890;Point(I,1),Point(I,5),Point(I,6),Point(I,7),Point(I,8),Point(I,9),Point(I,10)
2910 NEXT I
2930 PRINT USING 2920
2940 FOR I=1 TO NF-File1+1
2950 IMAGE MDD.DD,6X,MD.5D,2X,MD.5D,4X,MD.5D,2X,MD.5D,4X,MD.5D
2960 PRINT USING 2950;Point(I,1),Point(I,11),Point(I,12),Point(I,13),Point(I,14)
2970 ,Point(I,15)

47
2970 NEXT I
2980 !
2990 ! WRITE SUMMARY DATA FILE
3000 !
3010 Sum$="N"
3020 INPUT "Do you wish to write a Summary Data File (Y/N Default N) ?", Sum$
3030 IF Sum$="N" THEN GOTO 3210
3040 File$=Name$
3050 DISP "File ":File$"; "being written to disk"
3060 MASS STORAGE IS ":H8,0,0"
3070 CREATE File$,40
3080 ASSIGN File$ TO #1
3090 PRINT #1;Date$
3100 PRINT #1;Titl$
3110 PRINT #1;Name$
3120 PRINT #1;In
3130 PRINT #1;Nub, Numix1, Numix2, Numix3, Theta, Run
3140 PRINT #1;Vref, Ue, Df1, Df2, Df3
3150 PRINT #1;Ns, File1, NF
3160 FOR I=1 TO NF-N*File+1
3170 FOR J=1 TO 15
3180 PRINT #1;Point(I,J)
3190 NEXT J
3200 NEXT I
3210 RETURN
3220 !
3230 ! ************DRAW HISTOGRAMS************
3240 !
3250 Histogram: PLOTTER IS 13,"GRAPHICS"
3260 GRAPHICS
3270 K=1
3280 FOR I=1 TO 3
3290 Cmax=0
3300 FOR J=1 TO 1023 STEP 2
3310 Countbin(I,J)=Countbin(I,J)+Countbin(I,J+1)
3320 IF Countbin(I,J)>Cmax THEN Cmax=Countbin(I,J)
3330 NEXT J
3340 IF Cmax=0 OR Cmax=2000) THEN GOTO 3660
3350 LIMIT 0,184.47,0,149.8
3360 LOCATE 7.05,119.63,L11(K),U11(K)
3370 CLIP 7.05,119.63,L11(K),U11(K)
3380 K=K+1
3390 SCALE 1,1024,0,Cmax
3400 FRAME
3410 UNCLIP
3420 LINE TYPE 1
3430 CSIZE 2.8
3440 LORG 8
3450 FOR Y=1 TO 4
3460 Ypos=Y*Cmax/4
3470 PLOT 10,Ypos,-2
3480 PLOT -9,Ypos,-1
3490 MOVE 3,Ypos
3500 LABEL PROUNDC(Ypos,0)
3510 NEXT Y
3520 LORG 6
3530 Ypos=Cmax/20
3540 FOR X=1 TO 8
3550 Xpos=X*128
3560 PLOT Xpos,Ypos,-2

48
3570 PLOT Xpos,-Ypos,-1
3580 MOVE Xpos,-Ypos+1.3
3590 LABEL Xpos
3600 NEXT X
3610 FOR J=1 TO 1023 STEP 2
3620 IF Countbin[I,J]=0 THEN GOTO 3650
3630 MOVE J,0
3640 DRAW J,Countbin[I,J]
3650 NEXT J
3660 NEXT I
3670 MOVE 512,-Ypos*5
3680 LORG 6
3690 LABEL "PLOT OF COUNT VS NUMBER OF SAMPLES PER COUNT"
3700 DUMP GRAPHICS -Ypos*10
3710 RETURN
3720 !
3730 !
3740 ISOURCE
3750 ISOURCE ! NAM Find_vel
3760 ISOURCE ! This subroutine converts raw data counts to instantaneous velocities, then sums several different products of the velocity components. All input and output data is passed through the COMMON storage area. The inputs are the raw data array (Array), the Bragg shift frequency (Nub), the mixing frequencies (Numix1, Numix2, Numix3), the fringe spacings (Df1, Df2, Df3), the crossing angle of the third beam (Theta), and the number of samples in a data point (Ns).
3790 ISOURCE !
3800 ISOURCE EXT Get_value ! Declare subroutines stored outside of the main program.
3810 ISOURCE EXT Get_info ! outside of the main program.
3820 ISOURCE EXT Get_element
3830 ISOURCE EXT Int_to_rel
3840 ISOURCE EXT Rel_to_int
3850 ISOURCE EXT Rel_math
3860 ISOURCE EXT Put_value
3870 ISOURCE !
3880 ISOURCE Data_par: INT (*) ! Declare common variables.
3890 ISOURCE Ns_par: INT
4000 ISOURCE Ngs_par: INT
4010 ISOURCE Cbin_par: INT (*)
4020 ISOURCE Ran1_par: INT
4030 ISOURCE Ran2_par: INT
4040 ISOURCE Ran3_par: INT
4050 ISOURCE Sdev_par: REL
4060 ISOURCE Df1_par: REL
4070 ISOURCE Df2_par: REL
4080 ISOURCE Df3_par: REL
4090 ISOURCE Theta_par: REL
4100 ISOURCE Nub_par: REL
4110 ISOURCE Numix1_par: REL
4120 ISOURCE Numix2_par: REL
4130 ISOURCE Numix3_par: REL
4140 ISOURCE Su_par: REL
4150 ISOURCE Su_par: REL
4160 ISOURCE Su_par: REL
Reserve space for data array descriptor.

Reserve space for lookup tables for count to velocity conversion.

Reserve space for table of counts used to generate histograms.

Reserve space for various input and output variables.

Count and I are general purpose index variables. Count is usually 0, 1, or 2 to denote whether \( U \), \( V \), or \( W \) is being calculated.

Int, Address, and Offset are all general purpose storage areas.

R1, R2, and R3 are the count-to-frequency conversion factors.

Xvar and Yvar are general purpose real number storage areas.

\( U \), \( V \), and \( W \) are the instantaneous velocity components.
Cos and Sin are the cos and sin of Theta.

! Cos and Sin are the cos and sin of Theta.

The loop headed by Get_freq is repeated three times to get

the count-to-frequency conversion factors (which depend on

the range) for U, V, and W. Whenever a loop is controlled by
the variable "Count", the loop contains operations which are
the same for U, V, and W.

LDA = 0
STA Count

LDA Count ! Get the first word of the column of
LDB Ns ! the data array which contains the
MPY ! velocity component for which we want
STA Elementd ! to get the range.
LDA = Int
LDB = Arrayd
JSM Get_element
LDA Int ! Mask and rotate to get the four
LDB = 15360 ! bits containing the range.
AND B
SAR 10
TCA ! Subtract from 15 to get the
LDB = 15 ! actual range.
ADA B
LDB = Range1 ! Store the actual range in Range1
RDB Count ! so that we can transfer it to
STA B, I ! main program
LDB = 1
SZA Loopend

Loop: ! Use the range to find the power
SBL 1 ! of two needed for the divisor.
JSM Loop
Loopend: ! Convert the power of two into
LDA = Int ! a real number.
STA Oper_1
LDA = Yvar
STA Result
JSM Int_to_rel
LDA == 3.2E4 ! Divide 3.2E4 by the appropriate
LDB = Yvar ! power of two, using BCD math.
XFR 4
STB Oper_1
LDA = Yvar
STA Oper_2
LDA Count ! Decide whether to put the result
STA Result
SAL 2 ! in R1, R2, or R3, depending on Count.
ADA = R1
STA Result
LDA = 2
LDB = 147155B ! Now, finally, call the utility to
JSM Rel_math ! perform the division.
ISZ Count
LDA = 3 ! Increment and check Count so as
CPA Count ! to follow the loop three times.
JMP ++2
JMP Get_freq

LDA = Array1 ! Zero out the entire count-to-
LDB = 768 ! velocity conversion table so that
Continue: ! it must be recalculated for each
CLR 16 ! point. (This must be done if the
ADA = 16 ! mixing frequency or ranges are
DSZ B ! changed between counts.)
JMP Continue

JMP Get_freq
5970 ISOURCE LDA =Su  ! Set initial values of Su, Sv, Sw, Suu, etc. to zero.
5980 ISOURCE LDB =15
5990 ISOURCE Clear: CLR 4
6000 ISOURCE ADA =4
6010 ISOURCE DSZ B
6020 ISOURCE JMP Clear
6030 ISOURCE !
6040 ISOURCE LDA =Bin_u  ! Clear the areas of memory which
6050 ISOURCE LDB =192  ! will be used to hold the number of
6060 ISOURCE Clear2: CLR 16  ! samples per count for the histograms.
6070 ISOURCE ADA =16
6080 ISOURCE DSZ B
6090 ISOURCE JMP Clear2
6100 ISOURCE !
6110 ISOURCE LDA =Theta  ! Convert Theta from degrees to
6120 ISOURCE STR Oper_1  ! radians using the Rel_math
6130 ISOURCE LDA =Rad  ! utility.
6140 ISOURCE STR Oper_2
6150 ISOURCE LDA =Xvar
6160 ISOURCE STA Result
6170 ISOURCE LDA =2
6180 ISOURCE LDB =147155B
6190 ISOURCE JSN Rel_math
6200 ISOURCE !
6210 ISOURCE LDA =Xvar  ! Find the sine and cosine of
6220 ISOURCE STR Oper_1  ! Theta, and store them in the
6230 ISOURCE LDA =Sin  ! locations Sin and Cos, respectively.
6240 ISOURCE STA Result
6250 ISOURCE LDA =1
6260 ISOURCE LDB =34213B
6270 ISOURCE JSN Rel_math
6280 ISOURCE LDA =Cos
6290 ISOURCE STA Result
6300 ISOURCE LDA =1
6310 ISOURCE LDB =34224B
6320 ISOURCE JSN Rel_math
6330 ISOURCE !
6340 ISOURCE ! The loop defined by Get_int calculates intermediate values
6350 ISOURCE ! used in converting counts to velocities. These values are the
6360 ISOURCE ! same for all samples in a point, so they can be calculated
6370 ISOURCE ! separately. The loop calculates (Nub-NumixN) and (Mill*DfN),
6380 ISOURCE ! where N is 1, 2, and 3.
6390 ISOURCE LDA =3
6400 ISOURCE STR Count
6410 ISOURCE Get_int: LDA Count
6420 ISOURCE ADA =-1
6430 ISOURCE SAL 2
6440 ISOURCE STR Offset
6450 ISOURCE LDA =Nub  ! Find NumixN=Nub-NumixN.
6460 ISOURCE STR Oper_1
6470 ISOURCE LDA =Numix1
6480 ISOURCE ADA Offset
6490 ISOURCE STR Oper_2
6500 ISOURCE STA Result
6510 ISOURCE LDA =2
6520 ISOURCE LDB =1467173
6530 ISOURCE JSN Rel_math
6540 ISOURCE LDA =DfI  ! Find DfN=Mll*DfN.
6550 ISOURCE ADA Offset
6560 ISOURCE STR Oper_1

53
This section filters the data to remove counts which are excessively far from the mean. First the section creates a list of the number of times each particular count appears. The list consists of 3 1024 word arrays (called Bin_u, Bin_v, and Bin_w) where the address of each word corresponds to the count, and the value of the word indicates the number of times that particular count has appeared.

This section filters the data to remove counts which are excessively far from the mean. First the section creates a list of the number of times each particular count appears. The list consists of 3 1024 word arrays (called Bin_u, Bin_v, and Bin_w) where the address of each word corresponds to the count, and the value of the word indicates the number of times that particular count has appeared.
The mean and standard deviation of the data are calculated. The program then throws out all the data which are farther than Sdev standard deviations from the mean.

Check dev: LDB =Sdev

LDA B,I ! If Sdev > 0, use this section to filter the data. If not, continue with the program.
RZA ++2
JMP Set_begin

LDA =0 ! Get the average of each set of counts using the information in the Bin_u arrays.
STA Count ! counts using the information in
LDA =Zero ! the Bin_u arrays.

LDB Count ! First, set the average=0.

Set_begin:
LDA =13
STA Count

Get_avg:
LDA =Zero

Get the average of each set of counts using the information in the Bin_u arrays.
First, set the average=0.

LDA =13
STA Oper_1

Get a count and convert it to a real number.

LDA =Xvar
STA Oper_1

LDA Address ! Use the Bin_u arrays to find out how many samples there are with that count and convert that number to a real number.
SBL 2

Sum_count: LDA =I

Get a count and convert it to a real number.

LDA =Xvar
STA Oper_1

Multiply the count by the number of times it appears in the data.

LDA =Yvar
STA Oper_2

LDA =2

LDB =147037B

Keep a running sum of the product of the count times the number of times it appears. Store this sum in the place we will eventually use to store the average.

LDB Count

JMP Sum_count
Convert the number of samples to a real number.

Divide the sum stored in Avg_u by the number of samples to get the average value of the counts.

Get the standard deviation of each set of counts using the average and the information in the Bin_u arrays. First, set the standard deviation = 0.

Get a count and convert it to a real number.

Subtract the average count from the count we just got.
Square the difference between the average and the count we got to get the deviation from the mean.

Use the Bin_u arrays to find out how many counts there are with the value we have chosen, and convert that number to a real number.

Multiply the number of counts by the deviation from the mean.

Add the product to Dev_u, which currently contains a running sum of the number of counts times the deviation from the mean of each count.

Convert the number of samples to a real number.

Divide the sum stored in Dev_u by the number of samples to get the variance of the counts.

Take the square root of the variance to get the standard deviation.
8970 ISOURCE LDB =31458B
8980 ISOURCE JSM Rel_math
8990 ISOURCE !
9000 ISOURCE LDA Count
9010 ISOURCE ADA =1
9020 ISOURCE STA Count
9030 ISOURCE ADA =-3
9040 ISOURCE SZA ++2
9050 ISOURCE JMP Get_sdev
9060 ISOURCE !
9070 ISOURCE LDA =0 ! Now use the standard deviation to
9080 ISOURCE STA Count ! filter out all the counts whose
9090 ISOURCE Dev_filtr: LDA Count ! value is more than Sdev standard
9100 ISOURCE ADA =1 ! deviations away from the mean.
9110 ISOURCE LDB =1024
9120 ISOURCE MPY
9130 ISOURCE ADA =Bin_u
9140 ISOURCE STA Address
9150 ISOURCE LDA =1023
9160 ISOURCE STA 1
9170 ISOURCE !
9180 ISOURCE LDA Count
9190 ISOURCE SAL 2
9200 ISOURCE ADA =Dev_u
9210 ISOURCE STA Oper_1
9220 ISOURCE LDA =Sdev
9230 ISOURCE STA Oper_2
9240 ISOURCE LDA =Xvar
9250 ISOURCE STA Result
9260 ISOURCE LDA =2
9270 ISOURCE LDB =147037B
9280 ISOURCE JSM Rel_math
9290 ISOURCE !
9300 ISOURCE LDA =Xvar
9310 ISOURCE STA Oper_1
9320 ISOURCE LDA =Int
9330 ISOURCE STA Result
9340 ISOURCE JSM Rel_to_int
9350 ISOURCE !
9360 ISOURCE LDA Count
9370 ISOURCE SAL 2
9380 ISOURCE ADA =Avg_u
9390 ISOURCE STA Oper_1
9400 ISOURCE LDA =Offset
9410 ISOURCE STA Result
9420 ISOURCE JSM Rel_to_int
9430 ISOURCE !
9440 ISOURCE LDB Offset
9450 ISOURCE TCB
9460 ISOURCE Filtr_dev: LDA I
9470 ISOURCE ADA B
9480 ISOURCE SAM ++2
9490 ISOURCE TCA
9500 ISOURCE ADA Int
9510 ISOURCE SAP ++3
9520 ISOURCE LDA Address
9530 ISOURCE CLR 1
9540 ISOURCE DSZ Address
9550 ISOURCE DSZ I
9560 ISOURCE JMP Filtr_dev

58
Go through this whole process of checking the deviation and filtering the data three times, once for each channel.

The loop Begin is performed three times, once for U, V, and W. Each time through the data array is read, a count (a raw datum) is taken from it and converted into a velocity. The velocity is stored in U, V, or W depending on whether this is the first, second, or third iteration of the loop. The first time a particular count is encountered, the velocity corresponding to it is calculated using the intermediate values found in Get_int, and the velocity is stored in a table. If that count is found again in the data array, the corresponding velocity is looked up rather than being calculated again.

Now look in the appropriate part of the arrays created by the filter section. If the word corresponding to the count is zero, the count was filtered out and should be ignored. Ignore these three counts, go back and do the next three.
Now, use Count to find out which lookup table array we want to use, and use the count we got from the data array to find exactly where we want to go.

If that table entry is zero, calculate a velocity for it.

If we are calculating U, reverse the sign of \( (|\text{NumixN}| - \text{FrequencyN}) \) so as to reverse the sign of U. Leave V and \( \text{W}_v \) alone.

Find Velocity = \( (|\text{NumixN}| - \text{FrequencyN}) \times (\text{Mill} \times \text{DfN}) \) and store in a place in the lookup table corresponding to
10770 ISOURCE STA Oper_2 ! the data count.
10780 ISOURCE LDA Address
10790 ISOURCE ADA =-1
10800 ISOURCE STA Result
10810 ISOURCE LDA =2
10820 ISOURCE LDB =147037B
10830 ISOURCE JSM Rel_math
10840 ISOURCE !
10850 ISOURCE Over: LDA Address ! Transfer the velocity from
10860 ISOURCE ADA =-1 ! the lookup table to U, V, or W,
10870 ISOURCE LDB =U ! as appropriate.
10880 ISOURCE ADD Offset
10890 ISOURCE XFR 4
10900 ISOURCE !
10910 ISOURCE ISZ Count
10920 ISOURCE LDA =-3 ! Have U, V, and W all
10930 ISOURCE ADA Count ! been calculated?
10940 ISOURCE SZA ++2 ! If not, go back again.
10950 ISOURCE JMP Begin
10960 ISOURCE STA Count ! If so, set Count = 0.
10970 ISOURCE !
10980 ISOURCE ! Now we convert the W we have obtained (which is measured at an
10990 ISOURCE ! angle Theta to the V-axis) to the W we want (which should
11000 ISOURCE ! be measured at an angle of 90 degrees to the V-axis).
11010 ISOURCE ! Thus find W=(Wv-V*Cos(Theta))/Sin(Theta).
11020 ISOURCE !
11030 ISOURCE LDA =V
11040 ISOURCE STA Oper_1
11050 ISOURCE LDA =Cos
11060 ISOURCE STA Oper_2
11070 ISOURCE LDA =Yvar
11080 ISOURCE STA Result
11090 ISOURCE LDA =2
11100 ISOURCE LDB =147037B
11110 ISOURCE JSM Rel_math
11120 ISOURCE LDA Result
11130 ISOURCE STA Oper_2
11140 ISOURCE LDA =W
11150 ISOURCE STA Oper_1
11160 ISOURCE LDA =Yvar
11170 ISOURCE STA Result
11180 ISOURCE LDA =2
11190 ISOURCE LDB =146717B
11200 ISOURCE JSM Rel_math
11210 ISOURCE LDA Result
11220 ISOURCE STA Oper_1
11230 ISOURCE LDA =Sin
11240 ISOURCE STA Oper_2
11250 ISOURCE LDA =W
11260 ISOURCE STA Result
11270 ISOURCE LDA =2
11280 ISOURCE LDB =147155B
11290 ISOURCE JSM Rel_math
11300 ISOURCE !
11310 ISOURCE ! Now take running sums of U, V, W, and several products
11320 ISOURCE ! of these velocities. The sums are taken using the utility
11330 ISOURCE ! "Add". The sums are calculated in an unusual sequence in
11340 ISOURCE ! order to reduce the number of program steps needed to cal-
11350 ISOURCE ! culate them.
11360 ISOURCE !
11970 ISOURCE LDA =Xvar
11980 ISOURCE STA Result
11990 ISOURCE LDA =2
12000 ISOURCE LDB =147037H
12010 ISOURCE JSM Rel_math -
12020 ISOURCE 
12030 ISOURCE LDA =Suuv ! Find Suuv=Suuv+(U*Y*V)
12040 ISOURCE LDB Result
12050 ISOURCE JSM Add
12060 ISOURCE 
12070 ISOURCE LDA =Yv ! Find U*Y*W
12080 ISOURCE STA Oper_1
12090 ISOURCE LDA =W
12100 ISOURCE STA Oper_2
12110 ISOURCE LDA =Xvar
12120 ISOURCE STA Result
12130 ISOURCE LDA =2
12140 ISOURCE LDB =147037H
12150 ISOURCE JSM Rel_math -
12160 ISOURCE 
12170 ISOURCE LDA =Suuv ! Find Suuv=Suuv+(U*Y*W)
12180 ISOURCE LDB Result
12190 ISOURCE JSM Add
12200 ISOURCE 
12210 ISOURCE LDA =V ! Find V*V
12220 ISOURCE STA Oper_1
12230 ISOURCE STA Oper_2
12240 ISOURCE LDA =Xvar
12250 ISOURCE STA Result
12260 ISOURCE LDA =2
12270 ISOURCE LDB =147037H
12280 ISOURCE JSM Rel_math -
12290 ISOURCE 
12300 ISOURCE LDA =Suu ! Find Suv=Suu+(Y*V)
12310 ISOURCE LDB Result
12320 ISOURCE JSM Add
12330 ISOURCE 
12340 ISOURCE LDA =Sw ! Find Sw=Sw+W
12350 ISOURCE LDB =W
12360 ISOURCE JSM Add
12370 ISOURCE 
12380 ISOURCE LDA =U ! Find U*W
12390 ISOURCE STA Oper_1
12400 ISOURCE LDA =Xvar
12410 ISOURCE STA Result
12420 ISOURCE LDA =2
12430 ISOURCE LDB =147037H
12440 ISOURCE JSM Rel_math -
12450 ISOURCE 
12460 ISOURCE LDA =Suw ! Find Suw=Suw+(U*W)
12470 ISOURCE LDB Result
12480 ISOURCE JSM Add
12490 ISOURCE 
12500 ISOURCE LDA =W ! Find Y*W
12510 ISOURCE STA Oper_1
12520 ISOURCE LDA =V
12530 ISOURCE STA Oper_2
12540 ISOURCE LDA =Xvar
12550 ISOURCE STA Result
12560 ISOURCE LDA =2

63
12570  ISOURCE          LDB  =147037B
12580  ISOURCE          JSM  Rel_math
12590  ISOURCE          !  Find Suw=Suw+(V*W)
12600  ISOURCE          LDA  =Suw
12610  ISOURCE          LDB  Result
12620  ISOURCE          JSM  Add
12630  ISOURCE          !  Find W*W
12640  ISOURCE          LDA  =W
12650  ISOURCE          STA  Oper_1
12660  ISOURCE          STA  Oper_2
12670  ISOURCE          LDA  =Xvar
12680  ISOURCE          STA  Result
12690  ISOURCE          LDA  =2
12700  ISOURCE          LDB  =147037B
12710  ISOURCE          JSM  Rel_math
12720  ISOURCE          !  Find Suw=Suw+(W*W)
12730  ISOURCE          LDA  =Suw
12740  ISOURCE          LDB  Result
12750  ISOURCE          JSM  Add
12760  ISOURCE          !  Find U*W*W
12770  ISOURCE          LDA  =U
12780  ISOURCE          STA  Oper_1
12790  ISOURCE          LDA  =Uu
12800  ISOURCE          STA  Result
12810  ISOURCE          LDA  =2
12820  ISOURCE          LDB  =147037B
12830  ISOURCE          JSM  Rel_math
12840  ISOURCE          !  Find Suww=Suww+(U*W*W)
12850  ISOURCE          LDA  =Suww
12860  ISOURCE          LDB  Result
12870  ISOURCE          JSM  Add
12880  ISOURCE          !  Continue to calculate running
12890  ISOURCE          DSZ  I
12900  ISOURCE          !  sums until out of samples.
12910  ISOURCE          !  Now place the finished sums in the COMMON region so that
12920  ISOURCE          !  the BASIC program has access to them, and then return to
12930  ISOURCE          !  the BASIC program.
12940  ISOURCE          !  Replace:
12950  ISOURCE          LDA  =Range1
12960  ISOURCE          LDB  =Ran1_par
12970  ISOURCE          JSM  Put_value
12980  ISOURCE          LDA  =Range2
12990  ISOURCE          LDB  =Ran2_par
13000  ISOURCE          JSM  Put_value
13010  ISOURCE          LDA  =Range3
13020  ISOURCE          LDB  =Ran3_par
13030  ISOURCE          JSM  Put_value
13040  ISOURCE          LDA  =Su
13050  ISOURCE          LDB  =Su_par
13060  ISOURCE          JSM  Put_value
13070  ISOURCE          LDA  =Su
13080  ISOURCE          LDB  =Su_par
13090  ISOURCE          JSM  Put_value
13100  ISOURCE          LDA  =Su
13110  ISOURCE          LDB  =Su_par
13120  ISOURCE          JSM  Put_value
13130  ISOURCE          LDA  =Suw
13140  ISOURCE          LDB  =Suw_par
13150  ISOURCE          JSM  Put_value
13160  ISOURCE          LDA  =Svv

64
LDB =Suv_par
J5M Put value
LDA =Suv
J5M Put value
LDA =Suu
LDB =Suuv_par
J8M Put value
LDA =Suvw
LDB =Suvw_par
J8M Put value
LDA =Suvw
LDB =Suvw_par
J8M Put value
LDA =Suvw
LDB =Suvw_par
J8M Put value
LDA =Suvw
LDB =Suvw_par
J8M Put value
LDA =Suvw
LDB =Suvw_par
LDA =Ngs
LDB =Ngs_par
J8M Put value
LDA =Bin
Transfer the contents of Bin_n
to the common area without using
the slow HP-supplied external
subroutines.

LDA =Bin_u
Transfer: XFR 16
ADDA Count
ADB Count
DSZ I
JMP Transfer
RET 1
LIT 200
! The utility "Add" is used to add up the running sums.

Add:
ISZ Ut1count
STB Oper_1
STB Oper_2
STB Result
LDA =2
LDB =146721B
JSM Rel_math
DSZ Ut1count
RET 1
JSM Ut1end
ACKNOWLEDGEMENTS

We are grateful to Dr. Oktay Ozcan for help in the initial design of the system and to Dr. Dennis Johnson for many helpful comments. This work was supported by the Fluid Dynamics Research Branch, NASA Ames Research Center under Grant NCC-2-294.
REFERENCES


5. OPERATOR'S MANUAL FOR NASA 3D-LDV COMPUTER INTERFACE, NASA Ames Research Center, Writing Assoc., Sunnyvale, CA, July 1, 1982.

# TABLE 1

## MACRODYNE RANGE SETTINGS

<table>
<thead>
<tr>
<th>MANTISA</th>
<th>1</th>
<th>500</th>
<th>1024</th>
</tr>
</thead>
<tbody>
<tr>
<td>RANGE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>32  GHz</td>
<td>64 MHz</td>
<td>31.28 MHz</td>
</tr>
<tr>
<td>1</td>
<td>16  &quot;</td>
<td>32  &quot;</td>
<td>15.64 &quot;</td>
</tr>
<tr>
<td>2</td>
<td>8   &quot;</td>
<td>16  &quot;</td>
<td>7.20  &quot;</td>
</tr>
<tr>
<td>3</td>
<td>4   &quot;</td>
<td>8   &quot;</td>
<td>3.91  &quot;</td>
</tr>
<tr>
<td>4</td>
<td>2   &quot;</td>
<td>4   &quot;</td>
<td>1.95  &quot;</td>
</tr>
<tr>
<td>5</td>
<td>1   &quot;</td>
<td>2   &quot;</td>
<td>977 kHz</td>
</tr>
<tr>
<td>6</td>
<td>500 MHz</td>
<td>1 &quot;</td>
<td>488 &quot;</td>
</tr>
<tr>
<td>7</td>
<td>250 &quot;</td>
<td>500 kHz</td>
<td>244 &quot;</td>
</tr>
<tr>
<td>8</td>
<td>125 &quot;</td>
<td>250 &quot;</td>
<td>122 &quot;</td>
</tr>
<tr>
<td>9</td>
<td>62.5 &quot;</td>
<td>125 &quot;</td>
<td>61 &quot;</td>
</tr>
<tr>
<td>10</td>
<td>31.25 &quot;</td>
<td>62.5 &quot;</td>
<td>30 &quot;</td>
</tr>
<tr>
<td>11</td>
<td>15.62 &quot;</td>
<td>31.25 &quot;</td>
<td>15 &quot;</td>
</tr>
<tr>
<td>12</td>
<td>7.81 &quot;</td>
<td>15.62 &quot;</td>
<td>7.6 &quot;</td>
</tr>
<tr>
<td>13</td>
<td>3.90 &quot;</td>
<td>7.81 &quot;</td>
<td>3.8 &quot;</td>
</tr>
<tr>
<td>14</td>
<td>1.95 &quot;</td>
<td>3.90 &quot;</td>
<td>1.9 &quot;</td>
</tr>
</tbody>
</table>
Fig. 1  Schematic of the optics table layout.
Fig. 2

Schematic of the transmitting optics.

- Mirror
- Polarization Rotator
- Dichroic Filter (Splits Green, Line)
- Four Beam Matrix from Optics Table
- Lens $f = 15$ in.
- Measuring Volume
MAIN AND OFF-AXIS ELLIPSOIDS HAVE SAME DIMENSIONS

Fig. 3 Details of the probe volumes in the 3-D system.
Fig. 4  Schematic of the receiving optics.
Fig. 5 Evaluation of the effective probe length.

Fig. 6 Schematic of the signal processing electronics.
Fig. 7 Computer interface simplified block diagram.
Fig. 8 Block diagram for data acquisition program.
**OPEN DATA FILE**

- LOCATION X, Y, Z
- CALIBRATION CONSTANTS
- RAW DATA A(i,j) i=1, Ns, j=1,3

**READ DATA FILE**

- LOCATION X, Y, Z
- CALIBRATION CONSTANTS
- RAW DATA A(i,j) i=1, Ns, j=1,3

**CONSTRUCT LOOK-UP TABLE**

- FREQ = 3.2 X 10^4/COUNT X 2^{RANGE}
- VELOCITY = [FREQ - (BRAGG-MIX)] * FRINGE SPACING

**CALCULATE VELOCITIES FROM RAW DATA AND LOOK-UP TABLE**

**COMPUTE SIGNAL MOMENTS**

\[
\begin{align*}
\frac{1}{Ns} \sum u(i), \frac{1}{Ns} \sum v(i), \frac{1}{Ns} \sum w(i), \frac{1}{Ns} \sum u^2(i), \frac{1}{Ns} \sum v^2(i), \frac{1}{Ns} \sum w^2(i), \frac{1}{Ns} \sum u^2(i)v(i), \frac{1}{Ns} \sum u^2(i)w(i), \frac{1}{Ns} \sum v^2(i)w(i), \frac{1}{Ns} \sum u(i)v(i)w(i)
\end{align*}
\]

**STATISTICAL QUANTITIES FROM MOMENTS**

\[
\begin{align*}
\bar{u}, \bar{v}, \bar{w}, \bar{u}^2, \bar{v}^2, \bar{w}^2, \bar{u}v, \bar{u}w, \bar{v}w, \bar{u}^2v, \bar{u}^2w, \bar{v}^2w
\end{align*}
\]

**PLOT HISTOGRAMS**

PLOT NUMBER OF SAMPLES FOR EACH COUNT

**PRINT RESULTS**

**ANOTHER POINT?**

**PLOT PROFILES OF QUANTITIES**

\[
\begin{align*}
\bar{u}/U_0, \bar{v}/U_0, \bar{w}/U_0, \bar{u}^2/U_0^2, \bar{v}^2/U_0^2, \bar{w}^2/U_0^2, \bar{u}v/U_0^2, \bar{u}w/U_0^2, \bar{v}w/U_0^2, \bar{u}^2v/U_0^2, \bar{u}^2w/U_0^2, \bar{v}^2w/U_0^2
\end{align*}
\]

**STOP**

---

**Fig. 9** Block diagram for data reduction program.
DIGITAL MULTIplexER INPUTS 16 BIT WORDS

DIGITAL DATA FROM COUNTER

A/D DIGITAL OUTPUTS 12 BITS 2'S COMPLEMENT

OUTPUT TO PDP-11/DR-11B

DIGITAL OUTPUTS 12 BITS 2'S COMPLEMENT

OUTPUT TO HP9845 98032A

ANALOG INPUTS

Fig. 10 NASA LDV-A/D computer interface connections and settings.
Fig. 11 Experimental rig.
Fig. 12 Secondary velocity plots.
Fig. 13  Comparison of X-wire and LDV measurements in a vortex/mixing layer interaction.
End of Document