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 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 3-Component Laser-Doppler Velocimeter Data Acquisition and Reduction System

L. C. Rodman
J. K. Bell
R. D. Mehta

CONTRACT NCC-2-294
January 1986

NASA
A 3-Component Laser-Doppler Velocimeter Data Acquisition and Reduction System

L. C. Rodman
J. H. Bell
R. D. Mehta
Stanford University
Stanford, California

Prepared for
Ames Research Center
under Cooperative Agreement NCC-2-294
January 1986

NASA
National Aeronautics and
Space Administration
Ames Research Center
Moffett Field, California 94035
TABLE OF CONTENTS

ABSTRACT ......................................................... iii
LIST OF FIGURES ................................................... iv
NOMENCLATURE ..................................................... v
1. INTRODUCTION .................................................. 1
2. OPTICAL SYSTEM AND SIGNAL PROCESSING HARDWARE .......... 3
   2.1 Optics Table ................................................. 4
   2.2 Transmitting Optics .......................................... 5
   2.3 Receiving Optics ............................................. 6
   2.4 Signal Processing Hardware .................................. 6
   2.5 Computer Interface ........................................... 7
   2.6 HP 9845A Desk-Top Computer .............................. 8
3. DATA ACQUISITION AND REDUCTION SOFTWARE .................. 8
   3.1 Data Acquisition Software ................................... 9
   3.2 Data Reduction Software .................................... 10
4. OPERATING PROCEDURES ........................................ 15
   4.1 Alignment Procedures ....................................... 15
   4.2 Signal Processing ........................................... 17
5. INHERENT PROBLEMS ............................................ 19
6. SAMPLE RESULTS ................................................ 21
7. CONCLUDING REMARKS ......................................... 22
APPENDIX -- SOFTWARE FOR THE HP 9845B COMPUTER .......... 23
ACKNOWLEDGEMENTS ................................................. 67
REFERENCES ......................................................... 68
TABLE 1 ............................................................ 69
FIGURES ............................................................ 70
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.
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. 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 \( w_v = (v \cos 45° + w \sin 45°) \). 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 (Macrodyne models 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 Macrodyne. 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. Jumpers 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 0-13 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 10
\[ f_{\text{res}} = 3.2 \times 10^4 / 2^R \times D \]  

(2)

where \( D \) = raw data.

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

\[
\text{velocity} = d_r \ast \left( (f_{\text{bragg}} - f_{\text{mix}}) - f_{\text{res}} \right)
\]

(3)

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

\[
\begin{align*}
S_u &= S<u(i)> / Ns \\
S_v &= S<v(i)> / Ns \\
S_w &= S<w(i)> / Ns \\
S_{uu} &= S<u(i)u(i)> / Ns \\
S_{uv} &= S<u(i)v(i)> / Ns \\
S_{uw} &= S<u(i)w(i)> / Ns \\
S_{vv} &= S<v(i)v(i)> / Ns \\
S_{vw} &= S<v(i)w(i)> / Ns \\
S_{ww} &= S<w(i)w(i)> / Ns \\
S_{uvw} &= S<u(i)v(i)w(i)> / Ns \\
S_{uw} &= S<u(i)w(i)> / Ns \\
S_{vw} &= S<v(i)w(i)> / Ns \\
S_{uw} &= S<w(i)w(i)> / Ns \\
S_{uvw} &= S<u(i)v(i)w(i)> / Ns \\
\end{align*}
\]

(4)

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

\[
\begin{align*}
\bar{u} &= S_u \\
\bar{v} &= S_v \\
\bar{w} &= S_w \\
u^*^2 &= S_{uu} - S_u S_u \\
v^*^2 &= S_{vv} - S_v S_v
\end{align*}
\]

(5)
\[ w'^2 = Sww - SwSw \]
\[ u'v' = Suv - SuSV \]
\[ u'w' = Swu - SuSw \]
\[ v'w' = Sv - SwSV \]
\[ u'^2v' = Suuv - 2SuSV - SvSuu + 2SwSuSu \]
\[ u'^2w' = Suuw - 2SuSw - SwSuu + 2SwSuSu \]
\[ u'v'^2 = Suvv - 2SvSvv - SuSvv + 2SuSvSv \]
\[ u'w'^2 = Suww - 2Ssww - SuSw + 2SuSwSw \]
\[ u'w'v' = Suwv - SuSw - SvSw - SwSuv + 2SuSvSw \]

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,

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

becomes

\[ f_{\text{int}} = f_{\text{bragg}} - f_{\text{mix}} ; \quad d_{\text{int}} = d_f \ast 10^6 \]

hence,

\[ \text{velocity} = d_{\text{int}} \ast (f_{\text{int}} - f_{\text{res}}) \]

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 $w = (w_y - v \cos 45^\circ) / \sin 45^\circ$.

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.
Plots histograms of the raw data for all three channels.

Performs steps 1 through 11 for each profile point.

Plots profiles of the reduced data.

Tabulates normalized data profiles.

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

\[
C_j = \frac{\sum_{i=1}^{1024} i \times S_i}{N_s} \quad (10)
\]

where \( j = 1, 2, 3 \) = channel number

\( C_j \) = average count

\( N_s \) = number of samples.

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

\[
\sigma_j^2 = \frac{\sum_{i=1}^{1024} S_i (i - C_j)^2}{N_s} \quad (11)
\]
where $\sigma_j$ = 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 increased.

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 ($f_{res} = f_{bragg} - f_{mix} - f_{dopp}$). If $f_{mix}$ 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 $\Delta t(f_{bragg} - f_{mix})$, 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 $f_{mix}$ 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 (Macodyne) 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 Macodyne processor. An additional check can be made on the range setting by monitoring the analog output from the Macodyne 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: \( \text{words} = \#\text{inputs} + 2 \). 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 μs 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 $\overline{w^2}$ 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. $\overline{v^2}$ seems to agree very well whereas $\overline{u^2}$ 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 $w_\tau$ channels.) However, the normal stress measurements agree to within $10\%$, and the shear stresses are consequently affected. $\overline{u'v'}$ is somewhat low (about $20\%$) whereas $\overline{u'w'}$ is low by almost a factor of two. The measurement of $\overline{u'w'}$ with the present system seems to be very sensitive. This is due to the fact the $w_\tau * u$ is of the same order as $u * v \cos 45^\circ$, so any small error in the measurement of these velocities or the off-axis angle can result in large errors in $\overline{u'w'}$. $\overline{v'w'}$ in vortex affected flows is generally of the same order as $\overline{u'w'}$ and this seems to be the case with
the present measurements. With a cross-wire, $w'w'$ 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-JP 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
10 ! PROGRAM TO ACQUIRE DATA FROM THREE-COMPONENT LDV SYSTEM
20 ! The program asks for initialization data and calculates the
30 ! calibration constants from them. It reads 3 channels of raw
40 ! LV data from the LDV A/D CI and writes them to a disk file
50 ! together with the calibration constants. The program can reduce
60 ! the data and display real-time histograms of the raw LV data
70 ! if desired.
80 !
90 OPTION BASE 1
100 ICOM 16600
110 IDELETE ALL
120 IASSEMBLE Find_vel
130 IASSEMBLE Data_trans
140 IASSEMBLE Draw_hist
150 COM INTEGER Dats(3,2000),D1(2000,6),Ns,Nn 'D1 is data buffer
160 COM REAL DF1,DF2,DF3
170 COM REAL Theta,Hue,Num:1x:1,Num:2x2,Num:3x3
180 COM REAL Su,Sv,lw,Suu,Suv,Swu,Suw,Suv,Suv,Suw,Suw
190 DIM Date%(80),Files%(6),Names%(4),Titles%(80)
200 REAL Xpos,Ypos,Zpos,Pos(30,3)
210 INTEGER A,B,Ran,1,Ran,2,3
220 INTEGER Run,Dn,ns,H
230 REAL Phi,D,F,Phi2
240 REAL lam1,La:rm2,Lam:3,Db,Prwld,Prln:1
250 REAL Re,U
260 REAL Prwld2,Prwld3,Prlen:1,Prlen:2
270 REAL Nfr1,Nfr:2,4fr:3
280 REAL U,V,W
290 Ns=2000
300 !
310 PRINT
320 PRINT " ** << PROGRAM LDV : 3-COMPONENT VELOCITY DATA >> ** 
330 PRINT
340 PRINT "PROGRAM STRUCTURE"
350 PRINT "1. INITIALIZE VARIABLES AND CALCULATE PARAMETERS"
360 PRINT "2. ACQUIRE DATA FROM A/D"
370 PRINT "3. WRITE TO FLOPPY DISC"
380 PRINT
390 !
400 ! ** CHECK HISTOGRAMS **
410 !
420 Ans$="N"
430 INPUT "DO YOU WISH TO LOOK AT HISTOGRAMS ? (Y/N, DEFAULT N)"$,Ans$
440 IF Ans$="Y" THEN GOSUB Hist
450 !
460 ! ** INITIALIZE RUN **
470 !
480 PRINT " ** INITIALIZATION ** 
490 Run:=1
500 PRINT " ENTER RUN PARAMETERS:
510 PRINT
520 INPUT "Enter date and time:",Dates$
530 INPUT "Enter l-Line Name For Profile ("",Titles$
540 INPUT "No. of data samples per point (2000 maxm.) :",Ns
550 !
560 ! BEAM SPACING J IS FIXED
570 !
580 D=.013
590 !
600 ! FOCAL LENGTH F IS FIXED
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

Phi1=ATN(D/2/F)
Phi2=Phi1+2

CALCULATE FRINGE SPACINGS FROM WAVELENGTHS AND HALF-ANGLES

DF1=Lam1/2/SIN(Phi1)
DF2=Lam2/2/SIN(Phi1)
DF3=Lam3/2/SIN(Phi1)

BEAM DIAMETER Db IS FIXED

Db=1.28E-3

CALCULATE PP03E VOLUME WIDTH AND LENGTH

Pruid1=4*Lam1*F/PI/Db*COS(Phi1)
Prilen1=4*Lam1*F/PI/Db*SIN(Phi1)
Pruid2=4*Lam2*F/PI/Db*COS(Phi1)
Prilen2=4*Lam2*F/PI/Db*SIN(Phi1)
Pruid3=Pruid2
Prilen3=Prilen2

Nfr1=Pruid1/DF1
Nfr2=Pruid2/DF2
Nfr3=Nfr2

GET MORE RUN PARAMETERS

INPUT "Enter Bragg shift frequency (MHz) ": Nub

KII=3

INPUT "Is this a two-channel or three-channel run (2/3, default 3) ": KII

IF KII=3 THEN GOTO 1110

INPUT "Enter mixing frequency (MHz, 2 nos.) ": Num1x1, Num1x2
Num1x3=0
Theta=0
GOTO 1110

INPUT "Enter mixing frequency (MHz, 3 nos.) ": Num1x1, Num1x2, Num1x3

INPUT "Enter 3rd beam angle (deg.) ": Theta

INPUT "Enter tunnel reference voltage (volts) ": Vref

PRINTER IS 0

PRINT HEADER

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

PRINT \"BEAM SPACINGS (m) = ", D
1210 PRINT "TOTAL ANGLE BETWEEN BEAMS (Degs) =", Ph2
1220 PRINT "FRINGE SPACINGS (m) =", Df1, Df2, Df3
1230 PRINT "PROBE WIDTHS (m) =", Prw1, Prw2, Prw3
1240 PRINT "PROBE LENGTHS (m) =", Prl1, Prl2, Prl3
1250 PRINT "NO. OF FRINGES =", Nfr1, Nfr2, Nfr3
1260 PRINT "Bragg SHIFT FREQUENCY (MHz) =", Nub
1270 PRINT "MIXING FREQUENCY (MHz) =", NUMx1, NUMx2, NUMx3
1280 PRINT "THIRD BEAM SET ANGLE (Degs) =", Th3:
1290 PRINT "RUN NUMBER = ", Run
1300 PRINT "TUNNEL REFERENCE VOLTAGE (volts) =", Vref
1310 PRINT TOTA~ ANG~E BETWEEN BEAMS (Degs) =", Ph2
1320 PRINT "FRINGE SPACINGS (m) =", Df1, Df2, Df3
1330 PRINT "PROBE WIDTHS (m) =", Prw1, Prw2, Prw3
1340 PRINT "PROBE LENGTHS (m) =", Prl1, Prl2, Prl3
1350 PRINT "NO. OF FRINGES =", Nfr1, Nfr2, Nfr3
1360 PRINT "Bragg SHIFT FREQUENCY (MHz) =", Nub
1370 PRINT "MIXING FREQUENCY (MHz) =", NUMx1, NUMx2, NUMx3
1380 PRINT "THIRD BEAM SET ANGLE (Degs) =", Th3:
1390 PRINT "RUN NUMBER = ", Run
1400 PRINT "TUNNEL REFERENCE VOLTAGE (volts) =", Vref
1410 DATA 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180
1420 IF Ans$ = "Y" THEN GOTO 540
1430 DATA 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180
1440 IF Ans$ = "Y" THEN GOTO 540
1450 IF Ans$ = "Y" THEN GOTO 540
1460 IF Ans$ = "Y" THEN GOTO 540
1470 IF Ans$ = "Y" THEN GOTO 540
1480 IF Ans$ = "Y" THEN GOTO 540
1490 IF Ans$ = "Y" THEN GOTO 540
1500 IF Ans$ = "Y" THEN GOTO 540
1510 IF Ans$ = "Y" THEN GOTO 540
1520 IF Ans$ = "Y" THEN GOTO 540
1530 IF Ans$ = "Y" THEN GOTO 540
1540 IF Ans$ = "Y" THEN GOTO 540
1550 IF Ans$ = "Y" THEN GOTO 540
1560 IF Ans$ = "Y" THEN GOTO 540
1570 IF Ans$ = "Y" THEN GOTO 540
1580 IF Ans$ = "Y" THEN GOTO 540
1590 IF Ans$ = "Y" THEN GOTO 540
1600 IF Ans$ = "Y" THEN GOTO 540
1610 IF Ans$ = "Y" THEN GOTO 540
1620 IF Ans$ = "Y" THEN GOTO 540
1630 IF Ans$ = "Y" THEN GOTO 540
1640 IF Ans$ = "Y" THEN GOTO 540
1650 IF Ans$ = "Y" THEN GOTO 540
1660 IF Ans$ = "Y" THEN GOTO 540
1670 IF Ans$ = "Y" THEN GOTO 540
1680 IF Ans$ = "Y" THEN GOTO 540
1690 IF Ans$ = "Y" THEN GOTO 540
1700 IF Ans$ = "Y" THEN GOTO 540
1710 IF Ans$ = "Y" THEN GOTO 540
1720 IF Ans$ = "Y" THEN GOTO 540
1730 IF Ans$ = "Y" THEN GOTO 540
1740 IF Ans$ = "Y" THEN GOTO 540
1750 IF Ans$ = "Y" THEN GOTO 540
1760 IF Ans$ = "Y" THEN GOTO 540
1770 IF Ans$ = "Y" THEN GOTO 540
1780 IF Ans$ = "Y" THEN GOTO 540
1790 IF Ans$ = "Y" THEN GOTO 540
1800 IF Ans$ = "Y" THEN GOTO 540
1810 GOTO 1410
1820 END
1830 ! **************************** END OF MAIN PROGRAM LDV ****************************
1840 ! **************************** END OF MAIN PROGRAM LDV ****************************
1850 ! **************************** END OF MAIN PROGRAM LDV ****************************
1860 Atod:  | Subroutine for input from the LDV-A/D CI
1870 ! Enter routine with Ns = no. samples
1880 DISP "Press CONT to initiate data acquisition"
1890 PAUSE
1900 DISP "Acquiring Data"
1910 RESET 10
1920 WRITE 10,5;0
1930 WRITE 10,5;1  | start handshake by setting CTL0
1940 Ht-GNs
1950 FOR I=1 TO 5
1960 Dummy=READBIN(10)
1970 NEXT I
1980 REDIM D1(Ns,6)
1990 2000 ' ENTER I0 WFSH nt NOFORMAT;D1(*)  | fast data acquisition
2010 WRITE 10,5;0
2020 PRINT
2030 DISP "Data acquisition complete"
2040 PRINT
2040 ICALL Data_trans)
2050 RETURN
2060 ' DATA FILE
2070 DISP "Press CONT to initiate data acquisition"
2080 PAUSE
2090 File:  | write data file to floppy disc
2100 PRINTER IS 16
2110 PRINT
2120 PRINT "** DATA FILE WRITE TO FLOPPY DISK **"
2130 PRINT
2140 PRINT "At this point be sure there is a floppy in drive 0 of"
2150 PRINT "the 9895A with space for a file of 101, 256-byte records."
2160 PRINT
2170 PRINT
2180 Ans$="Y"
2190 INPUT: "DO YOU WISH TO WRITE THESE DATA TO DISK? (Y/N, DEFAULT Y)" ,Ans$
2200 IF Ans$="N" THEN GOTO 2390
2210 Files=Name$&VAL$(Run)
2220 DISP "File ":File$;" being written to disk"
2230 MASS STORAGE IS ":H8,0,0"  | set floppy drive (9895A drive 0) as default
2240 CREATE Files,101  | open file with 101 records 256 bytes each
2250 ASSIGN Files# TO #1
2260 PRINT #1;Date$
2270 PRINT #1;Tit1$
2280 PRINT #1;Name$
2290 PRINT #1;Dn
2300 PRINT #1:Nub,Num1x1,Num1x2,Num1x3,Theta,Run
2310 PRINT #1:Vref,Uref,D1,Df1,DF2,DF3
2320 PRINT #1:Xpos,Ypos,Zpos,Ns
2330 MAT PRINT #1;Data$
2340 PRINT "***** File write completed *****"
2350 ASSIGN # TO #1  | close data file
2360 MASS STORAGE IS ":H8,0,1"  | reset program disk as mass storage
2370 GOTO 2390
2380 Run=Run-1
2390 RETURN
2400 ! **************************** END OF MAIN PROGRAM LDV ****************************
Uguess: Estimate U-component of velocity

DISP "CALCULATING ESTIMATES OF U AND V VELOCITIES"

IF K11 = 3 THEN GOTO 2490

DATA 0,0,0

RESTORE 2460

READ Sw, Suw, Swu

Ubar = Su/Ns

Wbar = Su/Ns

Upri2 = Su/Ns - Ubar * Ubar

Vpri2 = Su/Ns - Vbar * Vbar

Wpri2 = Su/Ns - Wbar * Wbar

Uvbar = Swu/Ns - Ubar * Ybar

Uwbar = Suw/Ns - Ubar * Wbar

Ubar = Su/Ns

Ybar = Sv/Ns

Wbar = Sw/Ns

Uprl2 = Suu/Ns - Ubar * Ubar

Yprl2 = Sv/Ns - Ybar * Ybar

Wprl2 = Sw/Ns - Wbar * Wbar

Uvbar = Swu/Ns - Ubar * Ybar

Uwbar = Suw/Ns - Ubar * Wbar

PRINTER IS 0

PRINT "ESTIMATE OF QUANTITIES FROM SAMPLES: ", Ns

PRINT "Ubar = ", Ubar

PRINT "Vbar = ", Vbar

PRINT "Wbar = ", Wbar

PRINT "Uprl2 = ", Uprl2

PRINT "Vpri2 = ", Vpri2

PRINT "Wpri2 = ", Wpri2

PRINT "Uvbar = ", Uvbar

PRINT "Uwbar = ", Uwbar

PRINT "Yvbar = ", Yvbar

PRINT "Ywbar = ", Ywbar

GOTO 1700

SOURCE NAME: Find_vel

SOURCE 1 This subroutine converts raw data counts into 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 (Arrayd), the Bragg shift frequency (Fhub), the mixing frequencies (Num1, Num2, Num3), the fringe spacings (DF1, DF2, DF3), the crossing angle of the third beam (Theta), and the number of samples in a data point (Ns).

The outputs are the summations of various products of the velocity components, including U, V, W, U*U, V*V, W*W, U*V, U*W, and V*W.

2300 SOURCE

2400 SOURCE EXT Get_value 1 Declare subroutines stored outside of the main program.

2500 SOURCE EXT Get_info

2600 SOURCE EXT Get_element

2700 SOURCE EXT Int_to_rel

2800 SOURCE EXT Rel_math

2900 SOURCE EXT Put_value

2900 SOURCE

2910 SOURCE COMMON

2920 SOURCE Data_par: INT (*)

2930 SOURCE D1_par: INT (*)

2940 SOURCE Ns_par: INT

2950 SOURCE Nn_par: INT

2960 SOURCE Df1_par: REL

2970 SOURCE Df2_par: REL

2980 SOURCE Df3_par: REL

2990 SOURCE Theta_par: REL

3000 SOURCE Hub_par: REL
3010 ISOURCE Nmix1_par: REL
3020 ISOURCE Nmix2_par: REL
3030 ISOURCE Nmix3_par: REL
3040 ISOURCE Suv_par: REL
3050 ISOURCE Sv_par: REL
3060 ISOURCE Sw_par: REL
3070 ISOURCE Suw_par: REL
3080 ISOURCE Suv2_par: REL
3090 ISOURCE Svw_par: REL
3100 ISOURCE Suv3_par: REL
3110 ISOURCE Suv4_par: REL
3120 ISOURCE Suv5_par: REL
3130 ISOURCE !
3140 ISOURCE Arrayd: BSS 39 1 Reserve space for data array
3150 ISOURCE Elementd: EQU Arrayd+16 1 descriptor.
3160 ISOURCE Array1: BSS 4096 1 Reserve space for lookup tables used
3170 ISOURCE Array2: BSS 4096 1 for count to velocity conversion.
3180 ISOURCE Array3: BSS 4096 1 Reserve space for various input
3190 ISOURCE Ns: BSS 1 1 and output variables.
3200 ISOURCE Df1: BSS 4 1
3210 ISOURCE Df2: BSS 4 1
3220 ISOURCE Df3: BSS 4 1
3230 ISOURCE Theta: BSS 4 1
3240 ISOURCE Nub: BSS 4 1
3250 ISOURCE Numb1: BSS 4 1
3260 ISOURCE Numb2: BSS 4 1
3270 ISOURCE Numb3: BSS 4 1
3280 ISOURCE Su: BSS 4 1
3290 ISOURCE Sw: BSS 4 1
3300 ISOURCE Suv: BSS 4 1
3310 ISOURCE Suv2: BSS 4 1
3320 ISOURCE Suv3: BSS 4 1
3330 ISOURCE Suv4: BSS 4 1
3340 ISOURCE Suv5: BSS 4 1
3350 ISOURCE Suv6: BSS 4 1
3360 ISOURCE Suv7: BSS 4 1
3370 ISOURCE Count: BSS 1 1 Count and I are general purpose index vari-
3380 ISOURCE I: BSS 1 1 ables. Count is usually 0, 1, or 2, to denot-
3390 ISOURCE Check: BSS 1 1 whether U,V, or W is being calculated.
3400 ISOURCE Int: BSS 1 1 Int, Address, and Offset are all general
3410 ISOURCE Address: BSS 1 1 purpose storage areas.
3420 ISOURCE Offset: BSS 1 1
3430 ISOURCE R1: BSS 4 1 R1, R2, and R3 are the count-to-
3440 ISOURCE R2: BSS 4 1 frequency conversion factors.
3450 ISOURCE R3: BSS 4 1
3460 ISOURCE Xvar: BSS 4 1 Xvar and Yvar are general purpose real
3470 ISOURCE Yvar: BSS 4 1 number storage areas.
3480 ISOURCE U: BSS 4 1 reserve space for instantaneous velocity
3490 ISOURCE V: BSS 4 1 components.
3500 ISOURCE W: BSS 4 1
3510 ISOURCE Uv: BSS 4 1
3520 ISOURCE Uv: BSS 4 1
3530 ISOURCE Cos: BSS 4 1 Cos and Sin are the cosine and sine of Theta.
3540 ISOURCE Sin: BSS 4 1
3550 ISOURCE Rad: DAT 5.729578E1
3560 ISOURCE Mill: DAT 1.66
3570 ISOURCE One: DAT 1.
3580 ISOURCE Zero: DAT 0.
3590 ISOURCE !
3600 ISOURCE SUB
3610  ISOURCE  Find_vel:
3620  ISOURCE  LDR  =Ns
3630  ISOURCE  LDB  =Ns_par
3640  ISOURCE  JSM  Get_value  ! Get number of samples.
3650  ISOURCE  LDB  =Arrayd
3660  ISOURCE  JSM  Get_value  ! Get parameters of data array.
3670  ISOURCE  LDA  =Df1
3680  ISOURCE  LDB  =Df1_par
3690  ISOURCE  JSM  Get_value
3700  ISOURCE  LDA  =Df2
3710  ISOURCE  LDB  =Df2_par
3720  ISOURCE  JSM  Get_value
3730  ISOURCE  LDA  =Df3
3740  ISOURCE  LDB  =Df3_par
3750  ISOURCE  JSM  Get_value
3760  ISOURCE  LDA  =Theta
3770  ISOURCE  LDB  =Theta_par
3780  ISOURCE  JSM  Get_value
3790  ISOURCE  LDA  =Hub
3800  ISOURCE  LDB  =Hub_par
3810  ISOURCE  JSM  Get_value
3820  ISOURCE  LDA  =Numix1
3830  ISOURCE  LDB  =Numix1_par
3840  ISOURCE  JSM  Get_value
3850  ISOURCE  LDA  =Numix2
3860  ISOURCE  LDB  =Numix2_par
3870  ISOURCE  JSM  Get_value
3880  ISOURCE  LDA  =Numix3
3890  ISOURCE  LDB  =Numix3_par
3900  ISOURCE  JSM  Get_value
3910  ISOURCE
3920  ISOURCE  The loop headed by Get_freq is repeated three times to get
3930  ISOURCE  the count-to-frequency conversion factors (which depend on
3940  ISOURCE  the range) for U, V, and W. Whenever a loop is controlled by
3950  ISOURCE  the variable "Count", the loop contains operations which are
3960  ISOURCE  the same for U, V, and W.
3970  ISOURCE  Loop:
3980  ISOURCE
3990  ISOURCE  Loopend:
4000  ISOURCE  Get_freq:
4010  ISOURCE  LDR  =Cou
4020  ISOURCE  LDB  =Ns
4030  ISOURCE  MPY  ! the data array which contains the
4040  ISOURCE  STA  Element
4050  ISOURCE  LDR  =Int
4060  ISOURCE  LDB  =Arrayd
4070  ISOURCE  JSM  Get_element
4080  ISOURCE  LDA  =Int
4090  ISOURCE  LDB  =15360
4100  ISOURCE  AND  B
4110  ISOURCE  SRL  10
4120  ISOURCE  TCA
4130  ISOURCE  LDB  =15
4140  ISOURCE  ADR  B
4150  ISOURCE  LDB  =1
4160  ISOURCE  S2A  Loopend
4170  ISOURCE  Loop:
4180  ISOURCE  SBL  1
4190  ISOURCE  JMP  Loop
4200  ISOURCE  Loopend:
4210  ISOURCE  LDA  =Int
4220  ISOURCE  Convert the power of two into a
4210 ISOURCE STA Oper_1
4220 ISOURCE LDA =Yvar
4230 ISOURCE STA Result
4240 ISOURCE JSM Int_to_rel
4250 ISOURCE LDA =3.2E4
4260 ISOURCE LDB =Xvar
4270 ISOURCE LDA =Yvar
4280 ISOURCE STA Oper_2
4290 ISOURCE LDA Count
4300 ISOURCE LDB =4
4310 ISOURCE MPY
4320 ISOURCE ADA =R1
4330 ISOURCE STA Result
4340 ISOURCE LDA =2
4350 ISOURCE LDB =1471558
4360 ISOURCE JSM Rel_math
4370 ISOURCE LDA =3
4380 ISOURCE CPA Count
4390 ISOURCE JMP ++2
4400 ISOURCE LDA =Array1
4410 ISOURCE CLR 16
4420 ISOURCE ADA =16
4430 ISOURCE DSZ B
4440 ISOURCE JSM Continue
4450 ISOURCE LDA =Su
4460 ISOURCE LDB =9
4470 ISOURCE CLR 4
4480 ISOURCE ADA =4
4490 ISOURCE DSZ B
4500 ISOURCE JMP Clear
4510 ISOURCE LDA =Theta
4520 ISOURCE STA Oper_1
4530 ISOURCE LDB =769
4540 ISOURCE LDA =Rad
4550 ISOURCE STA Oper_2
4560 ISOURCE LDA =Xvar
4570 ISOURCE LDA =2
4580 ISOURCE LDB =1471558
4590 ISOURCE JSM Rel_math
4600 ISOURCE LDA =Xvar
4610 ISOURCE STA Oper_1
4620 ISOURCE LDA =Sin
4630 ISOURCE STA Result
4640 ISOURCE LDA =1
4650 ISOURCE LDB =34213B
4660 ISOURCE JSM Rel_math
4670 ISOURCE LDA =Cos
4680 ISOURCE LDA =1
4690 ISOURCE LDB =34224B
4700 ISOURCE JSM Rel_math

real number.

Divide 3.2E4 by the appropriate power of two, using BCD math.

Decide whether to put the result in R1, R2, or R3, depending on Count.

Now, finally, call the utility to perform the division.

Increment and check Count so as to follow the loop three times.

Zero out the entire count-to-velocity conversion table so that it must be recalculated for each point. (This must be done if the mixing frequencies or ranges are changed between counts.)

Set initial values of Su, Sv, Sw, Suu, etc. to zero.

Convert Theta from degrees to radians using the Rel_math utility.

Find the sine and cosine of Theta, and store them in the locations Sin and Cos, respectively.
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 \((\text{Hub} - \text{Num} \times \text{N})\) and \((\text{Mill} + \text{DfN})\), where \(N\) is 1, 2, and 3.

The loop 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 is 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 any 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.

Figure out which element of the data array we want to pick up.
I 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 (NumiNatxN)-FrequencyN.

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

// Leave V and W alone.
6010 ISOURCE STA Oper_2
6020 ISOURCE LDA =Xvar
6030 ISOURCE STA Result
6040 ISOURCE LDA =2
6050 ISOURCE LDB =146717B
6060 ISOURCE JSM Rel_math
6070 ISOURCE Same sign: NOP
6080 ISOURCE
6090 ISOURCE LDA Result
6100 ISOURCE STA Oper_1
6110 ISOURCE LDA =DF1
6120 ISOURCE ADA Offset
6130 ISOURCE STA Oper_2
6140 ISOURCE LDA Address
6150 ISOURCE ADA =-1
6160 ISOURCE STA Result
6170 ISOURCE LDA =2
6180 ISOURCE LDB =1470378
6190 ISOURCE JSM Rel_math
6200 ISOURCE
6210 ISOURCE Over:
6220 ISOURCE LDA Address
6230 ISOURCE ADA =-1
6240 ISOURCE LDB =U
6250 ISOURCE ADA Offset
6260 ISOURCE XFR 4
6270 ISOURCE ISZ Count
6280 ISOURCE LDA =-3
6290 ISOURCE ADA Count
6300 ISOURCE Pause: NOP
6310 ISOURCE SZA ++2
6320 ISOURCE JMP Begin
6330 ISOURCE STA Count
6340 ISOURCE
6350 ISOURCE Now we convert the \( u \) we have obtained (which is measured at
6360 ISOURCE an angle \( \Theta \) to the \( V \)-axis) to the \( W \) we want (which should
6370 ISOURCE be measured at an angle of 90 degrees to the \( V \)-axis).
6380 ISOURCE Thus find \( W = (\mathbf{u} - \mathbf{V} \times \cos(\Theta)) / \sin(\Theta) \).
6390 ISOURCE
6400 ISOURCE LDA =V
6410 ISOURCE STA Oper_1
6420 ISOURCE LDA =Cos
6430 ISOURCE STA Oper_2
6440 ISOURCE LDA =Xvar
6450 ISOURCE STA Result
6460 ISOURCE LDA =2
6470 ISOURCE LDB =1470378
6480 ISOURCE JSM Rel_math
6490 ISOURCE LDA Result
6500 ISOURCE STA Oper_2
6510 ISOURCE LDA =W
6520 ISOURCE STA Oper_1
6530 ISOURCE LDA =Yvar
6540 ISOURCE STA Result
6550 ISOURCE LDA =2
6560 ISOURCE LDB =1467178
6570 ISOURCE JSM Rel_math
6580 ISOURCE LDA Result
6590 ISOURCE STA Oper_1
6600 ISOURCE LDA =Sin

34
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.

```
<table>
<thead>
<tr>
<th>Line</th>
<th>Source Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>6610</td>
<td>ISOURCE STA Oper_2</td>
</tr>
<tr>
<td>6620</td>
<td>ISOURCE LDA =U</td>
</tr>
<tr>
<td>6630</td>
<td>ISOURCE STA Result</td>
</tr>
<tr>
<td>6640</td>
<td>ISOURCE LDA =2</td>
</tr>
<tr>
<td>6650</td>
<td>ISOURCE LDB =1471553</td>
</tr>
<tr>
<td>6660</td>
<td>ISOURCE JSM Rel_math</td>
</tr>
<tr>
<td>6670</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6680</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6690</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6700</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6710</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6720</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6730</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6740</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6750</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6760</td>
<td>ISOURCE LDA =Su</td>
</tr>
<tr>
<td>6770</td>
<td>ISOURCE LDB =U</td>
</tr>
<tr>
<td>6780</td>
<td>ISOURCE JSM Add</td>
</tr>
<tr>
<td>6790</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6800</td>
<td>ISOURCE LDA =u</td>
</tr>
<tr>
<td>6810</td>
<td>ISOURCE STA Oper_1</td>
</tr>
<tr>
<td>6820</td>
<td>ISOURCE STA Result</td>
</tr>
<tr>
<td>6830</td>
<td>ISOURCE LDA =2</td>
</tr>
<tr>
<td>6840</td>
<td>ISOURCE LDB =147037B</td>
</tr>
<tr>
<td>6850</td>
<td>ISOURCE JSM Rel_math</td>
</tr>
<tr>
<td>6860</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6870</td>
<td>ISOURCE LDA =Su</td>
</tr>
<tr>
<td>6880</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6890</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6900</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6910</td>
<td>ISOURCE LDA =Su</td>
</tr>
<tr>
<td>6920</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6930</td>
<td>ISOURCE LDB =v</td>
</tr>
<tr>
<td>6940</td>
<td>ISOURCE JSM Add</td>
</tr>
<tr>
<td>6950</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6960</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6970</td>
<td>ISOURCE LDA =u</td>
</tr>
<tr>
<td>6980</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>6990</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>7000</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>7010</td>
<td>ISOURCE LDB =147037B</td>
</tr>
<tr>
<td>7020</td>
<td>ISOURCE JSM Rel_math</td>
</tr>
<tr>
<td>7030</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>7040</td>
<td>ISOURCE LDA =Su</td>
</tr>
<tr>
<td>7050</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>7060</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>7070</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>7080</td>
<td>ISOURCE LDA =v</td>
</tr>
<tr>
<td>7090</td>
<td>ISOURCE STA Oper_1</td>
</tr>
<tr>
<td>7100</td>
<td>ISOURCE STA Oper_2</td>
</tr>
<tr>
<td>7110</td>
<td>ISOURCE LDA =xvar</td>
</tr>
<tr>
<td>7120</td>
<td>ISOURCE STA Result</td>
</tr>
<tr>
<td>7130</td>
<td>ISOURCE LDA =2</td>
</tr>
<tr>
<td>7140</td>
<td>ISOURCE LDB =147037B</td>
</tr>
<tr>
<td>7150</td>
<td>ISOURCE JSM Rel_math</td>
</tr>
<tr>
<td>7160</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>7170</td>
<td>ISOURCE LDA =SuV</td>
</tr>
<tr>
<td>7180</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>7190</td>
<td>ISOURCE</td>
</tr>
<tr>
<td>7200</td>
<td>ISOURCE</td>
</tr>
</tbody>
</table>
```

Find $Su=Su+U$

Find $U*=U$

Find $Su=Su+(U*U)$

Find $U*=\sum$ (U*U)

Find $U*=U*V$

Find $U*=\sum$ (U*V)

Find $V*=V*V$

Find $V*=\sum$ (V*V)
LDA = Sw     \ Find Su=Sw+W
LDB = W
JSM Add

LDA = U     \ Find U=W
LDB = Xvar
STA Oper_1
LDA = Xvar
STA Result
LDA = 2
LDB = 147037B
JSM Rel_math

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

LDA = W     \ Find V=W
STA Oper_1
LDB = V
STA Oper_2
LDA = Xvar
STA Result
LDA = 2
LDB = 147037B
JSM Rel_math

LDA = Sw     \ Find Suw=Sww+(U*W)
LDB Result
JSM Add

LDA = W     \ Find W+W
LDB = Xvar
STA Oper_1
STA Oper_2
LDA = Xvar
STA Result
LDA = 2
LDB = 147037B
JSM Rel_math

LDA = Sw     \ Find Suw=Suw+(W*W)
LDB Result
JSM Add

LDA = Sw     \ Continue to calculate running sum until out of samples.
LDB = Sw_par
JSM Put_value

LDA = Su
LDB = Su_par
JSM Put_value
LDA = Sw
LDB = Sw_par
JSM Put_value
LDA = Su
LDB = Su_par
JSM Put_value
LDA = Suw

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 = Sw
LDB = Sw_par
JSM Put_value
LDA = Su
LDB = Su_par
JSM Put_value
LDA = Suw
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.
**Data_trans Subprogram**

- **Start1:**
  - LDA = Array
  - LDA = Data_par
  - LDA = Array1
  - LDB = D1_par
  - LDB = Array1
  - JSM Get_info
  - JSM Get_value

- **Start2:**
  - LDA = Array
  - LDA = Data_par

**Hist:**

Subroutine to produce online histograms from raw LDV data.

ON KEY #9 GOTO 9270

Hn=3

Ns_temp=Ns
3010 "Ho. of data samples per point (2000 max.):", Hs
3020 INPUT "No. of data samples per point (2000 max.)":", Hs
3030 REDIM Data(3, Hs)
3040 REDIM Dl(Hs, 6)
3050 DISP "Press CONT to initiate data acquisition, press K0 to return to main program."
3060 PAUSE
3070 GCLEAR
3080 GRAPHICS
3090 FOR 1 TO 10
3100 WRITE 10, 5; 0
3110 NEXT I
3120 ENTER 10 WFHS NT NOFORMAT; D1(*) 'Fast data acquisition
3130 WRITE 10, 5; 0
3140 ICALL Data_trans
3150 ICALL Draw_hist
3160 IF Hs=2000 THEN GOTO 9270
3170 Nt=1
3180 GOTO 9110
3190 IF Hs=2000 THEN DUMP GRAPHICS
3200 WRITE 10, 5; 0
3210 Ns=Ns_temp
3220 REDIM Data(3, Hs)
3230 REDIM Dl(Hs, 6)
3240 EXIT GRAPHICS
3250 RETURN
3260 ISOURCE
3270 ISOURCE NAM Draw_hist
3280 ISOURCE The subroutine works by first going to the samples-per-count
3290 ISOURCE tables and using them to draw the old histograms in black
3300 ISOURCE (Bit=0) to erase them. Then it calculates new samples-per-
3310 ISOURCE count tables from the data acquired from the LDV, and uses
3320 ISOURCE the new tables to draw histograms in white (Bit=1). Then the
3330 ISOURCE subroutine returns to the main program. The very first time
3340 ISOURCE (determined by Ntimes=Hs) the subroutine is called it doesn’t
3350 ISOURCE erase the old histograms because there aren’t any.
3360 ISOURCE
3370 ISOURCE EXT Get_value ' Declare subroutines stored
3380 ISOURCE EXT Get_info ' outside of the main program.
3390 ISOURCE EXT Get_element
3400 ISOURCE
3410 ISOURCE COM
3420 ISOURCE Data_par: INT(*) ' Declare common variables.
3430 ISOURCE Dl_par: INT(*)
3440 ISOURCE Ns_par: INT
3450 ISOURCE Ntimes_par: INT
3460 ISOURCE Array: BSS 39 ' Reserve space for data array
3470 ISOURCE Element: EQU Array+16 ' descriptor.
3480 ISOURCE Array1: BSS 1024 ' Reserve space for tables which hold
3490 ISOURCE Array2: BSS 1024 ' the number of samples per count.
3500 ISOURCE Array3: BSS 1024
9610  SOURCE Hs:  BSS 1  
9620  SOURCE Count:  BSS 1  
9630  SOURCE I:  BSS 1  
9640  SOURCE Check:  BSS 1  
9650  SOURCE Int:  BSS 1  
9660  SOURCE Address:  BSS 1  
9670  SOURCE X_coord:  BSS 1  
9680  SOURCE Y_coord:  BSS 1  
9690  SOURCE Bit:  BSS 1  
9700  SOURCE Ntimes:  BSS 1  
9710  SOURCE  
9720  SOURCE  
9730  SOURCE  
9740  SOURCE  
9750  SOURCE Draw_hist:  LDA =Hs  
9760  SOURCE LDB #Hs_par  
9770  SOURCE  
9780  SOURCE LDH =Arrayd  
9790  SOURCE LDB =Data_par  
9800  SOURCE  
9810  SOURCE LDH =Ntimes  
9820  SOURCE LDB =Ntime_par  
9830  SOURCE  
9840  SOURCE  
9850  SOURCE LDA =0  
9860  SOURCE STA Bit  
9870  SOURCE LDA Ntimes  
9880  SOURCE RZA +=2  
9890  SOURCE JMP Acquire  
9900  SOURCE  
9910  SOURCE Do_graph:  LDA =0  
9920  SOURCE STA Count  
9930  SOURCE Make_hist:  LDA Count  
9940  SOURCE ADR =1  
9950  SOURCE LDB =1024  
9960  SOURCE STB I  
9970  SOURCE MPY  
9980  SOURCE  
9990  SOURCE ADR =Array1  
10000  SOURCE ADR =-1  
10010  SOURCE STA Address  
10020  SOURCE Make Rod:  LDB Address, I  
10030  SOURCE DSZ Address  
10040  SOURCE ADR Address, I  
10050  SOURCE DSZ Address  
10060  SOURCE DSZ I  
10070  SOURCE RZB +=2  
10080  SOURCE JMP Skip_bits  
10090  SOURCE STB Int  
10100  SOURCE ADR =150  
10110  SOURCE BSN +=3  
10120  SOURCE LDB =150  
10130  SOURCE STB Int  
10140  SOURCE Make_bit:  LDA =150  
10150  SOURCE LDB Count  
10160  SOURCE SAR 1  
10170  SOURCE STA X_coord  
10180  SOURCE Make_bit:  LDA =150  
10190  SOURCE LDB Count  
10200  SOURCE
10210 ISOURCE ADB = 1
10220 ISOURCE MPY
10230 ISOURCE LDB Int
10240 ISOURCE TCB
10250 ISOURCE ADA B
10260 ISOURCE STR Y_coord
10270 ISOURCE LDA 13
10280 ISOURCE STR Pw
10290 ISOURCE LDA 513
10300 ISOURCE SFC *
10310 ISOURCE STR R5
10320 ISOURCE LDA Y_coord
10330 ISOURCE LDB 36
10340 ISOURCE MPY
10350 ISOURCE LDB X_coord
10360 ISOURCE SBR 4
10370 ISOURCE ADA B
10380 ISOURCE CMA
10390 ISOURCE SFC *
10400 ISOURCE STA R4
10410 ISOURCE STA R7
10420 ISOURCE STR R4
10430 ISOURCE LDA X_coord
10440 ISOURCE AND =17B
10450 ISOURCE LDB Bit
10460 ISOURCE SBL 15
10470 ISOURCE IOR B
10480 ISOURCE SFC *
10490 ISOURCE STR R4
10500 ISOURCE STA R7
10510 ISOURCE DSZ Int
10520 ISOURCE JMP Make_bit
10530 ISOURCE Skip_bits: DSZ 1
10540 ISOURCE JMP Make_loc
10550 ISOURCE ISZ Count
10560 ISOURCE Acquire: LDA Bit
10570 ISOURCE SRA ++2
10580 ISOURCE JMP Stop
10590 ISOURCE LDA =-3
10600 ISOURCE SRA Count
10610 ISOURCE JMP Make_hist
10620 ISOURCE ISZ Count
10630 ISOURCE Acquire: LDA Bit
10640 ISOURCE SRA ++2
10650 ISOURCE JMP Stop
10660 ISOURCE LDA =-3
10670 ISOURCE SRA Count
10680 ISOURCE LDA =-3
10690 ISOURCE SRA Count
10700 ISOURCE CLR 16
10710 ISOURCE ADA =16
10720 ISOURCE DSZ B
10730 ISOURCE JMP Clear
10740 ISOURCE Clear:
10750 ISOURCE LDA =0
10760 ISOURCE STR Count
10770 ISOURCE LDA =5
10780 ISOURCE STR Check
10790 ISOURCE LDA Ms
10800 ISOURCE STR I
10810  ISOURCE  ' Begin:
10820  ISOURCE      |  LDA  Count        | Figure out which element of the
10830  ISOURCE      |  LDB  Rs          |  data array we want to pick up.
10840  ISOURCE      |  MPY              |
10850  ISOURCE      |  ADA  I           |
10860  ISOURCE      |  ADA  =-1         |
10870  ISOURCE      |  STR  Elementd    | Get a raw datum from the data array.
10880  ISOURCE      |  LDA  =Int        |
10890  ISOURCE      |  LDB  =Arrayd     |
10900  ISOURCE      |  JSR  Get_element |
10910  ISOURCE      |  LDA  Int         |
10920  ISOURCE      |  LDB  =1023       | Strip off the first six
10930  ISOURCE      |  AND  B           |  bits of the raw data word.
10940  ISOURCE      |  LDB  Count       |  See if Count = Check.
10950  ISOURCE      |  TCB              |
10960  ISOURCE      |  ADA  Check       |
10970  ISOURCE      |  S2Z  Straight    |  If true, use the modified data
10980  ISOURCE      |  TCA              |  word as an index. If not, use
10990  ISOURCE      |  ADA  =1024       |  1024 minus the data word.
11000  ISOURCE      |  STR  Int         |  Store the count we have gotten.
11010  ISOURCE      |  LDA  =1024       |  Increment the appropriate place
11020  ISOURCE      |  LDB  Count       |  in the samples-per-count table
11030  ISOURCE      |  MPY              |  by one.
11040  ISOURCE      |  ADA  =Arrayd     |
11050  ISOURCE      |  ADA  Int         |
11060  ISOURCE      |  ISZ  R,A,I       |
11070  ISOURCE      |  ISZ  Count       |  Have U,V, and W all been done?
11080  ISOURCE      |  LDA  =-3         |
11090  ISOURCE      |  ADA  Count       |
11100  ISOURCE      |  S2Z  =+2         |  If not, go back again.
11110  ISOURCE      |  JMP  Begin       |
11120  ISOURCE      |  JMP  Begin       |
11130  ISOURCE      |  STR  Count       |  If so, set Count=0.
11140  ISOURCE      |  DSZ  I           |  Continue to fill samples-per-count
11150  ISOURCE      |  JMP  Begin       |  tables until out of samples.
11160  ISOURCE      |  JMP  Begin       |
11170  ISOURCE      |  ISZ  Bit         |  How set Bit=1 and go back and draw
11180  ISOURCE      |  JMP  Do_graph    |  the new histograms in white.
11190  ISOURCE      |  RET  I           |
11200  ISOURCE      |  JMP  Begin       |
11210  ISOURCE      |  RET  I           |
11220  ISOURCE      |  END  Draw_hist   |
REM PROGRAM STAT
10 PROGRAM TO REDUCE RAW DATA FROM THREE-COMPONENT LDV SYSTEM
20 Input is 3 channels of raw LV data and calibration constants
30 Output is the three components of mean velocity, all the
40 components of Reynolds stress and some selected third-order
50 products. Can output out counts which are excessively far
60 from the mean. Display histograms of raw and
70 filtered data for all 3 channels. Plots data at end and displays
80 reduced data for entire run. Can write a summary file containing
90 all reduced data and calibration constants.
100 OPTION BASE 1
110 ICOM 10000
120 IDELETE ALL
130 IASSEMBLE Find_vel
140 COM INTEGER Date(3,2000),N,Ng,Coutbin(3,1024)
150 COM INTEGER Rangel,Ran2,Ran3
160 COM REAL Sdev,Df1,Df2,Df3
170 COM REAL Theta,Hub,Numx1,Numx2,Numx3
180 COM REAL Su,Su,Su,Su,Su,Su,Su,Su,Su
190 COM REAL Su,Su,Su,Su,Su,Su
200 DIM L1(3),R1(3),L1(3),U1(3),U(3)
210 DIM Date[#00],File[#6],Name[#4],Title[#80]
220 REAL Xpos,Ypos,Zpos
230 REAL Point(20,15)
240 REAL Re,Ue
250 INTEGER Flno,A,Run
260 DATA 9,0,9,9,9
270 READ Max_y,Min_y,Filter,Sdev
280 DATA 75.64,4.64,15.64,99.,69.,39.
290 MAT READ L1,U1
300 DATA 5,46.87,41.82,123
310 MAT READ L,R1
320 DATA -0.8,2.4,0.,.035,-.0025,.01
330 READ L1m(1),U1m(1),L1m(2),U1m(2),L1m(3),U1m(3)
340 DATA .4,.085,.0825
350 MAT READ Scale
360 PRINT "** << PROGRAM STAT : 3-COMPONENT VELOCITY DATA >> **"
370 PRINT "PROGRAM STRUCTURE"
380 PRINT "1. Read raw data from floppy disc"
390 PRINT "2. Convert to velocity"
400 PRINT "3. Calculate statistics"
410 PRINT "4. Print results"
420 PRINT "5. Write to disc file"
430 INPUT "** Read raw data from floppy disc **"
440 INPUT "Enter parent filename (or E to exit program) ":",Name$
450 IF Name$="E" THEN GOTO 1780
460 INPUT "Enter no of first and last data files ":,File1,Num
470 K11=3
480 INPUT "Is this a two-channel or three-channel run (2/3, Default 3) ":,K11
490 FOR Hists=1 TO 3
500 INPUT "Is this a two-channel or three-channel run (2/3, Default 3) ":,K11
510 NEXT Hists
520 NEXT
530 NEXT
540 NEXT
550 INPUT "Enter parent filename (or E to exit program) ":",Name$
560 IF Name$="E" THEN GOTO 1780
570 INPUT "Enter no of first and last data files ":,File1,Num
580 K11=3
590 INPUT "Is this a two-channel or three-channel run (2/3, Default 3) ":,K11
600 NEXT Hists=1 TO 3
610 NEXT Hists
620 NEXT
630 NEXT
640 NEXT
650 NEXT
660 NEXT
670 NEXT
680 NEXT
690 NEXT
700 NEXT
710 NEXT
720 NEXT
730 NEXT
740 NEXT
750 NEXT
760 NEXT
770 NEXT
780 NEXT
790 NEXT
800 NEXT
810 NEXT
820 NEXT
830 NEXT
840 NEXT
850 NEXT
860 NEXT
870 NEXT
880 NEXT
890 NEXT
900 NEXT
910 NEXT
920 NEXT
930 NEXT
940 NEXT
950 NEXT
960 NEXT
970 NEXT
980 NEXT
990 NEXT
1000 NEXT
610 INPUT "Do you wish to see histograms of the data points (Y/N, Default Y) ", Hist$  
620 Filter$="N"  
630 INPUT "Do you wish to filter the output (Y/N, Default N)?", Filter$  
640 IF Filter$="N" THEN GOTO 660  
650 INPUT "Enter standard deviation to throw out", Sdev  
660 DISP "READING RAW DATA FROM DISK"  
670 MASS STORAGE IS ":H8,9,0"  
680 PRINTER IS 0  
690 File$=File$  
700 IF File$>N THEN GOTO 1740  
710 Files$=Name$&VAL$(File$)  
720 ASSIGN Files$ TO #1  
730  
740 READ #1;Date$  
750 READ #1;T1t1$  
760 READ #1;Name$  
770 READ #1;Dn  
780 READ #1;Nub,Nu1x1,Nu1x2,Nu1x3,Theta,Run  
790 READ #1;Vref,Up,DF1,DF2,DF3  
800 READ #1;Xpos,Ypos,Zpos,Ns  
810 MAT READ #1;Data  
820  
830 PRINT Tit1$  
840 PRINT "DATE AND TIME OF TEST ":,Date$  
850 PRINT "FILE NAME ON DISK ":,Name$  
860 PRINT "DISK NUMBER ":,Dn  
870 PRINT "RUN NO. (POINT NO. IN PROFILE) ":,Run  
880 PRINT "X,Y,Z ":,Xpos,Ypos,Zpos  
890  
900 ' CALCULATE VELOCITIES  
910 DISP "CALCULATING VELOCITIES AND TAKING RUNNING SUMS"  
920  
930 ICALL Find vel  
940  
950 ' CALCULATE STATISTICS  
960 DISP "CALCULATING STATISTICS"  
970  
980 IF Kill w=3 THEN GOTO 1030  
990 DATA 0,0,0,0,0,0,0  
1000 RESTORE 990  
1010 READ Su,Suu,Suv,Suu,Suuw,Suuw,Suww  
1020  
1030 Ubar=Su/Ns  
1040 Vbar=Sv/Ns  
1050 Wbar=Sv/Ns  
1060 Up1r2=Suw/Ngs-Ubar*Ubar  
1070 Vpr12=Suv/Ngs-Vbar*Vbar  
1080 Wpr12=Suv/Ngs-Wbar*Wbar  
1090 Uvbar=Suw/Ngs-Ubar*Vbar  
1100 Uvbar=Suw/Ngs-Ubar*Wbar  
1110 Vvbar=Sv/Ngs-Vbar*Vbar  
1120 Uvbar=Suw/Ngs-Ubar*Vbar  
1130 Uvbar=Suw/Ngs-Ubar*Wbar  
1140 Uvbar=Suw/Ngs-Ubar*Vbar  
1150 Uvbar=Suw/Ngs-Ubar*Wbar  
1160 Uvbar=Suw/Ngs-Ubar*Vbar  
1170  
1180 ' STORE RESULTS IN ARRAY FOR PLOTTING  
1190  
1200 File$=File$=File$+1
1210 Point(Fl,1)=Ypos
1220 Point(Fl,2)=Ubar
1230 Point(Fl,3)=Vbar
1240 Point(Fl,4)=Wbar
1250 Point(Fl,5)=Uprl2
1260 Point(Fl,6)=Wprl2
1270 Point(Fl,7)=Wprl2
1280 Point(Fl,8)=Uvbar
1290 Point(Fl,9)=Uwbar
1300 Point(Fl,10)=Vvbar
1310 Point(Fl,11)=Uwbar
1320 Point(Fl,12)=Uuubar
1330 Point(Fl,13)=Uuubar
1340 Point(Fl,14)=Uuubar
1350 Point(Fl,15)=Uuubar
1360 IF Ypos<Max_y THEN GOTO 1390
1370 Max_y=Ypos
1380 Min_y=Ubar
1390 IF Ypos>Min_y THEN GOTO 1450
1400 Min_y=Ypos
1410 Max_y=Ubar
1420 IF PLOT RESULTS
1430 PRINT "Bragg shift frequency = ", Nub
1440 PRINT "Mixing frequencies = Num1x1, Num1x2, Num1x3
1450 PRINT "Third beam angle = ", Theta
1460 PRINT "Run number = ", Run
1470 PRINT "Tunnel reference voltage (volts) = ", Vref
1480 PRINT "Fringe spacings = ", Df1, Df2, Df3
1490 PRINT "No. of samples = ", Ns
1500 PRINT "Ranges = ", Range1, Range2, Range3
1510 PRINT "Filter [no. of std. deviations] = ", Sdev
1520 PRINT "Adjusted no. of samples = ", Ns
1530 PRINT "Ubar = ", ROUND(Ubar, 4)
1540 PRINT "Vbar = ", ROUND(Vbar, 4)
1550 PRINT "Wbar = ", ROUND(Wbar, 4)
1560 PRINT "Up'l2 = ", ROUND(Uprl2, 4)
1570 PRINT "Wprl2 = ", ROUND(Wprl2, 4)
1580 PRINT "Uvbar = ", ROUND(Uvbar, 4)
1590 PRINT "Uwbar = ", ROUND(Uwbar, 4)
1600 PRINT "Vvbar = ", ROUND(Vvbar, 4)
1610 PRINT "Wvbar = ", ROUND(Wvbar, 4)
1620 PRINT "Uvbar = ", ROUND(Uvbar, 4)
1630 PRINT "Wvbar = ", ROUND(Wvbar, 4)
1640 PRINT "UUvbar = ", ROUND(UUvbar, 4)
1650 PRINT "UVvbar = ", ROUND(UVvbar, 4)
1660 PRINT "UUwbar = ", ROUND(UUwbar, 4)
1670 PRINT "UVwbar = ", ROUND(UVwbar, 4)
1680 PRINT "UWWbar = ", ROUND(UWWbar, 4)
1690 PRINT "UWWbar = ", ROUND(UWWbar, 4)
1700 PRINT IF Hist$="Y" THEN GOSUB Histogram
1710 Flno=Flno+1
1720 GOTO 700
1730 Plot$="Y"
1740 INPUT "Do you wish to see the turbulence quantities (Y/N, Default Y) ?", Pt
1750 IF Pt="N" THEN GOSUB Plot
1760 PRINT "MASS STORAGE IS ": H8, 0, 1
1770 PRINT "END"
1810 !*********************************************************PLOT DATA*************************************************************
1820 !
1830 Plot: PRINTER IS 16
1840 PRINT "Estimate of normalizing velocity is:";Ue
1850 INPUT "Enter correct normalizing velocity if different.";Ue
1860 Graph$="Y"
1870 PRINT "Do you wish to graph the turbulence quantities (Y/N, Default Y) ?", Graph$
1880 PRINTER IS 0
1890 PRINT "Normalizing velocity is:";Ue
1900 !
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-N+1+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 0,184.47,0,149.1
2200 LOCATE L(I),R(I),39,99
2210 CLIP L(I),R(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),Ulmin(I),Ymin,Ymax
2330 FRAME
2340 AXES Scale(I),1,0,0
2350 UNCLIP
2360 !
2370 LORG 0
2380 CSIZE 2.5
2390 Ypos=Ymin
2400 MOVE Llim(I)+Scale(I)/5,Ypos
2410 DRAW L1m(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 = LIm(I)
2510 MOVE Xpos,Ymin -.04
2520 LABEL DROUND(Xpos,2)
2530 MOVE Xpos,Ymin
2540 DRAW Xpos,Ymin +.08
2550 IF Xpos < Lim(I) THEN GOTO 2510
2560 !
2570 FOR J = 1 TO Nf-File +1
2580 Ypos = Point(J,1)
2590 FOR K = 1 TO 3
2600 ON K GOTO 2640,2700,2760
2610 MOVE Point(J,(I-1)*3+K+1),Ypos
2620 SETGU
2630 ON K GOTO 2640,2700,2760
2640 FOR Arc = 0 TO 360 STEP 20
2650 PDIR Arc
2660 RPL0T .5,0
2670 NEXT Arc
2680 PDIR 0
2690 GOTO 2810
2700 RPL0T .5,.5,-2
2710 RPL0T .5,-.5,-1
2720 RPL0T -.5,-.5,-1
2730 RPL0T -.5,.5,-1
2740 RPL0T .5,.5,-1
2750 GOTO 2810
2760 RPL0T .5,0,-2
2770 RPL0T 0,-.5,-1
2780 RPL0T .5,0,-2
2790 RPL0T -.5,0,-1
2800 GOTO 2810
2810 SETUU
2820 NEXT K
2830 NEXT J
2840 NEXT I
2850 DUMP GRAPHICS
2870 "uu/Uo2","/n"
2880 PRINT USING 2860
2890 FOR I = 1 TO Nf-File +1
2900 IMAGE MDD.DD,6X,MD.DDDD,2X,MD.DDDD,4X,MD.DDDD,2X,MD.DDDD,2X,MD.
2910 PRINT USING 2890;Point(I,1),Point(I,1),Point(I,1),Point(I,1),Point(I,7),Point(I,8),Po
2920 IMF(I,9),Point(I,10)
2930 NEXT I
2960 IMAGE MDD.DD,6X,MD.DDDD,2X,MD.DDDD,4X,MD.DDDD,2X,MD.DDDD,2X,MD.
2970 PRINT USING 2960
2980 FOR I = 1 TO Nf-File +1
2990 IMAGE MDD.DD,6X,MD.5D,2X,MD.5D,4X,MD.5D,2X,MD.5D,4X,MD.5D
3000 PRINT USING 2990;Point(I,1),Point(I,11),Point(I,12),Point(I,13),Point(I,14)
3010 PRINT USING 2920
3020 FOR I = 1 TO Nf-File +1
3030 PRINT USING 3020
3040 FOR I = 1 TO Nf-File +1
2970 NEXT I
2980 IF Sum$="N" THEN GOTO 3120
2990 ALTER SUMMARY DATA FILE
3000 IF Sum$="N" THEN GOTO 3020
3010 INPUT "Do you wish to write a Summary Data File (Y/N Default N) ?",Sum$
3020 IF Sum$="N" THEN GOTO 3210
3030 Files=Name$
3040 DISP "File ":Files," being written to disk"
3050 MASS STORAGE IS ":MB,0,0"
3060 CREATE Files,40
3070 ASSIGN Files TO #1
3080 PRINT #1;Date$
3090 PRINT #1;Title$
3100 PRINT #1;Name$
3110 PRINT #1;On
3120 PRINT #1;Hub,Numx1,Numx2,Numx3,Theta,Run
3130 PRINT #1;Vref,UE,DF1,DF2,DF3
3140 PRINT #1;Ns,Files,NF
3150 FOR I=1 TO NF-Nfile+1
3160 FOR J=1 TO 15
3170 FOR J=1 TO 15
3180 NEXT J
3190 NEXT J
3200 NEXT I
3210 RETURN
3220 IF Sum$="N" THEN GOTO 3660
3230 FOR I=1 TO 3
3240 FOR J=1 TO 1023 STEP 2
3250 Countbin(I,J)=Countbin(I,J)+Countbin(I,J+1)
3260 IF Countbin(I,J)>Cmax THEN Cmax=Countbin(I,J)
3270 NEXT J
3280 IF Cmax>0 THEN GOTO 3660
3290 LIMIT 0,184.47,0,149.8
3300 LOCATE 7.05,119.63,L1(K),U1(K)
3310 CLIP 7.05,119.63,L1(K),U1(K)
3320 K=K+1
3330 SCALE 1,1024,0,Cmax
3340 FRAME
3350 UNCLIP
3360 LINE TYPE 1
3370 CSIZE 2.8
3380 LORG 8
3390 FOR Y=1 TO 4
3400 Ypos=Y*Cmax/4
3410 PLOT 10,Ypos,-2
3420 PLOT -9,Ypos,-1
3430 MOVE 3,Ypos
3440 LABEL POUND(Ypos,0)
3450 NEXT Y
3460 FOR X=1 TO 8
3470 Xpos=X*128
3480 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,-Ypos
3640 DRAW J,Countbin(I,J)
3650 NEXT J
3660 NEXT I
3670 MOVE 512,-Ypos
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
3760 ISOURCE
3770 ISOURCE
3780 ISOURCE
3790 ISOURCE
3800 ISOURCE
3810 ISOURCE
3820 ISOURCE
3830 ISOURCE
3840 ISOURCE
3850 ISOURCE
3860 ISOURCE
3870 ISOURCE
3880 ISOURCE
3890 ISOURCE
3900 ISOURCE
3910 ISOURCE
3920 ISOURCE
3930 ISOURCE
3940 ISOURCE
3950 ISOURCE
3960 ISOURCE
3970 ISOURCE
3980 ISOURCE
3990 ISOURCE
4000 ISOURCE
4010 ISOURCE
4020 ISOURCE
4030 ISOURCE
4040 ISOURCE
4050 ISOURCE
4060 ISOURCE
4070 ISOURCE
4080 ISOURCE
4090 ISOURCE
4100 ISOURCE
4110 ISOURCE
4120 ISOURCE
4130 ISOURCE
4140 ISOURCE
4150 ISOURCE
4160 ISOURCE

NAM Find_vel

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 (Arrayd), the Bragg shift frequency (Nub), the mixing frequencies (Num1x1, Num1x2, Num1x3), the fringe spacings (Df1, Df2, Df3), the crossing angle of the third beam (Theta), and the number of samples in a data point (Ns).

The outputs are the summations of various products of the velocity components, including U, V, W, U*U, U*V, U*W, U*V, U*W, U*V, and U*V.

Data parms:
Hs_parm:
Hg_parm:
Cbin_parm:
Ran1_parm:
Ran2_parm:
Ran3_parm:
Sdev_parm:
Df1_parm:
Df2_parm:
Df3_parm:
Theta_parm:
Hub_parm:
Nmix1_parm:
Nmix2_parm:
Nmix3_parm:
Su_parm:
Sv_parm:
Su_parm:

Declare subroutines stored outside of the main program.

Declare common variables.
4170 ISOURCE Suu_par: REL
4180 ISOURCE Suv_par: REL
4190 ISOURCE Suw_par: REL
4200 ISOURCE Suw_par: REL
4210 ISOURCE Suw_par: REL
4220 ISOURCE Suw_par: REL
4230 ISOURCE Suw_par: REL
4240 ISOURCE Suw_par: REL
4250 ISOURCE Suw_par: REL
4260 ISOURCE Suw_par: REL
4270 ISOURCE Suw_par: REL
4280 ISOURCE Suw_par: REL
4290 ISOURCE Arrayd: BSS 39
4300 ISOURCE Elementd: EQU Arrayd+16
4310 ISOURCE Arrayl: BSS 4096
4320 ISOURCE Array2: BSS 4096
4330 ISOURCE Array3: BSS 4096
4340 ISOURCE Bin_u: BSS 1024
4350 ISOURCE Bin_v: BSS 1024
4360 ISOURCE Bin_w: BSS 1024
4370 ISOURCE Range1: BSS 1
4380 ISOURCE Range2: BSS 1
4390 ISOURCE Range3: BSS 1
4400 ISOURCE Na: BSS 1
4410 ISOURCE Dfl: BSS 4
4420 ISOURCE Df2: BSS 4
4430 ISOURCE Df3: BSS 4
4440 ISOURCE Thevai: BSS 4
4450 ISOURCE Hul: BSS 4
4460 ISOURCE Numx1: BSS 4
4470 ISOURCE Numx2: BSS 4
4480 ISOURCE Numx3: BSS 4
4490 ISOURCE Su: BSS 4
4500 ISOURCE Sv: BSS 4
4510 ISOURCE Sw: BSS 4
4520 ISOURCE Swu: BSS 4
4530 ISOURCE Swv: BSS 4
4540 ISOURCE Swu: BSS 4
4550 ISOURCE Swu: BSS 4
4560 ISOURCE Swu: BSS 4
4570 ISOURCE Swu: BSS 4
4580 ISOURCE Swu: BSS 4
4590 ISOURCE Swu: BSS 4
4600 ISOURCE Swu: BSS 4
4610 ISOURCE Swu: BSS 4
4620 ISOURCE Swu: BSS 4
4630 ISOURCE Count: BSS 1
4640 ISOURCE I: BSS 1
4650 ISOURCE Check: BSS 1
4660 ISOURCE Int: BSS 1
4670 ISOURCE In_2: BSS 1
4680 ISOURCE Address: BSS 1
4690 ISOURCE Offset: BSS 1
4700 ISOURCE R1: BSS 4
4710 ISOURCE R2: BSS 4
4720 ISOURCE R3: BSS 4
4730 ISOURCE Xvar: BSS 4
4740 ISOURCE Yvar: BSS 4
4750 ISOURCE U: BSS 4
4760 ISOURCE V: BSS 4

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.
4779  ISOURCE  W: BSS 4
4789  ISOURCE  Uu: BSS 4
4799  ISOURCE  Uv: BSS 4
4809  ISOURCE  Cos: BSS 4
4819  ISOURCE  Sin: BSS 4
4829  ISOURCE  Max_u: BSS 1
4839  ISOURCE  Max_v: B"3 1
4849  ISOURCE  Max_w: BSS 1
4859  ISOURCE  Avg_u: BSS 4
4869  ISOURCE  Avg_v: BSS 4
4879  ISOURCE  Avg_w: BSS 4
4889  ISOURCE  Dev_u: BSS 4
4899  ISOURCE  Dev_v: BSS 4
4909  ISOURCE  Dev_w: BSS 4
4919  ISOURCE  Ngs: BSS 4
4929  ISOURCE  Sdev: BSS 4
4939  ISOURCE  Rad: DAT 5.729578E1
4949  ISOURCE  Mill: DAT 1.6E6
4959  ISOURCE  One: DAT 1.
4969  ISOURCE  Zero: DAT 0.
4979  ISOURCE  LIT 80
4989  ISOURCE
4999  ISOURCE  SUB
5009  ISOURCE  Find_vel: LDA =N
5019  ISOURCE  LDB =N_par
5029  ISOURCE  LSM Get_value
5039  ISOURCE  LDA =Rayd
5049  ISOURCE  LDB =Data_par
5059  ISOURCE  LSM Get_info
5069  ISOURCE  LDA =Sdev
get number of samples.
5079  ISOURCE  LDB =Sdev_par
5089  ISOURCE  LSM Get_value
5099  ISOURCE  LDA =Df1
5109  ISOURCE  LDB =Df1_par
5119  ISOURCE  LSM Get_value
5129  ISOURCE  LDA =Df2
5139  ISOURCE  LDB =Df2_par
5149  ISOURCE  LSM Get_value
5159  ISOURCE  LDA =Df3
5169  ISOURCE  LDB =Df3_par
5179  ISOURCE  LSM Get_value
5189  ISOURCE  LDA =Theta
5199  ISOURCE  LDB =Theta_par
5209  ISOURCE  LSM Get_value
5219  ISOURCE  LDA =Nu6
5229  ISOURCE  LDB =Nu6_par
5239  ISOURCE  LSM Get_value
5249  ISOURCE  LDA =Num1x1
5259  ISOURCE  LDB =Num1x1_par
5269  ISOURCE  LSM Get_value
5279  ISOURCE  LDA =Num2x2
5289  ISOURCE  LDB =Num2x2_par
5299  ISOURCE  LSM Get_value
5309  ISOURCE  LDA =Num1x3
5319  ISOURCE  LDB =Num1x3_par
5329  ISOURCE  LSM Get_value
5339  ISOURCE  the loop headed by Get_freq is repeated three times to get
5349  ISOURCE  the count-to-frequency conversion factors (which depend on
5359  ISOURCE  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.

Get the first word of the column of
the data array which contains the
velocity component for which we want
to get the range.

Mask and rotate to get the four
bits containing the range.

Subtract from 15 to get the
actual range.

Store the actual range in RangeH
so that we can transfer it to the
main program

Use the range to find the power
of two needed for the divisor.

Convert the power of two into a
real number.

Divide 3.2E4 by the appropriate
power of two, using BCD math.

Decide whether to put the result
in R1, R2, or R3, depending on Count

How, finally, call the utility to
perform the division.

Increment and check Count so as
to follow the loop three times.

Zero out the entire count-to-
velocity conversion table so that it must be recalculated for each
point. (This must be done if the
mixing frequency or ranges are
changed between counts.)
LDA =Su
LDB =15
CLR 4
ADA =4
DSZ B
JMP Clear
LDA =Bin_u
LDB =192
CLR 16
ADA =16
DSZ B
JMP Clear2
LDA =Theta
STA Oper_1
LDA =Rad
STA Oper_2
LDA =Xvar
STA Result
LDA =1
LDB =147155B
JSM Rel_math
LDA =Xvar
STA Oper_1
LDA =Sin
STA Result
LDA =1
LDB =34213B
JSM Rel_math
LDA =Cos
STA Result
LDA =1
LDB =34224B
JSM Rel_math
LDA =3
LDA Count
STA Count
LDA Count
ADA =-1
SAL 2
LDA =Hub
STA Oper_1
LDA =Numx1
ADA Offset
STA Oper_2
STA Result
LDA =2
LDB =146717B
JSM Rel_math
LDA =DfT
STA Oper_1
LDA =NHub
STA Oper_2
LDA =NumxN
NHub =NumxN
Hub =Numx1
Offset
Find NumxN=NumxN.
Find Numx1=NumxN.
Find Numx1=NHub.
Find DfT=MILL+DfN.
Find DfT=MILL+DfN.
Find NumxN=NHub=NumxN.
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 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.

Next 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.

Figure out which element of the data array we want to pick up.

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

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

Store the count we have gotten.

Go to the address in the array corresponding to the value of the count.

Increment the word by one, indicating that one more count with that value has been read.

When Sdev>8, use this section to filter the data.
7170 ISOURCE: The mean and standard deviation of the data are calculated.
7180 ISOURCE: The program then throws out all the data which are farther than
7190 ISOURCE: $\text{Sdev}$ standard deviations from the mean.
7200 ISOURCE:
7210 ISOURCE Check_dev: LDB =Sdev
7220 ISOURCE ADB =1
7230 ISOURCE LDA B.I
7240 ISOURCE RZA =2
7250 ISOURCE JMP Set_begin
7260 ISOURCE
7270 ISOURCE: If $\text{Sdev} > 0$, use this section to
7280 ISOURCE: filter the data. If not, continue
7290 ISOURCE: with the program.
7300 ISOURCE Get_avg: LDA =0
7310 ISOURCE STA Count
7320 ISOURCE LDA =Zero
7330 ISOURCE STA Court
7340 ISOURCE SBL 2
7350 ISOURCE ADB =Avg_u
7360 ISOURCE XFR 4
7370 ISOURCE LDA Count
7380 ISOURCE ADB =1
7390 ISOURCE LDB =1024
7400 ISOURCE MPY
7401 ISOURCE RDA =Bin_u
7410 ISOURCE STA Address
7411 ISOURCE LDA =1023
7420 ISOURCE STA 1
7421 ISOURCE: Get a count and convert it to
7430 ISOURCE: a real number.
7440 ISOURCE Sum_count: LDA =1
7441 ISOURCE STA Oper_1
7442 ISOURCE LDA =Xvar
7443 ISOURCE STA Result
7444 ISOURCE JSM Int_to_rel
7445 ISOURCE
7446 ISOURCE: Use the Bin_u arrays to find out how
7447 ISOURCE: many samples there are with that
7448 ISOURCE: count and convert that number to a
7449 ISOURCE: real number.
7450 ISOURCE Get_count: LDA Address
7451 ISOURCE STA Oper_1
7452 ISOURCE LDA =Yvar
7453 ISOURCE STA Result
7454 ISOURCE JSM Int_to_rel
7455 ISOURCE
7456 ISOURCE: Multiply the count by
7457 ISOURCE: the number of times it appears in
7458 ISOURCE: the data.
7459 ISOURCE Count := LDA =Xvar
7460 ISOURCE STA Oper_1
7461 ISOURCE STA =Yvar
7462 ISOURCE STA Oper_2
7463 ISOURCE STA Result
7464 ISOURCE STA =2
7465 ISOURCE LDA =Source
7466 ISOURCE LDB =1478379
7467 ISOURCE JSM Rel_math
7468 ISOURCE
7469 ISOURCE: Keep a running sum of the product of
7470 ISOURCE: the count times the number of times
7471 ISOURCE: that it appears. Store this sum in
7472 ISOURCE: the place we will eventually use to
7473 ISOURCE: store the average.
7474 ISOURCE LDB Count
7475 ISOURCE SBL 2
7476 ISOURCE ADB =Avg_u
7477 ISOURCE STA Oper_1
7478 ISOURCE STA =Yvar
7479 ISOURCE STA Oper_2
7480 ISOURCE STA =2
7481 ISOURCE LDA =Source
7482 ISOURCE LDB =1467219
7483 ISOURCE JSM Rel_math
7484 ISOURCE
7485 ISOURCE: DSZ Address
7486 ISOURCE DSZ 1
7487 ISOURCE JMP Sum_count

55
Conversion of 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 to 0.

Get a count and convert it to a real number.

Subtract the average count from the count we just got.
LDA =Yvar  
STA Oper_1  
STA Oper_2  
LDA =Xvar  
STA Result  
LDA =2  
LDB =147037B  
JSM Rel_math  
LDA Address  
STA Oper_1  
LDA =Yvar  
STA Oper_2  
STA Result  
LDA =2  
LDB =147037B  
JSM Rel_math  
LDA =Yvar  
STA Oper_1  
LDA =Yvar  
STA Result  
LDA =2  
LDB =147037B  
JSM Rel_math  
LDA =Yvar  
STA Oper_1  
LDA =Yvar  
STA Result  
LDA =2  
LDB =146721B  
JSM Rel_math  
DSZ Address  
DSZ 1  
JMP Dev_count  
LDA =Ns  
STA Oper_1  
LDA =Xvar  
STA Result  
JSM Int_to_rel  
LDB Offset  
ADB =Dev_u  
STB Oper_1  
STB Result  
LDB =Xvar  
STB Oper_2  
LDA =2  
LDB =147155B  
JSM Rel_math  
LLA =Dev_u  
RDR Offset:  
STA Oper_1  
STA Result  
LDA =1  

378 SOURCE
380 SOURCE
380 SOURCE
390 SOURCE
400 SOURCE
410 SOURCE
420 SOURCE
430 SOURCE
440 SOURCE
450 SOURCE
460 SOURCE
470 SOURCE
480 SOURCE
490 SOURCE
500 SOURCE
510 SOURCE
520 SOURCE
530 SOURCE
540 SOURCE
550 SOURCE
560 SOURCE
570 SOURCE
580 SOURCE
590 SOURCE
600 SOURCE
610 SOURCE
620 SOURCE
630 SOURCE
640 SOURCE
650 SOURCE
660 SOURCE
670 SOURCE
680 SOURCE
690 SOURCE
700 SOURCE
710 SOURCE
720 SOURCE
730 SOURCE
740 SOURCE
750 SOURCE
760 SOURCE
770 SOURCE
780 SOURCE
790 SOURCE
800 SOURCE
810 SOURCE
820 SOURCE
830 SOURCE
840 SOURCE
850 SOURCE
860 SOURCE
870 SOURCE
880 SOURCE
890 SOURCE
900 SOURCE
910 SOURCE
920 SOURCE
930 SOURCE
940 SOURCE
950 SOURCE
960 SOURCE

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

I 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.

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

I 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.

I Convert the number of samples to a real number.

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

I Take the square root of the variance to get the standard deviation.
Now use the standard deviation to filter out all the counts whose value is more than Sdev standard deviations away from the mean.

8970 ISOURCE LDB =31453B
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 SRA ++2
9050 ISOURCE JMP Get_sdev
9060 ISOURCE
9070 ISOURCE LDA =0
9080 ISOURCE STA Count
9090 ISOURCE Dev_filter LDA Count
9100 ISOURCE ADA =1
9110 ISOURCE LDB =1024
9120 ISOURCE Mpy
9130 ISOURCE ADA =Bin_u
9140 ISOURCE STA Address
9150 ISOURCE LDA =1023
9160 ISOURCE STA I
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 =147837B
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 LDB =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 SRA ++2
9490 ISOURCE Tca
9500 ISOURCE ADA Int
9510 ISOURCE SRA ++3
9520 ISOURCE LDA Address
9530 ISOURCE CLR 1
9540 ISOURCE DSZ Address
9550 ISOURCE DSZ I
9560 ISOURCE JMP Filtr_dev
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 it 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.

Set_begin: LDA = 0

LDA Ns

STA

STA

STA

SBL 2

STB Offset

LT: Ns

MPY

ADA I

ADA = -1

LDA = Int

Get a raw datum from the data array.

LDB = Arrayd

JSR Get_element

LDA Int

LDB = 1023

Strip off the first six

LDB Count

See if Count = Check.

TCB

ADB Check

S2B Straight

If true, use the modified data

TCA

word as an index. If not, use

AD = 1024

1824 minus the data word.

STB:

The count has been gotten.

LDA Count

LDB = 1024

Now look in the appropriate part

LDB Count

of the arrays created by the filter

MPY

section.

ADA Int

ADA = -1

ADA = Bin_u

LDB A, I

R2B Goodcount

If the word corresponding to the

STB Count

count is zero, the count was

DSZ Ngs

filtered out and should be ignored.

DSZ I

Ignore these three counts, go back

JMP Begin

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 in the table we want to go.

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

Convert the count into a real number.

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

Find (Hub-Hum)xN-Frequency/N.

If we are calculating U, reverse the sign of ((Hub-Hum)xN)-Frequency, so as to reverse the sign of U. Leave V and W alone.

Find Velocity=((Hub-Hum)xN)-(Frequency/N)*MxDFH) and store in a place in the lookup table corresponding to
10770 ISOURCE STR Oper_2 1 the data count.
10780 ISOURCE LDA Address
10790 ISOURCE ADA =-1
10800 ISOURCE STR Result
10810 ISOURCE LDA =-2
10820 ISOURCE LDB =147037B
10830 ISOURCE JSM Rel_math
10840 ISOURCE Over:
10850 ISOURCE LDA Address 1 Transfer the velocity from
10860 ISOURCE ADA =-1 1 the lookup table to U, V, or W,
10870 ISOURCE LDB =U 1 as appropriate.
10880 ISOURCE ADA Offset
10890 ISOURCE XFR 4
10900 ISOURCE
10910 ISOURCE ISZ Count
10920 ISOURCE LDA =-3 1 Have U, V, and W all
10930 ISOURCE ADA Count 1 been calculated?
10940 ISOURCE SRA =+2 1 If not, go back again.
10950 ISOURCE JMP Begin
10960 ISOURCE STA Count 1 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=(W+V*Cos(Theta))/Sin(Theta).
11020 ISOURCE
11030 ISOURCE LDA =W
11040 ISOURCE STR Oper_1
11050 ISOURCE LDA =Cos
11060 ISOURCE STR Oper_2
11070 ISOURCE LDA =Xvar
11080 ISOURCE STA Result
11090 ISOURCE LDA =2
11100 ISOURCE LDB =147037B
11110 ISOURCE JSM Rel_math
11120 ISOURCE LDA Result
11130 ISOURCE STR Oper_2
11140 ISOURCE LDA =W
11150 ISOURCE STR Oper_1
11160 ISOURCE LDA =Yvar
11170 ISOURCE STR Result
11180 ISOURCE LDA =2
11190 ISOURCE LDB =146717B
11200 ISOURCE JSM Rel_math
11210 ISOURCE LDA Result
11220 ISOURCE STR Oper_1
11230 ISOURCE LDA =Sin
11240 ISOURCE STR Oper_2
11250 ISOURCE LDA =W
11260 ISOURCE STR 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
LDA =Su  \quad \text{Find Su} = Su + U \\
LDB =U \\
JSM Add \\

LDA Oper_2 \quad \text{Find U}\times U \\
STA Oper_1 \\
LDA =Uu \\
STA Result \\
LDA =2 \\
LDB =147037B \\
JSM Rel_math \\

LDA =Suu \quad \text{Find Suu} = Suu + (U \times U) \\
LDB Result \\
JSM Add \\

LDA =V \quad \text{Find U} \times U \times V \\
STA Oper_1 \\
LDA =Uu \\
STA Result \\
LDA =2 \\
LDB =147037B \\
JSM Rel_math \\

LDA =Suuv \quad \text{Find Suuv} = Suuv + (U \times U \times V) \\
LDB Result \\
JSM Add \\

LDA =Uu \quad \text{Find U} \times U \times W \\
STA Oper_1 \\
LDA =W \\
STA Oper_2 \\
LDA =Uu \\
STA Result \\
LDA =2 \\
LDB =147037B \\
JSM Rel_math \\

LDA =Suuw \quad \text{Find Suuw} = Suuw + (U \times U \times W) \\
LDB Result \\
JSM Add \\

LDA =W \quad \text{Find Su} = Su + V \\
LDB =V \\
JSM Add \\

LDA =U \quad \text{Find U} \times V \\
STA Oper_1 \\
LDA =Uu \\
STA Result \\
LDA =2 \\
LDB =147037B \\
JSM Rel_math \\

LDA =Suv \quad \text{Find Suv} = Suv + (U \times V) \\
LDB Result \\
JSM Add \\

LDA =Suv \quad \text{Find Suv} = Suv + (U \times V) \\
LDB Result \\
JSM Add \\

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

64
13170 ISOURCE LDB =Suv_par
13180 ISOURCE JSR Put_Value
13190 ISOURCE LDA =Suv
13200 ISOURCE LDB =Suv_par
13210 ISOURCE JSR Put_Value
13220 ISOURCE LDA =Suv
13230 ISOURCE LDB =Suv_par
13240 ISOURCE JSR Put_Value
13250 ISOURCE LDA =Suv
13260 ISOURCE LDB =Suvu_par
13270 ISOURCE JSR Put_Value
13280 ISOURCE LDA =Suv
13290 ISOURCE LDB =Suvu_par
13300 ISOURCE JSR Put_Value
13310 ISOURCE LDA =Suvu
13320 ISOURCE LDB =Suvu_par
13330 ISOURCE JSR Put_Value
13340 ISOURCE LDA =Suvu
13350 ISOURCE LDB =Suvu_par
13360 ISOURCE JSR Put_Value
13370 ISOURCE LDA =Suvu
13380 ISOURCE LDB =Suvu_par
13390 ISOURCE JSR Put_Value
13400 ISOURCE LDA =Suvu
13410 ISOURCE LDB =Suvu_par
13420 ISOURCE JSR Put_Value
13430 ISOURCE LDA =Suvu
13440 ISOURCE LDB =Suvu_par
13450 ISOURCE JSR Put_Value
13460 ISOURCE LDA =Ngs
13470 ISOURCE LDB =Ngs_par
13480 ISOURCE JSR Put_Value
13490 ISOURCE LDA =Bin_u
13500 ISOURCE LDB =192
13510 ISOURCE STB 1
13520 ISOURCE LDB =16
13530 ISOURCE STB Count
13540 ISOURCE LDB Chn_par
13550 ISOURCE Transfer: XFR 16
13560 ISOURCE ADA Count
13570 ISOURCE ADD Count
13580 ISOURCE DSZ 1
13590 ISOURCE JMP Transfer
13600 ISOURCE RET 1
13610 ISOURCE LIT 280
13620 ISOURCE LIT 280
13630 ISOURCE LIT 280
13640 ISOURCE LIT 280
13650 ISOURCE The utility "Add" is used to add up the running sums.
13660 ISOURCE LIT 280
13670 ISOURCE LIT 280
13679 ISOURCE Add: JSR Ulicount
13680 ISOURCE STA Oper_1
13690 ISOURCE STA Oper_2
13700 ISOURCE STA Result
13710 ISOURCE LDA =2
13720 ISOURCE LDB =146721B
13730 ISOURCE JSR Rel_math
13740 ISOURCE DSZ Ulicount
13750 ISOURCE RET 1
13760 ISOURCE LIT 2130
13770 ISOURCE Th. 1-1
13770 SOURCE 1
13780 SOURCE END Find_vel
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

LENS f = 15 in.

MIRROR

DICHROIC FILTER (SPLITS GREEN) (LINE)

LENS f = 15 in

MEASURING VOLUME

FOUR BEAM MATRIX FROM OPTICS TARI F
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.

78
(a) Overall schematic

(b) Details of boundary layer trips and coordinate system

Fig. 11 Experimental rig.
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.
END
DATE
FILMED

NOV 13 1986
End of Document